Denial-of-sleep attacks on IoT networks

Emilie Bout



Université de Lille

École Doctorale Mathématiques, sciences du numérique et de leurs interactions

Unité de Recherche Inria Lille - Nord Europe

Thèse préparée par **Emilie Bout**
en vue de l'obtention du grade de **Docteur en Informatique**
Discipline **Informatique et Applications**

Attaque par déni de sommeil sur les reseaux IoT

Thèse soutenue le 7 octobre 2022 devant le jury composé de :

Virginie Deniau	**Examinateur, Président**
Directrice de Recherche, Université Gustave Eiffel	
Abderrahim Benslimane	**Examinateur**
Professeur des Universités, Université d'Avignon	
Mauro Conti	**Examinateur**
Professeur des Universités, Université de Padova	
Hicham Lakhlef	**Rapporteur**
Maître de conférences, HDR, Université de Compiègne	
Nicolas Montavont	**Rapporteur**
Professeur, IMT Atlantique	
Valeria Loscri	**Directrice de thèse**
Chercheuse Permanent, HDR, Inria	
Antoine Gallais	**Co-Directeur de thèse**
Professeur des Universités, INSA	



University of Lille
Doctoral School MAthematics and DIgital Sciences
Research Institute Inria Lille - Nord Europe

Thesis prepared by **Emilie Bout**
with the view of obtaining the degree of **PhD in Computer Science**
Discipline **IT and Applications**

Denial-of-Sleep Attacks on IoT Networks

Thesis defended the 7 October 2022 in-front of the jury comprised of:

Virginie Deniau Research Director, University Gustave Eiffel	**Examinator, President**
Abderrahim Benslimane Full Professor, University of Avignon	**Examinator**
Mauro Conti Full Professor, University of Padova	**Examinator**
Hicham Lakhlef University Lecturer, HDR, University of Compiègne	**Reviewer**
Nicolas Montavont Full Professor, IMT Atlantique	**Reviewer**
Valeria Loscri Permanent Researcher, HDR, Inria	**Supervisor**
Antoine Gallais Full Professor, Inria	**Co-Supervisor**

Remerciements

Tout d'abord, je tiens à exprimer ma profonde gratitude a tous les membres du jury pour leur différents retours qui m'ont permis d'améliorer ce manuscrits. Je remercie mes deux rapporteurs, Dr. Hicham Lakhlef et Dr. Nicolas Montavont, d'avoir accepté de relire en détail ce manuscrit ainsi que ma profonde gratitude aux Dr. Mauro Conti et Dr. Abderrahim Benslimane pour avoir accepté d'examiner mes travaux. Merci également au Dr. Virginie Deniau pour avoir présidé mon jury de thèse. Le temps est précieux, et je vous serez toujours reconnaissant d'avoir employé ce temps pour mes travaux.

Je tiens également à remercier ma directrice de thèse Dr. Valeria Loscri pour son soutien et sa direction, sa disponibilité, sa réactivité ainsi que son écoute. A ses cotés j'ai pu énormément évoluer que ce soit de manière professionnelle ou humaine. Je tiens également à remercier mon co-directeur de thèse Dr. Antoine Gallais pour ces trois années a mes cotés, qui a pu apporter une autre vision lorsque le besoin s'en faisait ressentir. C'était un très grand plaisir de travailler avec la Direction Générale de l'Armement sans qui cette thèse n'aurait pas pu voir le jour. Dr. Adeline Bailly, Mme Chantal CAUDRON de COQUEREAUMONT, Dr. Véronique Serfaty et M Paul Le Guen, merci pour vos différents échanges et vos conseils. Bien évidemment, j'adresse de chaleureux remerciements a l'INRIA pour son accueil mais plus particulièrement a l'ensemble des membres de l'équipe FUN qui ont été présents durant ma thèse pour leurs réflexions scientifiques et leurs connaissances. Plus précisément a Edward Staddon, qui a debute sa these le meme jour, et qui a été d'un réel soutien moralement et scientifique durant ces trois ans.

Je tiens aussi à remercier toutes les personnes avec qui j'ai pu collaborer durant ma thèse. Merci a l'équipe SPRITZ de m'avoir accueilli durant ces deux mois et plus particulièrement aux Dr. Mauro Conti et Dr. Alessandro Brighente avec qui j'ai pu échanger avec plaisir de nombreuses idées. Je tiens aussi à remercier l'ISIT et L université de Lille pour m'avoir accorder deux bourses permettant ce séjour. Sans eux, cette collaboration entre nos deux laboratoires n'aurait pas pu aboutir. Je tiens également à exprimer mes remerciements à l'Université Gustave Eiffel avait qui j'ai eu l'occasion de collaborer et d'échanger sur des idées.

Pour finir, merci également à ma famille pour m'avoir toujours soutenu, encouragé et supporté dans mes choix (parfois non-censés) et pour leur précieuse présence pendant ces trois années. Sans oublier mon mari, Sid Ali, qui a été d'une patiente, d'une écoute et d'un encouragement pendant toutes les différentes phases de ma thèse. Je pense que je n'aurai jamais les mots pour lui dire assez merci pour tout le soutien qu'il a pu m'apporter.

Résumé

Ces dernières années, les réseaux de l'Internet des Objets (IoT) sont devenus les nouvelles cibles privilégiées des attaquants. Leur caractéristique fondamentale telle que leurs contraintes énergétiques et de calculs ont ouvert de nouveaux vecteurs d'attaques. Dans cette thèse, nous nous concentrons sur l'etude de vulnerabilites presents dans les réseaux sans fils afin de créer des frameworks permettant de lancer plusieurs types d'attaques. Nous montrons également comment ces frameworks peuvent aussi etre detournés en système de defense.

En parallèle, le paysage des menaces est en train de changer considérablement, et de nouvelles attaques, communément appelées attaques intelligentes, sont en train d'émerger grâce à l'utilisation de nouveaux procédés comme l'apprentissage automatique. Les attaquants sont maintenant capables de créer des attaques plus autonomes, robustes et efficaces qui arrivent à contourner les systèmes de détections et de contre-mesures actuels. C'est pourquoi, étudier la sécurité des réseaux sans fils en face de ces attaques nouveaux types d'attaque pour mieux les comprendre est devenu un enjeu important dans la recherche. Dans cette thèse, nous évaluons plusieurs vulnérabilités présentes dans les réseaux sans fils permettant de créer de nouvelles attaques intelligentes. Dans un premier temps, nous développons HARPAGON, un framework basé sur la théorie des chaînes de Markov et exploitant les vulnérabilités générées par le mécanisme du cycle d'utilisation. Le principal avantages d'HARPAGON est de prédire le moment optimal pour effectuer son attaque afin de réduire sa probabilité d'être détecté. Dans un même temps ce framework permet également a l'attaquant de conserver son énergie. Puis nous proposons un autre framework nommé FOLPETTI permettant de créer plusieurs types d attaques déjouant une contre-mesure bien connu dans les réseaux sans fils: le saut de canal. Nous montrons qu'avec l'aide de FOLPETTI, un attaquant est capable de prédire le futur canal de transmission afin d'augmenter son impact. Afin d'évaluer leur efficacité, nous avons développé un nouveau module sur le simulateur NS-3 permettant de simuler les attaques de brouillages. Puis, après avoir validé leurs comportements sur le simulateur, nous les avons évaluées grâce à des expérimentations sur un réel banc d'essais. Ces deux solutions, qui permettent d'augmenter les performances de plusieurs attaques, ne requièrent pas de connaissance au préalable de la part de l'attaquant et peuvent être implémenté sur des composants bons marchés.

Enfin, fortement inspirés des frameworks FOLPETTI et HARPAGON, nous avons implémenté une nouvelles attaques de brouillages, nomme ICARO, qui vise les drones illicites. Dans ce cas, nous montrons comment une attaque de brouillage peut être détournée en méthode de défense pour contrer des drones survolants des zones illicites. Le principal avantage de cette contribution est que ce nouveau type d'attaque permet de couper la communication d'un drone illicite avec son contrôleur sans sans perturber les communications aux alentours qui communiquent dans les mêmes fréquences.

Abstract

In recent years, Internet of Things (IoT) networks have become new favorite targets for attackers. Their fundamental characteristic such as their energy and calculation constraints are open to new attack vectors. In this thesis, we focus on the study of vulnerabilities present in wireless networks in order to create frameworks allowing to launch several types of attacks. We also show how these frameworks can also be used as a defense system.

At the same time, the threat landscape is changing dramatically and new attacks, commonly referred to as smart attacks, are emerging through the use of new processes like machine learning. Attackers are now able to create more autonomous, robust and efficient attacks that manage to advance current detection and countermeasure systems. This is why studying the security of wireless networks in the face of these new types of attacks to better understand them has become an important issue in research. In this thesis, we evaluate several vulnerabilities present in wireless networks allowing to create new intelligent attacks. First, we are developing HARPAGON, a framework based on the Markov chains theory and exploiting the vulnerabilities generated by the duty cycle mechanism. The main advantage of HARPAGON is to predict the optimal moment to carry out its attack in order to reduce its probability of being detected. At the same time this framework also allows the attacker to conserve energy. Then we propose another framework called FOLPETTI allowing to create several types of attacks thwarting a well-known countermeasure in wireless networks: channel hopping. We show that with the help of FOLPETTI, an attacker is able to predict the future transmission channel in order to increase its impact. In order to evaluate their effectiveness, we have developed a new module on the NS-3 simulator to simulate jamming attacks. Then, after validating their components on the simulator, we assigned them through experimentation on a real testbed. These two solutions; which increase the performance of several attacks, do not require prior knowledge on the part of the attacker and can be implemented on inexpensive components.

Finally, strongly inspired by the FOLPETTI and HARPAGON frameworks, we have implemented a new jamming attack, called ICARO, which targets illicit drones. In this case, we show how a jamming attack can be diverted as a defense method to counter drones flying over illicit areas. The main advantage of this contribution is that this new type of attack makes it possible to cut off the communication of an illicit drone with its controller without disrupting communications in the surrounding area.

Contents

List of Figures

List of Tables

Introduction 1

1.1 Wireless communication

The term "computer networks" first appeared in the 1950s and has now become an ordinary reality. These words designate a set of computer devices interconnected by a communication protocol that can exchange information between them. Thanks to technological advances, the definition of a "computer network" has enlarged and various communication paradigms have been developed. Indeed, initially wired networks have seen the emergence of a new category: wireless networks which, as its name suggests, represents a non-wired mode of communication. So much so that nowadays it has become unthinkable to conceive of a world without wireless communications. Users demand mobility, broadband and access to multimedia at all times. In order to meet user requirements and for many domain cases, wireless network standards have evolved and many protocols have been developed. Indeed, to cite just a few examples, the WiFi (802.11) protocol has been implemented to allow the operation of internal networks. The personal network of a house uses this protocol most of the time, and it has become so ingrained in our habits that it is inconceivable to envisage a residence without this installation. More than a comfort, it has become a real need to be able to perform administrative or work tasks. Additionally, to meet the need to instantly share content with a nearby user, the Bluetooth protocol was created to establish short-range wireless communication between two devices. The last example can be represented by the development of contactless payment which came with the invention of the radio frequency identification protocol (RFID). This type of protocol is a technology that uses electromagnetic fields to identify objects or tags which contains some stored information. As a result, the constant evolution of these protocols, as well as their advantages such as ease of installation, deployment and maintenance, have made it possible to increase the number of the user and connected objects on a large scale. These connected devices have been grouped into an object class called the Internet of Things (IoT).

1.2 The Internet of Things (IoT) networks

The Internet of Things (IoT) networks refer to a type of network which allows any object to be connected to each other using communication protocols. These objects have become ubiquitous in our daily lives, ranging from our mobile

phones to our watches and our transport (connected cars, connected scooters, etc.). The number of connected devices is estimated to exceed 500 billion by 2030 according to Cisco [1]. Broadly, an IoT network is a network that satisfies at least one of these several criteria such as interconnectivity, heterogeneity, dynamic changes, huge scale, security and connectivity, as explained in [2]. This type of device has been applied in several contexts as summarized in Fig 1.1 and explained below:

Figure 1.1: The different areas of IoT application.

- **Smart Environment:** This class includes smart cities and smart homes. The integration of sensors like cameras or intelligent objects (e.g. smart trash, home automation plug) has progressively improved certain points such as security, energy consumption or the hassle-free lifestyle. Singapore, Oslo, and Zurich are examples of smart cities [3].
- **Smart Agriculture:** Certain parameters in agriculture such as temperature, soil humidity, atmosphere or air pollution can be easily detected nowadays thanks to sensors. Therefore, many connected measurement devices have been developed in agriculture to monitor such data and act on them. The animal/farming tracking has also developed in the agricultural world by installing GPS chips on the animals to know their position and find them easily. Finally, the IoT additionally allows to easily manage logistics issues (stock, optimization)[4].
- **Smart Transport:** Thanks to IoT, the transportation sector has been revolutionized with the generation of Intelligent Transportation Systems (ITS) allowing the optimization of logistics and fleet management, providing new goods and services, transport control, driver assistance, etc. Indeed, problems such as how to reduce the traffic congestion or the impact of transport on the climate can be solved through the use of IoT networks [5].
- **Industry 4.0:** A new area of industry also called the 4th industrial revolution is currently ongoing. Its goal is to design an interconnected structure in which machines, systems and products will communicate continuously. IoT, therefore, has an essential role to play in this transformation to be able to collect necessary information such as the hours of use of a machine or the number of start-up cycles. This interconnection would make the industry more customized and increase productivity [6].
- **Health and Sport:** IoT is also incrementally present in the field of sport thanks to connected watches but also in the medical domain thanks to real-time blood glucose sensors which make it possible to alert doctors and to react instantly. The use of these devices has contributed to make the medical world more reactive to anomalies and to detect diseases quicker [7].
- **Aerospace and Defense 4.0:** As in the industrial world, a

new military era is underway. IoT can help defense in six ways: provide situational awareness on the battlefield, proactively maintain equipment, monitor a combatant's health, conduct remote training, and furnish real-time inventory. This modern strategy relies more on sensors to improve analysis and decision-making [8].

- **Smart Energies:** Limiting energy consumption is an increasingly important issue. The Smart Grid is seen as an eventual solution to address this problem. It is an intelligent electrical network that circulates information between the various players in the electrical system - supplier, distributors and consumers - in order to manage the electrical network. Smart Grids integrate energy and electricity systems with Information and Communication Technologies (ICT) and make it possible to ensure, at all times, the balance between power supply and demand [9].

1.3 IoT networks and Security

The security of IoT networks remains a crucial point in the field of research and in perpetual evolution. Indeed, this type of equipment represents an essential attack vector, and this is due to the large amount of private and sensitive data they convey. Moreover, as we have seen just above, the growing use of IoT in critical infrastructures such as the hospital or the military field requires an in-depth study of security. Indeed, an attack on IoT networks can cause significant damage beyond the digital world. For example, in December 2015, an attack on a Ukrainian IoT power grid left more than 230,000 users without electricity for more than three hours [10]. A research firm in Florida has demonstrated that the St. Jude hospital cardiac devices are subject to two major vulnerabilities that can put a patient's life at risk 1) a "crash" attack that can cause the device to disable communication; 2) "Battery drain" attack which can waste the energy of the device and makes it out-of-service [11]. Finally, it has been proven in [12] that it is possible to take control of a Tesla by targeting these IoT components. Consequently, attacks on IoT networks have multiplied in recent years and produce severe consequences on our world, potentially endangering a person's life.

Securing these systems remains a major challenge for several reasons. Indeed, on the connectivity side, there is not just one communication protocol dedicated to IoT networks. There are at least 10 communication protocols allowing the interconnection of objects: common protocols such as WiFi or Bluetooth protocols or more specific such as the LoRaWAN protocol [13]. As already mentioned, these protocols have been developed to meet several needs such as long distance range or low energy consumption. As a result, due to the

heterogeneity of these communication protocols, it becomes more complicated and tedious to perform security analysis on these networks. In addition, to complicate security analyses, there is not only one firmware or software for an identical object. Indeed, each manufacturer develops its own logic and its own software for each type of device for questions of needs and economics. For example, drones developed by Parrot or DJI firms use two different communication protocols and the application available on the phone to pilot the drone is not the same depending on the company, as shown in Fig 1.2. Thus, to study the security of a drone device, researchers or security analysts must carry out numerous experiments taking into account the diverse types of communication protocols and applications.

	Parrot: Anafi	**DJI**: Mini Pro 3
Type		
App	FreeFlight 6	DJI Fly
Controller	Wi-Fi 802.11a/b/g/n	ISM band proprietary protocol

Figure 1.2: Difference parameters between two drones

Finally, IoT devices are by definition constrained devices, whether in terms of energy or computing capacity or both. These specificities have created new attack vectors. Indeed, a new subclass targeting the batteries of IoT devices in order to deplete energy and put these objects out of service has emerged: the energy depletion attacks. They are also known as denial-of-sleep attacks. To conclude, the heterogeneity of IoT communication protocols, as well as their firmware and software and their physical constraints, make it difficult to implement security systems. Therefore, finding a single security solution for multiple types of devices is an extremely, very complicated challenge.

1.4 Thesis Statement

IoT networks have become prime targets for attackers, and research in security for IoT has increased in recent years [14]. Detection systems have become increasingly robust and autonomous and can sometimes target several types of attacks at the same time. This improvement was possible in part with the integration of the Machine Learning (ML) algorithms into the detection systems. However, like any technological advancement, ML can also be used for malicious purposes. Indeed, a modern type of attack based on more elaborate processes like game theory, and ML algorithms called smart attack has emerged. Social Media Automated Phishing and Reconnaissance (SnapR) is an example of the use of ML algorithms in the creation of attacks. Based on the user's Twitter hobbies, this new attack is able to automatically create a personalized message to phish its victim [15]. A new type of malware created by IBM companies, called *DeepLocker*, is based on a deep learning algorithm and can automatically adapt its strategies using indicators like geolocation [16]. Unthinkable a few years ago, these smart attacks are changing the attack landscape and becoming serious threats. This new type of attack makes it possible to bypass countermeasures by being more reactive and less dependent on the human

actor. Moreover, a smart attack possesses the capacity to converge as quickly as possible to the optimal solution. These new attacks can be seen as improvements to existing attacks. However, these approaches have also made it possible to create attacks previously unachievable due to their high data demand or excessive manual processing time. Consequently, it becomes urgent to study this new type of attack, in order to better understand and counter them.

At the same time, communication protocols are constantly improving and new paradigms are added over time to meet new needs. Although the issue of security is beginning to be taken into consideration more and more, the security of these new functionalities is most of the time not evaluated during their integration. Therefore, the objective of this thesis is to study attacks in IoT networks in order to highlight the potential flaws. In particular, we focus on the impact of new smart attacks on these networks to identify new vulnerabilities. Based on ML approaches, we focus on **creating new smart attacks** that can automatically infer the victim's strategy and adapt to it. The main objective is to create the most global framework possible in order to generate different types of attacks on different types of protocols. Therefore, an attacker is able with one and the same logic to attack on several types of protocol while increasing the effectiveness of existing attacks.

1.5 Thesis Organization

This thesis compiles the work we have done on the subject of network security and more particularly the creation of smart attacks over the past three years. It is divided into seven distinct chapters, the following of which are organized as follows:

In Chapter 2, we present the general context of this thesis and the state-of-art creation of smart attacks in wireless networks. More specifically, we focus on attacks created using machine learning.

Chapter 3 focuses on the study of two vulnerabilities present in wireless communication protocols. From these vulnerabilities, two frameworks allowing the creation of diverse kinds of attacks on different types of communication protocols, named respectively HARPAGON and FOLPETTI are described. Their effectiveness is investigated in this chapter through theoretical analysis and formal security analysis. Following this theoretical analysis, we propose to evaluate these new frameworks coupled with jamming attacks. To perform this evaluation, we create a new jamming attack module for the ns-3 simulator.

In chapter 4, after a brief state of the art of the different tools already present in the literature to simulate a jamming

attack, we describe our new module and its validation model. After we had proven the behavior of the simulator with results obtained on a real testbed, we evaluate the FOLPETTI framework coupled with a jamming attack.

In chapter 5, an evaluation of these two frameworks is made in the real world. First, we describe the test-bed created to prove the effectiveness of these two frameworks coupled to several types of attacks. Indeed, we evaluated the performance of the HARPAGON framework with a passive and active attack in terms of energy consumption and efficiency. An evaluation of the FOLPETTI framework is also completed in this chapter by comparing it to other attack strategies already carried out in the literature. Finally, we show the energy consumption of the victim employed by these two types of frameworks.

In the last chapter 6, we highlight the use of our two frameworks created during this thesis with real use cases. Indeed, we show that it is possible to decommission a drone flying over an illicit zone thanks to these frameworks. In this case, we use our frameworks as a defense system and we try to make them as targeted as possible. Thus, after describing the behavior of two types of drones, one equipped with a conventional standard communication system and the other more secure, we show how our frameworks can impact their communication.

1.6 List of publications

Journal articles

- Emilie BOUT, Valeria LOSCRI, Antoine GALLAIS. "HARPAGON: An energy management frameworkfor attacks in IoT networks". In IEEE Internet of Things Journal (May 2022).
- Emilie BOUT, Valeria LOSCRI, Antoine GALLAIS. "Evolution of IoT Security: the era of smart attacks". In IEEE Internet of Things Magazine (March 2022).
- Emilie BOUT, Valeria LOSCRI, Antoine GALLAIS. "How Machine Learning changes the nature of cyberattacks on IoT networks: A survey. In Communications Surveys and Tutorials, IEEE Communications Society, Institute of Electrical and Electronics Engineers (October 2021).

International conferences

- Emilie BOUT, Valeria LOSCRI, Antoine GALLAIS. "Energy and Distance evaluation for Jamming Attacks in wireless networks." In IEEE/ACM - The 24th International Symposium on Distributed Simulation and Real Time Applications. Prague, Czech Republic (September 2020)

- Emilie BOUT, Alessandro BRIGHENTE, Mauro CONTI, Valeria LOSCRI. "FOLPETTI: a novel Multi-armed bandit smart Attack for Wireless Network" . In ARES 2022 - 17th International Conference on Availability, Reliability and Security. Vienna, Austria (August 2022)

National conferences

- Emilie BOUT, Valeria LOSCRI. "Un nouveau module pour simuler des attaques de brouillage sur Ns-3". In: Cores 2022 - 7ème Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication, Saclay, France. (May 2022)
- Emilie BOUT, Valeria LOSCRI, Antoine GALLAIS. "Évaluation de l'énergie et de la distance pour les attaques de brouillage dans les réseaux sans fil". In: CORES 2021 – 6ème Rencontres Francophones sur la Conception de Protocoles, l'Évaluation de Performance et l'Expérimentation des Réseaux de Communication,La Rochelle, France. (September 2021)
- Emilie BOUT, Valeria LOSCRI, Antoine GALLAIS. "Energy effective jamming attacker in wireless network." In: Journée thématique du GT SSLR 2021 sur la sécurité des réseaux France. (May 2021)

Chapter

- Emilie BOUT, Valeria LOSCRI, Anna Maria VEGNI, Antoine GALLAIS. "Energy Saving as a Security Threat in LPWANs". In: (July 2022)

Vulgarization

- Emilie BOUT, Valeria LOSCRI. "Security of the Low Power Wide Area Networks (LPWAN)". In Encyclopedia of Cryptography, Security and Privacy. (July 2022).
- Emilie BOUT, Valeria LOSCRI. "Le machine learning, nouvelle porte d'entrée pour les attaquants d'objets connectés". In The Conversation (November 2021)
- Emilie BOUT, Valeria LOSCRI. "Sécurité des objets connectés : attaquer pour mieux se défendre". In The Conversation (June 2021)

In submission

- Emilie BOUT, Valentin BOUT, Alessandro BRIGHENTE, Mauro CONTI, Valeria LOSCRI. "Evaluation of Channel Hopping Strategies against Smart Jamming Attacks". In: IEEE Global Communications Conference. (December 2022)
- Emilie BOUT, Valeria LOSCRI. "An adaptable module for designing jamming attacks in WiFi networks for ns-3" In: MSWIM 22- International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems. (October 2022)

- Emilie BOUT, Valeria Loscri. "ICARO - An effective Countermeasure for drones Passive Attacks. In: Infocom 2023.

1.7 Relationship of Publications with Contributions

In this section, we provide the relationships of publications with contributions and the different chapters. The contributions can be summarized as follow:

- Creation of a survey focused on attacks based on Machine Learning algorithms on IoT networks since 2014. We also provide possible challenges and research perspectives to improve smart attacks. The publication '*How Machine Learning changes the nature of cyberattacks on IoT networks: A survey*' is related to this contribution. This contribution is explained in Chapter 2.
- Creation of a new framework, called HARAPAGON, based on a vulnerability present in the duty-cycle mechanisms to improve the efficiency of several types of attack and limit the energy consumption of an attacker. The publications *'Evolution of IoT Security: the era of smart attack'* and *'HARPAGON: An energy management framework for attacks in IoT network'* are related to this contribution. This contribution is explained in Chapter 3,4,5.
- Creation of a new framework, called FOLPETTI, based on a reinforcement learning algorithm to bypass the method of channel hopping mitigation. This framework allows to perform several type of attack and the related publications are: *'FOLPETTI: a novel Multi-armed bandit smart Attack for Wireless Network'* and *'Evaluation of Channel Hopping Strategies against Smart Jamming Attacks'*. This contribution is explained in Chapter 3,4,5.
- Creation of a new jamming attack module on the ns-3 simulator. The publications: *'Un nouveau module pour simuler des attaques de brouillage sur Ns-3'* and *'An adaptable module for designing jamming attacks in WiFi networks for ns-3'* explain this contribution. The details of this contribution can be found in Chapter 4.
- The study of the use of these two new frameworks in a real case. We explain how our frameworks can be exploited to create the most targeted attack possible in the event of an illicit drone. The publication 'I can block you - An effective Countermeasure for drones Passive Attacks.' corresponds to the chapter 6.

State of art of smart Attacks on IoT networks

2

In this chapter, we present the general context of this thesis. We initially focus on the attack vectors present in IoT networks as well as the diverse types of attacks we can generate from them. Following, we explain the motivations for creating Smart Attacks and the different types of machine learning algorithms that exist to implement such strategies. Finally, we provide a state-of-the-art of attack based on machine learning already created in the literature since 2014.

2.1 Vulnerabilities and Denial-Of-Sleep Attacks in IoT Networks

2.1.1 Vulnerabilities on IoT networks

Due to their specificities, IoT networks are subject to many critical vulnerabilities, reported in numerous works of the literature such as in [17]. We categorize these attack vectors into five categories as follows:

- **Deficient physical security:** In many cases, IoT objects are deployed in unsupervised areas. This accessibility can lead to the alteration of certain components such as the electrical circuit or even the complete replacement of the device. Under these conditions, adversaries can therefore easily take control of a node or modify the network topology by adding or removing devices. Additionally, cryptography primitives can be easily extracted, and the attacker can gain unrestricted access to information stored in the memory chip.
- **Energy and computation constraints:** Due to their use cases, most IoT devices are battery operated and have limited computational resources. Therefore, these restrictions are the source of many possible attack strategies in IoT networks. Indeed, by targeting the batteries, an attacker can attempt to deplete them in order to place the device out of service. Furthermore, overloading computing resources can also play a role in energy consumption.
- **Inadequate authentication and encryption:** IoT devices produce and convey a large amount of private data that must be protected to guarantee the four main fundamental security concepts which are: confidentiality, authenticity, integrity and, non-repudiation. These concepts ensure the information can only be read and modified by authorized persons. To resolve this kind of problem in generic wireless networks, encryption and

authentication algorithms are employed. However, due to the constraints of IoT networks, the implementation of classic algorithms is more complex because they are energy-intensive and resources-constrained. This is why, lightweight encryption algorithms have been developed to establish an equilibrium between security, performance, and resources. Although there is active research on this subject, many security challenges remain to be answered, as mentioned in [18]. To cite just one example, finding a balance between key size and the level of security it provides remains a problem. Indeed, a small key requires fewer resources during encryption, but it becomes more elementary for an attacker to estimate it. Finally, with the algorithms put in place especially for the IoT, new attacks can appear from these new logics as shown by [19] .

- **Unsecured access control:** Most of the time, IoT devices as well as their cloud components do not require the user to employ a complex and secure password. Typically, these objects have insecure password recovery mechanisms and poorly protected credentials. Moreover, the manufacturers do not oblige the user of the IoT device to reconfigure the initial password after its installation or change it regularly. All of these points allow an attacker to gain access to a device.
- **Inadequate update management:** To regularly correct security breaches, updates are necessary. However, the possibility of making updates is a problem that the majority of manufacturers do not take into account today for mainly budgetary reasons. In addition, when the update mechanism is available on IoT networks, the latter can in addition be the source of vulnerabilities. Indeed, an attacker can insert malicious code such as a virus in the update, and the latter will then be propagated at the same time. Therefore, the update process must also be protected and, in the context of the IoT, it must be as energy-efficient as possible. This last point is also a critical aspect of research as can be seen in the literature with [20].

All of these potential vulnerabilities have led to various attacks on IoT networks. Several authors have proposed a classification of possible attacks according to the vulnerability of each layer composing the architecture of an IoT system in [21] and [22] .

2.1.2 Denial-of-Sleep Attacks on IoT Networks

In this section, we only focus on attacks based on power constraint vulnerabilities that target the battery of an IoT device. Before describing the different categories of sleep denial attacks present in IoT networks, we briefly summarize their principle.

Table 2.1: Energy consumption for each states for different protocols. [24]

Power Consumption	LoRa	ZigBee	SigFox	WiFi
Transmit (range from 7 to 20 dBm)	$[18, 125]mA$	$[85, 500]mA$	$[22, 54]mA$	$[40, 67]mA$
Receive	$11mA$	$65mA$	N/A	$34mA$
Sleep	$1\mu A$	$55\mu A$	$1.5\mu A$	$1\mu A$

Principle of Denial-of-Sleep Attacks:

Denial-of-Sleep attack is a sub-class of DoS attacks determined by Wood and Stankovic as: "any event that diminishes or eliminates a network's capacity to perform its expected function" [23]. Its principal purpose is to completely drain a victim's energy by forcing them to perform an unexpected/illegal operation. To cause additional energy consumption, the attacker has two main alternatives. The first is to cause an energy overhead at the software or firmware level by forcing the victim to perform additional actions. The second is to play on the different paradigms present at the network level. Indeed, in [24], the authors report the energy consumption of diverse states of an IoT device present in a communication protocol and conclude that for different types of protocols, the emission action and receive uses the most power. To provide an example, we report several values of each state present in IoT protocols in Table 2.1. Therefore, if an attacker forces his victim to retransmit data or receive more data than expected, his power consumption will increase dramatically.

Classification of Denial-of-Sleep Attacks on Iot Networks:

We have categorized the sleep denial attacks presented in the literature, according to the targeted Open Systems Interconnection OSI stack layer:

- **Physical Attacks:** These types of attacks focus on the hardware components of the IoT system and the physical layer of the communication medium. In this category, two subtypes of classes can be expressed. One includes all attacks directly generated by human activities such as damaging or tampering with a node. An example of this type of attack can be the assault provoked by a sniper in April 2013 on a California Power Station that knocked out a power grid station [25]. The second subtype counts all attacks targeting the physical layer of the IoT network, such as jamming attacks. *Jamming attacks* aim to deliberately interfere with the transmission signal in order to disrupt it and render the information it conveys indistinct. This type of attack occupies the transmission channel. As a result, the victim postpones its emission and remains active during this time. In [26], the authors show that a jamming attack on a 6LoWPAN

protocol increases the victim's power consumption by 20%.

- **Network Attacks:** All attacks in this category focus on the network layer of the Open Systems Interconnection (OSI) model. This type of attack directly targets the operation of communication protocols and may correspond to one of two types: passive or active. Passive attacks relate to any attack methodology that does not involve any interaction by the attacker on the target system. For example, an *Eavesdropping Attack* which consists of listening to and recording as many packets as possible in order to subsequently perform data analysis is one of them. By their nature, this type of attack is complex to detect because the attacker does not influence the outcome of system operations. However, this type of attack does not directly lead to a denial of sleep but can be useful to collect the maximum information in order to create a more advanced attack targeting the battery of a device. On the contrary, during an active attack, the attacker is physically invested in the attack itself and aims to modify the operating result of the system by modifying, deleting, creating data, or causing a denial of service. In this sub-category, we can cite all the attacks that exploit the vulnerabilities of the routing protocol such as *Selective Forwarding Attacks*, whose goal is to change the routing of data.*Black-Hole Attack* is one of these types of attack and consists of forcing all messages to transit to a malicious node to be then dropped. In the same logic, the *Grey-Hole Attacks* have been created to minimize the probability to be detected. With this type of attack, the attacker instead of deleting all messages only deletes some of them. This selection can be done in a probabilistic or rational form by removing only certain types of packets. This type of attack can have a significant effect on the number of retransmissions and therefore on the energy consumption of the victim. Indeed, in [27], the authors show that for a rejection rate of 50% of packets, the nodes of the network lose 40% of energy due to the increase in the number of retransmissions that it generates.

  Instead of dropping packets, attackers at the network layer can also gamble on modifying or creating packets. Replay and Flooding attacks are based on this strategy. In a *Replay Attack*, the attacker uses legitimate captured data, which is then sent back to the original destination. *Flooding Attack* are more or less the same, but instead of re-sending legitimate packets, the attacker sends a high number of packet requests. Any node receiving a packet must process it to at least know if it was intended for it. This processing is costly in terms of energy and must be avoided as far as possible, which is not the case during replay or flooding situations.

- **Application Attacks:** This class includes all application-layer attacks that exploit known vulnerabilities in the software code of an IoT device. Trojan horse programs, worms, viruses, spyware, and malicious scripts are attacks that can be employed in this category. This type of attack can allow an attacker to access protected data, such as in a Structured Query Language(SQL) Injection attack. Otherwise, it can also evade security restrictions of IoT devices through stealth programs hiding malicious features or with seemingly harmless middleware that receives orders from a third party. For example, if a thermostat is configured to communicate data only when the temperature exceeds a certain threshold, it is possible to play on its environment so that the threshold is constantly reached. With this type of attack, the victim performs additional processing and hence reduces its lifetime.

Table 2.2: The different classes of denial-of-sleep attacks on IoT networks.

Physical Attacks	**Network Attacks**	**Application Attacks**
Vulnerabilities exploited		
Centered on the hardware components of the IoT system and the communication medium	Based of the vulnerabilities of communications protocols	Modify or exploit known vulnerabilities of the software code of the IoT device
Examples of attacks		
► Jamming Attacks ► Tampering Attacks ► Physical Damage	► Replay Attacks ► Routing Attacks ► Traffic Analysis Damage ► Spoofing Attacks	► Virus and Worms ► Spyware Attacks ► Trojan Horse

A summary of these different attacks is provided in Table 2.2. However, from the attacker's point of view, all of these attacks still include many weaknesses. Indeed, most of them rely on data analysis to discover the optimal strategy to limit the probability of being detected. This analysis is based on collecting and looking for a correlation between data most of the time, a tedious and time-consuming task for an attacker. Moreover, in the event of a change of environment, the attacker must start a new interpretation of the data. Therefore, fully automating attacks in some cases with ML algorithms can have significant benefits for an attacker. We explain in detail the different motivations for using machine learning in attacks in the following section.

2.2 Smart Denial-of-Sleep Attacks on IoT Networks

Before explaining in detail the principal motivation to use smart attacks, we briefly describe the different ML algorithm approaches that can be employed for this specific purpose.

2.2.1 Background of Machine Learning Algorithms

The fundamental goal of machine learning is "to teach computers to learn" and subsequently, to act and react by improving the way they learn and their knowledge on their own. It is a computer programming technique that uses algorithms and statistical models to give devices the ability to learn on their own without explicit programming. In this field, three common categories of machine learning approaches are present: supervised learning, unsupervised learning, and reinforcement learning. All the advantages and disadvantages of the various algorithms are summarized in the Table 2.3. Below we explain the difference between the three classes:

Supervised Learning:

Supervised machine learning methods are designed to learn by example and can be compared to learning in the presence of a teacher. Therefore, the algorithm needs training data consisting of inputs associated with the correct output. During the training phase, the algorithm looks for patterns in the data that correlate with the desired category. Then, based on this learning phase, the algorithm will be able to classify each new input and deduce its output. More commonly, supervised algorithms can be expressed in this form:

$$Y = f(X).$$

where Y represents the output variable, X the new entry (input) and f(x) the supervised algorithm. As this field is constantly evolving, several types of supervised machine learning have developed as below:

- **K-nearest neighbors algorithm (KNN):** This method, also called nearest neighbors, ranks a new entry based on the placement of the neighbors. Indeed, to classify a new input, the system finds the nearest K neighbors among the training data set and retains the most represented category among these K neighbors. Several methods, detailed in [28], for calculating the shortest path are used, such as Euclidean distance, Manhattan distance, or Hamming distance. Fig 2.1 illustrates how this algorithm works to classify a new entry.

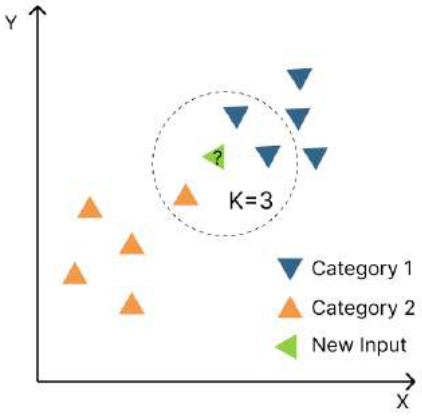


Figure 2.1: Example of K-nearest neighbors Algorithm

- **Decision Tree Learning (DT):** A decision tree breaks down a set of data into subsets. Algorithms like IDS3 and C4.5 allow to create a decision tree based on learning data where each internal node describes a test on a learning variable, each branch represents a result of the test, and each leaf contains the value of the target output.
- **Random Forest (RF):** As the name suggests, this method groups multiple decision trees and each tree is trained with a different set of data. The final output is the average of the outputs for each tree, as show in Fig 2.2.
- **Support Vector Machine (SVM):** The idea of this algorithm is to find a hyperplane that best divides the data of the training set into n classes. Consequently, the objective is to find a plane that has the highest margin, i.e the maximum distance between data points of both classes.
- **Linear Regression (LR):** This algorithm predicts the value of one variable (y) based on the value of another variable (x). The variable to be predicted is called the dependent variable. The goal is to find a linear relationship between the input and the dependent variable. Fig 2.3 is an example of this algorithm.

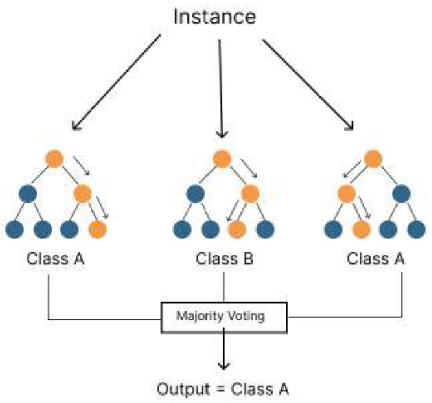


Figure 2.2: Example of Random Forest Algorithm

Unsupervised Learning

Unlike supervised algorithms, this class of machine learning does not require labeled data to learn. These are the algorithms that will find by themselves a correlation between the data. The goal is to find the most common features in order to form clusters or create rules to discover relationships between variables. Like the supervised category, several unsupervised algorithms exist in the literature, however, we only explain below the two most common algorithms used to create smart :

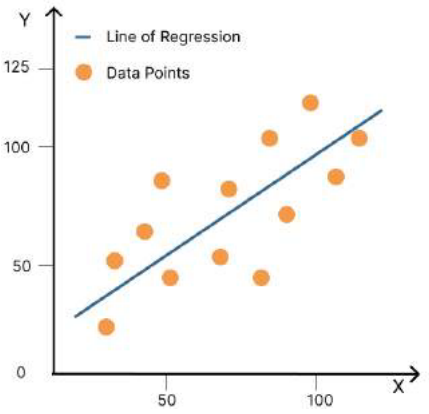


Figure 2.3: Example of Linear Regression Algorithm

- **K-means**: This algorithm recognizes a pattern in the data and groups the observations of the dataset into K distinct clusters. Thus, similar data get grouped together in the same cluster. To compare the degree of similarity between various observations, k-means employs the concept of dissimilarity distance.
- **Generative Adversarial Network (GAN):** This unsupervised algorithm aims to generate new synthetic data instances that can imitate data. This algorithm is composed of two neural networks: the generator and the discriminator, as shown in Figure 2.4. The generator requires random data as input and produces new instances. The discriminator receives data from the generator and tries to detect if the samples are real or generated. Thus, backpropagation is used to improve network accuracy. The generator receives feedback from

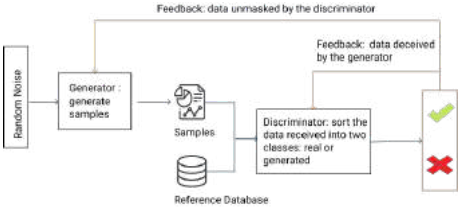


Figure 2.4: Example of Generative Adversarial Network Algorithm

the discriminator which signals whether the data is valid (generated or not).

Reinforcement Learning

Reinforcement learning is an online learning method, which does not require any training data. This method aims to reproduce human behavior, that is to say learning through experience. Indeed, reinforcement learning algorithms choose their future action taking into account their current environment. This method can be compared to the behavior of a child who is learning to walk and who has to choose between moving the left leg or the right leg. If the child falls, he deduces that his choice was wrong and will then understand that following the left leg he must move the right leg forward. In the same way, algorithms by reinforcement after each action receive a negative or positive reward in order to update their strategy. In mathematics, this process can be formalized in the form **Multi-Armed Bandit (MAB)** modeled by a Markov Decision Process(MDP) which can be described thus in the form of 5-tuples $\langle S, A, P, R, \gamma \rangle$, where:

- S is a finite set of states s;
- A is a finite set of actions a;
- $P_a(s^n, s^{n+1})$ is the probability that an action a in state s^n in time $n+1$;
- $R_a(s^n, s^{n+1})$ is the expected immediate reward received after transitioning state s^n to state s^{n+1} due to action a;
- $\gamma \in [0, 1]$ is a discount factor.

The main objective of a MDP is to find a policy π that associates an action with each state $\pi : S \rightarrow A$ to maximize the reward. Therefore, in reinforcement learning, the main problem is to infer the optimal balance between exploration and exploitation. Several policy algorithms to resolve this dilemma are presented in the literature [29]. E-greedy, Upper Confidence Bounds and Thompson Sampling algorithms can be example. These policy algorithms can be in some cases improved with the **Q-learning** method. This algorithm based on a Q-table makes it possible to record for each chosen action its maximum reward.

Model Assessment and Selection

The performance of a machine learning model can be assessed thanks to various tools implemented by the community a few years ago. Many effective methods to estimate the reliability of a machine learning algorithm and their operating parameters exist. In this part, we will only discuss the metrics that have been used during these three years to evaluate the effectiveness of new intelligent attacks.

Table 2.3: Machine Learning algorithms exploited for generating smart attacks.

Algorithm	Paradigms	Advantages	Disadvantages	Application in Attack
K-NN	Supervised	▶ Intuitive and Simple. ▶ Versatile: different distance criteria. ▶ Classification and Regression use.	▶ Curse of Dimensionality. ▶ Slow algorithm. ▶ Need homogeneous features.	Deduce IoT information like activities or sensitive data
DT		▶ Requires little data preprocessing. ▶ Work with numerical and categorical features.	▶ Instability. ▶ Complexity.	
RF		▶ Solve the high variance estimator problem of Decision tree. ▶ Classification and Regression use.	▶ High computational costs. ▶ Predictions are slower.	
SVM		▶ Effective in high dimensional spaces. ▶ Versatile: different Kernel functions. ▶ High Accuracy. ▶ Works well on smaller cleaner datasets	▶ Training time with SVMs can be high. ▶ Less effective on noisier datasets. ▶ Isn't suited to larger datasets.	
LR		▶ Easier to implement, interpret and very efficient to train. ▶ Small number of hyperparameters.	▶ Based on Assumption of linearity. ▶ Very sensitive to anomalies in the dataset.	Generate false data injection
K-means	Unsupervised	▶ Easy to understand and implement. ▶ Applicable to large data.	▶ Find the optimal parameter k. ▶ Dependent on initial values. ▶ Less effective on noisier datasets.	Deduce the frame activities in the network
GAN		▶ Produces very realistic data. ▶ No Markov chain Monte Carlo needed.	▶ Unstable to train. ▶ Huge computation.	Generate false data to deceive a IoT network or machine learning algorithm.
MAB	Reinforcement	▶ Online algorithm. ▶ No need prior information.	▶ Converging on the right solution can be slow. ▶ The learning process must start again when the environmental changes.	Carry out an attack without prior information on the victim
Temporal-Difference Learning		▶ Not require a model of the environment. ▶ Online algorithm.	▶ Less stable than Q-learning. ▶ May converge to the wrong solution.	
Q-Learning		▶ Not need to know the transition probability matrix.	▶ Behave poorly in some stochastic environments.	

The standard and basic metric employed in the academic literature is **accuracy**. Its mathematical definition has the following:

$$accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where:

- TP is a True Positive ; model correctly predicted the positive class.
- FP is a False Positive; model incorrectly predicted the negative class.

- FN is a False Negative; model incorrectly predicts the negative class
- TN is a True Positive; model correctly predicts the negative class

In some specific cases, accuracy may be too weak as indicator to prove the performance of an algorithm. This can be an example when working on a dataset with an over-represented class. This is why the metric **precision** corresponding to the ratio of correctly predicted positive observations to the number of total observations can be used:

$$precision = \frac{TP}{TP + FP}$$

2.2.2 Motivation of the creation of Smart Attacks

In 2018, Brundage et al. alert that a new dimension of attack is about to emerge through machine learning [30]. Although this innovative technology has allowed advances and automation of a lot of tasks in many areas such as security, this progress can be seen as a double-edged sword. The use of ML algorithms is becoming widespread thanks to the advance of free frameworks such as Tensorflow or OPenAi which allow them to be implemented easily [31, 32]. Consequently, an attacker does not need much Machine Learning knowledge and considerable time to implement this new type of attack. Moreover, machines have become increasingly powerful and accessible on the market. Indeed, it is currently possible to directly implement machine learning algorithms in hardware circuits to optimize throughput and adapt processor tasks. For example, Intel® has developed its Intel® FPGA Deep Learning Acceleration (DLA) suite which includes many key components to optimize machine learning algorithms in a Field-programmable gate array (FPGA) [33]. Therefore, all of these advancements make it easier for an attacker to design attacks based on machine learning algorithms. As a result, the threat landscape will gradually evolve along three different axes in the coming years:

- **The expansion of existing attacks:** The goal is to improve certain parameters of existing attacks, such as the speed of execution, reactivity, or automation of tasks that require human intervention. For example, the creation of green attacks, that is to say attacks that consume little energy but remain effective, remains a challenge in the literature. Most of the time, solving this problem relies on a trade-off between efficiency and energy that can be calculated by a ML algorithm. We can for example think of an attack which, according to its environment, makes the decision to fall asleep autonomously to save energy if it knows that this will be ineffective.

- **The introduction of new threats:** Additionally, creating an attack using ML can exploit existing but previously unexploitable vulnerabilities. Indeed, several attacks relying on an accurate study of hundreds of data require a lot of time for an attacker like the traffic analysis attack. Therefore, although these vulnerabilities are already present, few attacks have actually been implemented in the real world. In parallel, with supervised learning methods, it becomes easy to classify data and derive insights from it. In this manner, the analysis can be automated, the attacker saves time and, the probability of human error decreases. In addition, new vulnerabilities have also appeared in recent years and are exploitable due to machine learning. Adversarial machine learning attacks are examples of this and aim to deceive or hijack a machine learning model using data. This type of attack is expanding due to the constant use of machine learning in different areas. One of the techniques to deceive a machine learning algorithm is to contaminate the training dataset. This contamination can be produced by generating fake data similar to real ones with the help of unsupervised machine learning algorithms .
- **Change of the typical character of threat:** Finally, the use of ML algorithms can change the typical character of threats. Indeed, specific attacks could become widespread in the near future and target several types of threats at the same time. With the help of a machine learning algorithm, it becomes quite possible to create a framework allowing to deduce several vulnerabilities of a network and to launch different types of attacks according to this analysis. Thus, we will no longer have a methodology for one type of attack but an identical algorithm that can perform various types of attacks on diverse types of protocol.

Additionally, ML algorithms might respond better to the attributes of an *optimal attack*. After having considered several types of attacks on the IoT networks, we have been capable to deduce an optimal attack had the following common characteristics: being undetectable, proactive, frugal, adaptive, autonomous, robust, and requiring little knowledge. Indeed, an attack to improve its duration of action must be the least detectable possible to prevent its victim from reacting and adjusting its strategy. In the same idea, to extend its duration of action as much as possible, an attack must be frugal in energy. In an IoT context, it is conceivable that the attacker is also a network node with limited resources. In addition, in most cases, the attacker does not have a power source nearby when executing their attack. An optimal attack must also be proactive and adaptive. Most of the time the parameters of the victim's environment are unfixed over time and vary according to several factors. For example, the victim may be transmitting on different frequencies or be geographically mobile. An optimal attack must autonomously anticipate

these changes to remain effective and not increase the probabilities to be detected. A perfect attack should require very little basic knowledge in order to limit the time of eavesdropping, on a network. To avoid being identified the attacker must rapidly learn the essential features and components of the attack environment.

2.2.3 Classification of smart attacks

For all of reasons mentioned above, the integration of ML in the creation of attacks has become a new and open subject in the literature. Consequently, in order to better answer and explain what ML brings to the creation of an attack we have established a taxonomy composed of three main categories, as we can see in Fig 2.5. We classify these different motivations according to the type of ML algorithm. Based on this categorization, we can notice that one type of ML algorithm is used for a distinct purpose. In this taxonomy, we have also added the notion of targeted security principles which are confidentiality, integrity and availability. Confidentially term groups all the attacks allowing access to personal information or a system without prior authorization. Integrity, as the name suggests, refers to the reliability of the data i.e. no unauthorized person can modify the content of the data. Availability refers to the fact that the data or the system is available at all times. Finally, we also added the notion of passive or active attacks in this taxonomy. Passive attacks do not disrupt the system directly, the attacker does not aim to alter the behavior of the network, the primary objective is to analyze the data. This type of attack is more of a danger to data privacy. On contrary active attacks aims to intercept the connection and effort to modify the behavior of the network. Involving an active attack is generally difficult and requires more effort than passive attacks.

Supervised algorithms and more particularly classifications algorithms are frequently used for the **Data analysis**. This type of algorithms can be of great help for the analysis of data and behavior of the victim. The study could be automated with this method and save many resources like time. In addition, implementing an algorithm to infer information can reduce human mistakes during analysis such as inattention errors and therefore decrease the probability of being detected. Within this category, two sub-goals of using supervised learning for data analysis can be cited: the deduction of cryptographic information such as the password and the selection of the optimal target. Indeed, analyzing the data circulating on an IoT network or even certain details depending on an IoT device such as the energy can lead an attacker to guess the cryptography keys. This is the case for side channel attack, where an attacker aims at extracting cryptographic information from an IoT device, through measurement and analysis of physical parameters. Finally, by analyzing the

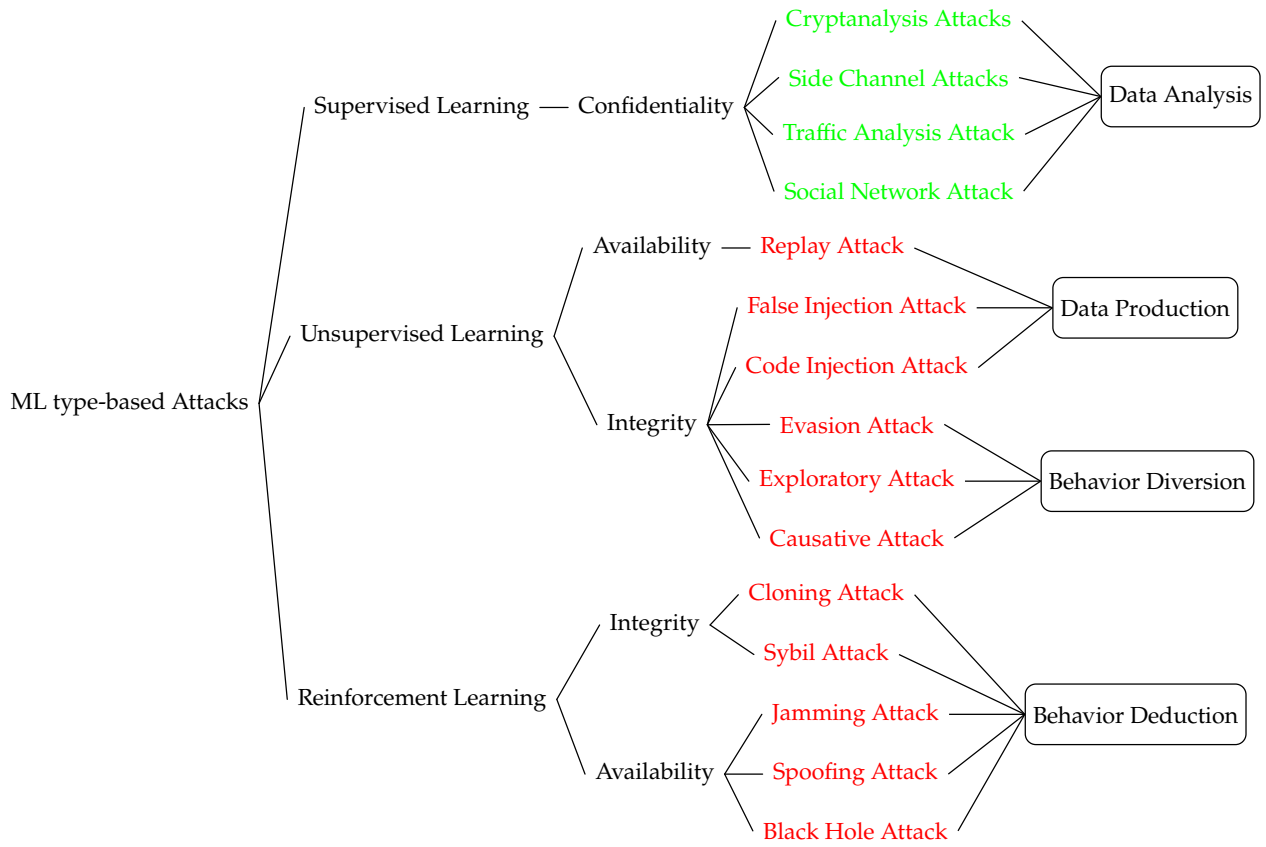


Figure 2.5: The different exploitation of ML into attack generation, Colors indicate the implication of the attacker : Passive attacks, Active attacks

traffic of the IoT networks, it is possible to deduce several pieces of information such as the frequency of use of an IoT object, its type and even the habits of users. This type of passive attack is named traffic analysis attack.

Unsupervised algorithms can be used to create attacks based on **data creation** such as replay or false injection. In this category, several sub-goals are present like the creation of content or the alteration of data. These two categories are very similar but it differs in the fact that for the first, the attacker creates new information in order to deceive the IoT network while in the second the attacker modifies existing data. Creating data can be very risky for an attacker because the data must be comparable to existing data so as not to alert defense systems. Some unsupervised machine learning algorithms like GANs enable the accurate generation of data and therefore reduce the probability of being detected. For example, in the case of an advanced replay attack, the attacker must intercept the data, modify it, and re-inject it into the network in order to disrupt the behavior of the victim [34]. Nevertheless, to reduce the probability to be detected, the modification must be minimal enough not to alert enemies but must be able to cause consequences. Unsupervised learning can be additionally used for the victim **behavioral diversion**. This purpose mainly covers adversarial-type attacks. The utilization of

machine learning algorithms is increasingly used in IoT networks, whether to manage resource allocation or security. However, machine learning algorithms represent the new attack vectors. Indeed, by generating false information with unsupervised learning algorithms it is possible to corrupt system training data to produce alternative behavior.

Finally, the last motivation to employ machine learning algorithms in the creation of an attack is to **detect the behavior of the victim**. With reinforcement learning, an attacker can automatically deduce the strategy of its victim and select the most opportune moment of attack in order to be as undetectable as possible. The main advantage of this method is that it requires little prior knowledge because the attacker gains knowledge as they attack. Also, the use of reinforcement learning in creating an attack is more interesting when the dynamic environment quickly changes its configuration. Indeed, the study of the behavior being automated and faster, the attacker does not waste time developing a strategy as soon as an environment parameter changes. In this category, we can find several types of active attacks such as jamming, spoofing or, grey hole attacks. Hence, in these cases, to be as undetectable and effective as possible, the attacker must have the same network parameters as the victim which can be inferred with reinforcement learning algorithms.

2.3 State-of-arts of Smart Jamming attacks

The main topic of this thesis was developed in collaboration with the French General Department of Armament and focuses on the creation of smart attacks. However, aware that the duration of a thesis is limited, we decided to focus on the study of attacks leading to denial-of-service and more particularly on jamming attacks in the context of IoT. We selected this type of attack because it has the particularity of being easy to set up for an attacker but difficult to detect. Indeed, an aggressor does not need to take control of a network node to carry out this type of assaults, unlike the routing attacks which is based on the prerequisite that the attacker is already inside the network. In the same logic, identifying and locating a jamming node is difficult because we do not have prior knowledge of its position. In the following section, we define the concept of jamming attack, its objectives and consequences. Then, we explain the type of learning approach that has been developed in the literature for generating smart jamming attacks.

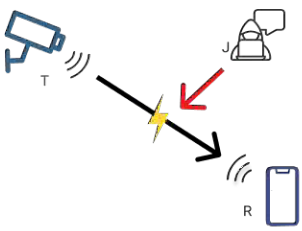


Figure 2.6: Process of jamming attack.

2.3.1 Jamming Attacks: an overview

The inherent openness of wireless communication techniques has made them vulnerable to jamming attacks. This kind of

attack consists of intentionally interfering with the communication medium to keep it occupied or to corrupt a signal transmission. As we can see in Figure 2.6, the attacker (J) transmits a signal to disrupt communication between the transmitter (T) and the receiver (R). The goal is to prevent the exchange between the legitimate nodes of the network (T and R) by voluntarily occupying the channel or by causing a collision in order to force T to re-emit. This type of attack is mostly conducted at the physical or MAC layer and the consequences can take place on a single device or on the entire network.

Four categories to classify jamming attack is present in the literature. This classification is made according to the level of strategy employed by the attacker and can be summarized as follows:

- **Constant Jamming Attacks:** The main objective is to continuously emit a signal regardless of the communication strategy of the victim. In this way, the attacker occupies the transmission channel and forces the victim to postpone his transmission. Most of the time, the attacker sends a radio signal composed of random bit without following a MAC protocol. the Easy to implement, this type of attack has the main disadvantages of being easily detectable because the attacker is active for a long time. Moreover, from the attacker's point of view, the constant jamming is not interesting because the latter consumes a lot of energy. As we have just seen above, the transmitter state is the node's most energy-consuming mode.
- **Random Jamming Attacks:** To solve the energy problem, the random strategy was developed. The concept is to transmit a signal of fixed duration at a random time interval. Therefore, the attacker consumes less power as it alternates transmission and sleep modes. Sleep mode is considered to be the least energy-consuming operating mode of a node.
- **Reactive Jamming Attacks:** Although the random strategy saved energy for the attacker, it remains highly detectable and inefficient. Indeed, with a random approach, the attacker can transmit as soon as a legitimate communication takes place and unnecessarily obscure the channel. With a reactive attack, the attacker jams only when activity on a channel has been detected, reducing their attack time and increasing their effectiveness. The main goal is not to occupy a channel but to cause the maximum of collision in order to force the node to retransmit.

In addition to existing countermeasures, methods of detection in face of jamming attacks have been developed. The first is based on statistical approaches. Several statistics measurements that may be employed to detect jamming attacks and a case study for each are presented in [35]. The first metric mentioned is a natural measurement, the signal strength.

Indeed a jamming attack affects the signal strength of a device. However, in practice, this method is binding because a node does not easily provide this metric. This second is the packet delivery ratio (PDR). As the signal strength, a jammer impacts the average of this metric. Indeed by corrupting a packet, the acknowledgement used to calculate the PDR will never be received. Thus when an attack takes place, the total average of the PDR drops. The last measure is the Carrier Sensing Time; it is the amount of time spent by a node to wait for a channel to become idle. An attacker can prevent a legitimate node from emitting by permanently occupying the channel. Consequently if a channel is busy for a long time, the total average of carrier sensing time increases. Nevertheless, the use of a statistical method to detect a jamming attack has some drawbacks like creating many false positives. That is why several proactive detection and countermeasures have been developed, such as 'JAM'(A Jammed-Area Mapping Service for Sensor Networks) or 'JAID'(An Algorithm for Data Fusion and Jamming Avoidance on WSNs) [36]. Moreover, to address the problem of adaptability to different environments (e.g. protocol, topology, number of nodes), several machine learning-based detection methods have emerged in recent years. They utilize different algorithms like deep neural networks or reinforcement learning [37, 38].

Several countermeasures to combat jamming attacks have been considered in last decade as described in [39]. Frequency Hopping Spread Spectrum (FHSS) is one of this method and consists of alternately using several transmission channels distributed in a frequency band [40]. This countermeasure is only effective if the attacker cannot infer the frequency hopping pattern. Another method to counter jamming attacks is the Direct-Sequence Spread Sequence (DSSS). This method reduces the interference and increases the resistance to jamming by adding pseudo-random signal to the signals. The messages are reconstructed by the receiver by filtering the noise to obtain the original data. The Hybrid FHSS/DSSS associates the advantages of FHSS and DSSS methods [41]. This system avoids interference by alerting several channels and by spreading its bandwidth.This method is easy to implement and remarkably increases resistance to jamming. Finally, the Ultra Wide Band Technology (UWB) is a radio modulation technique based on the transmission of very short pulses in a wide frequency band against jamming attacks [42]. These short pulses, therefore, considerably reduce the effectiveness of jamming attacks since it becomes more difficult for the attacker to target the signal.

These countermeasures and detection methods countered some basic jamming techniques like constant attacks. However, we will see in the following section that the integration of machine learning algorithms within these attacks has made it possible to thwart these approaches.

2.3.2 State of art of jamming attacks

From an attacker's perspective, traditional jamming attacks are becoming increasingly inefficient and energy-intensive. For these different reasons, reactive attacks are increasingly preferred. Nevertheless, choosing the optimal moment to jam a network without knowing the protocol and consuming little energy is a very complex problem. Machine learning can help solve this problem by first discovering the network protocol and then determining the optimal attack strategy. Jamming attacks based on this approach is an open and constantly evolving research topic, as revealed in the Table 2.4. During our research on this subject, we have identified three different levels of knowledge that an attacker should have. Consequently, we have defined three sub-categories which are: i) The attacker knows the protocol used in the network; ii) The jammer is oblivious to the configuration of the network, and iii) The attack takes place in a dynamic environment.

The attacker is aware of the protocol:

The attackers are aware of the protocol and topology implemented in the network. The objective of the following works was to prove that the use of machine learning could increase the performance of a jamming attack. Indeed the works developed in [43] demonstrate it is possible by using a Deep neural network algorithm and by knowing the MAC protocol implemented in the network to predict the length of the frame transmitted and to jam the channel only during this period. Therefore, by using machine learning, the attack time can be reduced as well as the battery consumption of the attacker. In another investigation, the attacker is aware that the network employs an 802.11 protocol with the Request to Send-Clear to Send (RTS-CTS) handshake mechanism [44]. According to this knowledge, they develop a Markov decision protocol (MDP) model based on the messages exchanged between a transmitter and a receiver. Thanks to this MDP state transition structure, the aggressor tries to establish the best strategy to limit energy consumption by using a delayed reinforcement learning algorithm. The goal is to determine which exchange pattern (RTS / CTS or DATA / Acknowledged ACK) is the most efficient to jam in terms of energy consumption and the probability of success. After simulating their result, the authors deduced that the most effective strategy was to jam the RTS-CTS frame. They compare this approach with a classical (constant, random) scrambling method and obtain better results with the delayed reinforcement learning algorithm.

The jammer is unconscious of the protocol used:

The initial researches were done taking into account that the attackers possess a large amount of preliminary information about the strategy used by the transmitter and the receiver, e.g. the protocol used, and the topology of the network. However, in a real context, obtaining such kind of knowledge is very complicated and research based on jammer attacks without any knowledge has developed accordingly.

In [45], an online reinforcement learning algorithm was introduced: 'Jamming Bandit', which allows finding an optimal attack strategy while having a reasonable computational complexity. For this, the authors invent a new multi-bandit problem-solving approach. The goal is to find the ideal modulation scheme, and duration of the pulsation to jam the transmission between the transmitter and the receiver. They simulated and compared their solution with the most common method of solving a multi-armed bandit problem: the ϵ-greedy algorithm and prove that their alternative converges faster to find the optimal solution. After testing their algorithm on a single and several victims, they demonstrate the superiority of their algorithm in terms of convergence in both scenarios. However, to establish the multi-armed bandit's reward, the authors need the ACK/NACK frames. However, the latter are impossible to obtain in a specific context like in the military domain or when the User Datagram Protocol (UDP) is employed as explained in [46]. Researches relied on previous article proves that it is possible to base only on standard rewards for all types of environment. They find two new kinds of reward, which are the change of power and the enduring time. Indeed they noticed that during successful jamming, the power increases. Also, experiments have demonstrated that the proposed algorithm converges faster to the optimal solution than that used with e-greedy, except when the discriminating parameter of the e-greedy algorithm is optimal (M). The objective thus becomes finding the optimal discriminant parameter (M) in the first time, in order to apply the e-greedy resolution algorithm in a second step. Thanks to simulation, they compare their results with the 'Jamming bandit'algorithm and show that their method converges faster towards an optimal jamming solution.

Research is carried out even further, taking into account the fact that the behavior of a network to be jammed does not only depend on its current state but also on its previous states in [47]. Indeed, a transmitter-receiver pair can choose the encoding modulation according to the result of the previous transmission. Simulation results show that using a Deep Q-learning paired with recurrent neural networks while taking into account the old and current actions of the network, leads to better results during decision making (decision of jamming or not jamming the network with the right parameters). In order to be effective, a jammer must be as proactive as possible

and have the shortest learning phase so as not to consume a lot of energy resources. Work aims to reduce the learning phase as shown in [46]. To address this problem, they combine the advantages of an orthogonal matching pursuit system (OMP) and a multi-agent system. They conduct the simulation in the MATLAB environment and reduce the learning time in three interactions.

The attack takes place in a dynamic environment:

All the previously mentioned studies are mainly based on a static environment, like a pair of transceivers communicating through a single channel. Yet, in reality, two nodes can possess several channels (named multi-channel hopping mechanism) to communicate in order to overcome interference and thus enhance the overall network performance.

The researches try to improve the previous works by using multi-channel support in [48]. For this purpose, an attacker uses Deep Reinforcement Learning algorithm, where he first observes the victim's behavior and then attacks according to the inferred channel hopping pattern. In response, the victim selects a new chain hopping strategy to reduce the attacker's effectiveness. In this case, the attacker needs to periodically interrupt the attack to learn the new victim's channel hopping pattern. The DRL strategy also relies on the assumption that, during the learning phase, the victim's channel hopping strategy remains static. Moreover, reinforcement learning algorithms may take time to converge on the optimal solution. This is the case for searches [47, 49]. Generally, this learning phase requires a lot of interactions. For example, 2×105 iterations are needed to find its ideal tactic in [49]. In addition, if the environment changes during this time, the jammer must restart its learning procedure. So this learning system(RL) is not effective in face of an adaptive environment. To provide a solution to this problem, Zhuansun Shaoshuaiset et al. set up a new system: Apprenticeship learning in [50]. Thanks to this method, few interactions are useful to converge towards the perfect solution. This method is, therefore, advantageous in terms of efficiency and energy expended.

Authors further reduce the gap by taking into account a cognitive transmitter in [51]. This equipment is able to automatically adapt its parameters according to its environment. Indeed, in this work, the Inter-Technology Communication (CTC) is used to allow to answer the problem of interconnectivity between the IoT devices of different protocols. To achieve direct communication among heterogeneous devices, three methods can be used, such as the change of power level, the change of packet length, and the reordering of the packet. In this work, authors implement a reactive jamming system named JamCloack over a CTC protocol. This attack is composed of both a detection phase and a jamming attack phase. The first step observes and detects CTC activities by

Table 2.4: Jamming attacks based on Machine Learning algorithms in the literature.

Ref	Date	Environment	Test Environment	ML Algorithm	Remarks
[43]	2014	Aware of protocol	Simulation	RL Delayed	Determine the optimal frame to jam in RTS-CTS handshake mechanism
[44]	2018	Aware of protocol	Simulation	DNN	Predict the length of the frame transmitted
[45]	2016	unconscious of the protocol	Simulation	Jamming Bandit	Create their own online reinforcement learning algorithm: "Jamming Bandit".
[47]	2018	unconscious of the protocol	Simulation	Q-learning	Take into account the previous state of the network
[46]	2019	unconscious of the protocol	Simulation	OMP, MAB	Reduce the learning time in three interactions with the environment.
[49]	2019	dynamic environment	Simulation	Apprenticeship	A few interactions are required to converge in the optimal strategy.
[50]	2019	dynamic environment	Real testbed	Q-learning	Experience in a real testbed on a multiple-input and multiple-output (MIMO).
[51]	2018	dynamic environment	Real testbed	K-means	Implement reactive jamming attack over CTC

classifying the traffic, thanks to the K-means algorithm. They demonstrate their new attack in a real testbed and reduce the packet delivery ratio (PDR) by 80.8%.

2.3.3 Conclusion

An important remark that we can make based on the state-of-art, is that reinforcement learning algorithms are often applied in the analysis of the behavior of a network and therefore of the autonomous implementation of an optimal attack strategy. Indeed, having no knowledge of the network and therefore of the data to be processed, it is impossible to use classification algorithms. As we have just seen through the studies of jamming attacks, using such algorithms has many advantages:

- Finding the optimal strategy, i.e. maximizing the impact on the network while minimizing the probability of being detected
- Develop scalable attacks based on the target's environment that can be adaptive. Indeed, we could think that

this type of attack could lead to a framework which would allow, depending on the configuration of the network (protocol used, number of IoT devices, etc.) to choose an optimal attack strategy autonomously.
- Bypass defence systems based on changing network configuration such as hopping channel.

To improve the robutness of jamming attack, we can determine these new lines of research in the near future, such as:

- **Compare the consumption of battery:** One of the objectives of using ML algorithms in jamming attacks is to quickly find the optimal strategy in order to consume less battery for the attacker. Moreover, one of the primary purposes of a jamming attack is to cause excessive consumption of the victim battery to achieve a denial of sleep. However, no comparative study of battery consumption, either from an attacker or a victim point of view, was carried out in a real context during most of the previous experiments.
- **Target multiple victims:** Indeed, all the experiments mentioned above are carried out only on a single pair of receiver and transmitter. However, the strategy of a jamming attack can vary if the attacker targets several nodes or several types of communications protocols.
- **Increase learning speed of algorithms:** The learning speed of the main solutions required many interactions and time to find the optimal attack strategy. During this time, the attacker is detectable and it is a crucial point from an attacker perspective to reduce the learning time.

Consequently, based on these observations, we decided to study during this thesis the vulnerabilities present in several communication protocols in order to create adaptive jamming attacks. The main goal was to create frameworks to generate advanced jamming attacks on several communication protocols. These frameworks have the ability to adapt to their environment such as channel hopping. We also determined to do a study on the energy consumption caused by these attacks on the energy consumption of their victim. Indeed, the main objective of this study is the creation of denial-of-sleep attacks. Finally, after the brief analysis of the different types of sleep denial attacks, we can notice that the effectiveness of an attack depends on the interaction between the victim and the attacker but also in the case of dynamic situations of the parameters environment (like the selected channel). Therefore, although these frameworks were originally developed to create jamming attacks, we show that they can be used to generate other types of attacks. These studies were initially carried out on a simulator. However, we did not content ourselves with simulations, we also validated their results from real experiments.

Analysis of vulnerabilities in wireless networks

3

The emergence of wireless networks, looked at as a considerable technical evolution, has promoted the creation of new methods of exchanging data and accessing resources. This is how Internet of Things networks were born, allowing devices that are often limited in resources and energy to exchange information between themselves and the rest of the world. However, by abandoning the barrier of physical isolation, this technical advancement has brought out the question of security. Indeed, the process of wireless networks essentially consists of the propagation of radio waves between a transmitter and a recipient where devices within range of either party can pick up and analyze the communication. Many types of attacks,previously impossible on wired networks, have emerged in recent decades, such as jamming or passive listening attacks. As these attacks have developed, communication protocols have become more and more robust by adapting first off to their specific use case, but also through the integration of detection and mitigation systems. Thus, in addition to defense mechanisms, power saving or data reduction methods have been integrated within these wireless protocols in order to adapt to IoT devices. In this chapter, we are interested in the resistance of these mechanisms and the possibility that they can lead to new security breaches. Indeed, the addition of certain mechanism can induce an attacker to guess certain essential information when setting up his attack. For example, deauthentication attacks against WiFi protocol was created thanks to the addition of control frames, ensuring the security of the connection and the number of clients associated with an access point as illustrated in Fig 3.1. In parallel, with the different advancements in machine learning technologies, attacks on wireless networks have gained a new level of intelligence. Based on the same logic, we wanted to study the exploitation of flaws present on several wireless communication protocols in order to create an intelligent framework to develop multiple type of attacks.

In this chapter, we describe two security flaws present in many wireless communication protocols, allowing the creation of multiple types of known attacks. First, we demonstrate that it is possible to rely on the duty cycle mechanism of a victim in order to create a framework allowing the creation of attacks with low energy consumption. This framework, called HARPAGON, predicts the optimal period of attack and considerably reduce the probability of being detected. Then, we focus on the study of the vulnerabilities generated by a well known method: a channel hopping. We present another smart framework, FOLPETTI, based on multi-armed bandit

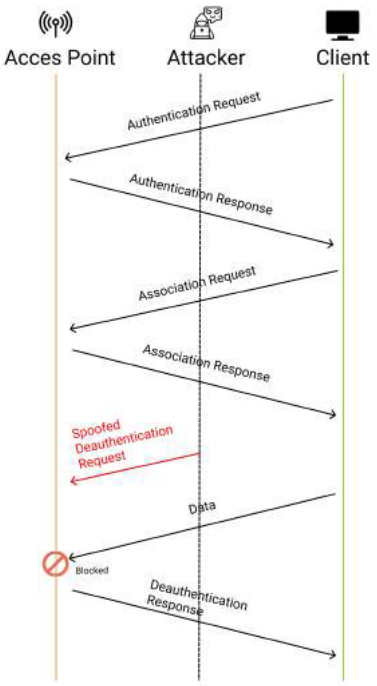


Figure 3.1: Process of Deauthentification Attack

algorithm where we show that an attacker can anticipate the future communication channel of their victim, even if they employ an elaborated channel selection strategy.

3.1 HARPAGON : an attack based on duty-cycle vulnerabilities

Most known active attacks on wireless networks can be performed at any time to have effects. However, by attacking blindly and without following the victim's transmission strategy, an attacker has a high probability of being identified quickly and it is not effective. For example, constant jamming over time leads to the interruption of communications between two legitimate nodes. This behavior is easily detectable as both nodes will suddenly lose connectivity. In the case of a blind replay attack, the packet re-injected by the attacker into the network may never be processed by the victim and thus have no impact. Indeed, it is possible that at the same time the target receiver was also in a listening phase or in a sleep state. Consequently, attacking a network without following the communication strategy used by the victims is inefficient and energy-consuming. Based on these observations, we can determine that the probability of success of an attack depends on its interaction with its victim. Indeed, an attacker must react for a certain time to a certain event for the attack to succeed. For example, in the case of a reactive jamming attack, the attacker must transmit a packet or a signal at the same time as the victim transmits a packet. Similarly, during a replay attack, the attacker must re-inject data between the transmission and reception of a packet between two nodes. In the case of eavesdropping attacks that record communications on networks, the success of the attack requires that the attacker listens to a communication at the same time that the target sends a packet.

The duty cycle mechanism is present in many wireless protocols communication and aims to reduce the energy consumption of a node. With this process, four operating modes have been defined and included in several IoT protocols. Each mode is employed for diverse purposes and provides different energy consumption levels. However, this mechanism can turn into a new security flaw because the victim's behavior becomes predictable. Indeed, on the basis of previous observations, the attacker is able to predict the duration of transmission or listening of its victim and optimise this attack period. Relying on these states to minimize the energy consumption of a process and maximize its efficiency is not new. Indeed, in [52, 53], authors show the effectiveness of the neighbour discovery process based on the alternating/switching states of the wireless nodes in inquiry, scan and sleep state. We derive a similar theoretical framework,

called HARPAGON, based on Markov Chain Theory for modelling an attacking node. The derived analytical framework allows the attacker node to compute the probability of staying in each state in order to achieve the following objectives: a) Maximisation of the attack effectiveness as the probability that attack occurs in the same slot when the victim node is transmitting by minimising the energy expenditure; b) Given a certain limitation cost, the maximisation of the probability that the attack is occurring in a certain time interval.

3.1.1 Duty cycle mechanism: a vulnerability

The use of wireless communication networks may be unsuitable in certain applications due to their limitation on their lifetime. Indeed, it would be inconceivable to think of changing the batteries of a camera to monitor a hostile place every day.It was therefore necessary to maximize the lifespan of these networks while respecting the requirements of the quality of services. In most wireless networks usage, devices send data at specific times such as following an event or at regular time intervals. Moreover one of the most consuming components of wireless network devices is radio communication. Indeed, as reported in Table 2.1 in the previous Chapter 2, the action of transmission(TX) and reception consume a lot of energy. This is why, at the level of MAC layer, two additional modes have been included in the operation of wireless node, which are the idle and the sleep states. Thus, by alternating the states, it is able to reduce the percentage of time during which the radio remains active to limit its energy consumption.

The duty-cycle mechanism is based on the idea that the radio transceiver must be turned off if it has no more data to send and/or receive. The most known duty-cycling method is the scheduled duty cycle where the time is divided into cycles and transmission takes place only during the active time. Fig 3.2 shows the difference in behavior between a communication without duty cycle (normal) and a communication using a scheduled duty cycle mechanism. However, this method implies that the node must receive or transmit data regularly. New more elaborated duty-cycle methods have been created in recent years as in [54]. In this paper, authors establish a new Optimal Policy Derivation for Transmission Duty-Cycle Constrained in SIGFOX and LORA network. The goal is to find a new policy to maximize the number of reported events (prioritized by their importance) while complying with the ISM regulations.

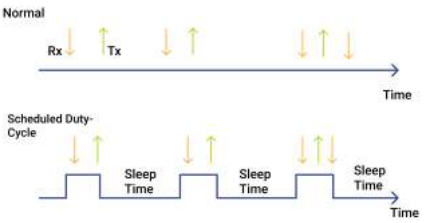


Figure 3.2: Comparison of normal transmission and transmission employed the schedule duty-cycle

This method is present in many wireless communication protocols and nodes admitting these 4 states:

- Receiving (R_x): In this mode the node has two choices: receive a packet or evaluate the network performances such as channel occupancy.

- Transmitting (T_x): The device emits one or more packets during a specific time.
- Sleep(S): The device is inactive (off) and the amount of power consumed is often considered close to zero.
- Idle (D): In this state, the node is still active but is not executing any instructions. However, it can react to external traffic and switch to Transmit/Receive mode at any time. Feeney and Nillson concluded that the idle power is nearly as large as that of receiving data, but consumes less energy than the transmitting mode [55].

Using the duty cycle paradigm, nodes in a network follow a very specific transmission pattern that can be exploited by an attacker to infer certain information such as transmission time.

3.1.2 HARPAGON principle

Before going into the mathematical detail of this framework, we will give a brief overview of the entire functioning of this framework and how it can be exploited by an attacker. HARPAGON is a tool allowing to create several types of attack based on the interaction between the victim and the attackers while finding a compromise between the energy expended and the effectiveness of the attack. The workflow of this tool is described in Fig 3.3.

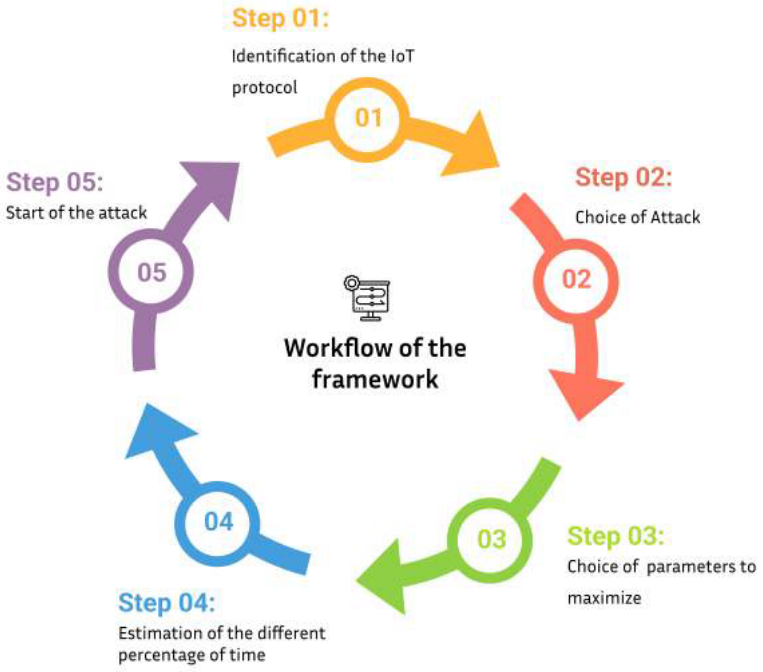


Figure 3.3: System flow of HARPAGON framework

As the effectiveness of an attack depends on the match between the IoT protocol targeted and used to create the attack,

the HARPAGON first step is to identify the communication protocol used by its victim. Indeed, to be able to listen to a network or interact on it, it is necessary to know the communication protocol used. The attacker is equipped of several network interfaces corresponding to several communication protocols and switches between them until he finds the communication protocol. To perform this step, the attacker has several network interfaces corresponding to several types of communication protocol such as 802.11 or 802.14.5. In listening mode, it switches from one interface to another until it finds a communication generated by its victim. Thus it can suppose the communication protocol on which its attack must be carried out. Once this information has been obtained, the attacker must choose the type of attack it wishes to implement. In this step, the attacker has the choice between the most basic attacks known in the literature requiring interaction between his victim and him such as jamming, replay or eavesdropping attacks. With this data the framework asks the user which parameters to maximize. Indeed, the framework was developed to meet two objectives: to maximize the effectiveness of the attack or to minimize its energy expenditure. Therefore, for the first objective, with a given energy cost, the framework calculates the maximum in terms of the percentage of the attack's success. The second is the opposite, i.e. with the maximum percentage of efficiency that the attacker wishes to achieve, the frame calculates the minimum energy necessary. With this information, HARPAGON calculates the percentage of time that the attacker must spend in each operating states. This computation represents the fourth step of the workflow and all the details of the mathematical process are given in the following Section 3.1.3. Finally, the last step consists in carrying out the attack with the data provided at the output of step 4. Thus, the attacker knows the necessary time it must spend in each of the operating states and can launch its attack. For example, if it is a jamming attack, the jamming will take place during the transmission time.

3.1.3 HARPAGON mathematical process

In this section, we explain in more detail the mathematical process that takes place during step 4 of the HARPAGON workflow. This framework is based on the interaction between the attacker node model (ANM) and the transmitter node model (TNM). Based on duty-cycle mechanism and Markov Chain Theory it is possible to represent the states of the Transmitter and Attacker nodes in which they can be at a certain time t.

For example, for the attack to be successful, both the transmitter and the attacker must be in a specific state at the same time. Indeed, to maximize the effectiveness of a jamming attack, the attacker must be in transmission mode at the

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_i
Attacker	R_x	S	D	T_x	S	D	R_x	...
Victim	T_x	S	D	T_x	S	D	T_x	...
IATM	■			■			■	...

■ Passive Attack

■ Active Attack

Figure 3.4: Example of the merge process for the attacker node and the victim node

same time that the victim. On the contrary, for a passive attack such as a eavesdropping attack, the attacker must be in listening mode at the same time as the victim transmits a packet. Table 3.4 illustrates the merging process between the ANM and TNM models, that we have named Interaction Attacker Transmitter Model (IATM). The IATM process slots marked as green (s1 and s7) corresponds to time slots where the eavesdropping attack has a high success probability.

Fig 3.5 describes the respective probability of switching from one macro state to another for the two models, given as $1/T$, where T_{mode} represents the average time spent in the state $mode$. The $mode$ is equivalent to the different operating state involved in the duty-cycle mechanism. Consequently, attacker/transmitter node is characterized by four different states : Attack/Transmitting T_{xi} , Listening/Receiving R_{xi} , Idle D and Sleep S states.

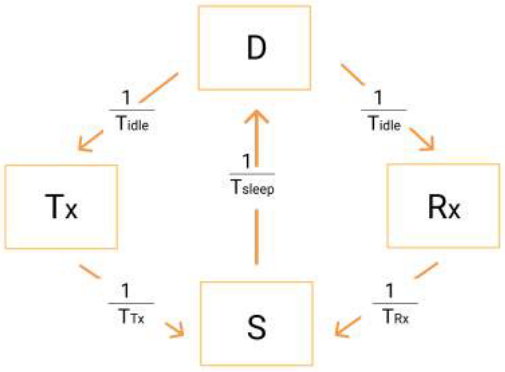


Figure 3.5: Markov Chain of the duty-cycle process of a node

Let us introduce the space state of the attack node F^J , the attacker node model state can be expressed as:

$$S_c^{(J)}(t) \in F^{(J)} = \{R_{x_1}, \ldots, R_{x_M}, T_{x_1}, \ldots, T_{x_M}, D, S\} \quad (3.1)$$

and the steady-state probability vector π^J of this model can

be resolve with the linear system of equations:

$$\begin{cases} \pi^{(J)} \times Q^{(J)} = 0 \\ \sum_{s \in F^{(J)}} [\pi^J]_s = 1 \end{cases} \tag{3.2}$$

where the transition rate matrix from a state to another $[Q^J]_{S_1,S_2} \in Q^{(J)}$ can be defined as:

$$Q_J = \begin{pmatrix} -\frac{1}{T_s} & \frac{1}{T_s} & 0 & 0 \\ 0 & -\frac{2}{T_{idle}} & \frac{1}{T_{idle}} & \frac{1}{T_{idle}} \\ \frac{1}{T_{Rx}} & 0 & -\frac{1}{T_{Rx}} & 0 \\ \frac{1}{T_{Tx}} & 0 & 0 & \frac{1}{T_{Tx}} \end{pmatrix} \tag{3.3}$$

Hence, the Interaction Attacker Transmitter Model (IATM) which corresponds to the interaction between the ANM and the TNM models can be formulated:

$$S^{(IATM)}(t) = (S^{(J)}(t), S^{(Tx)}(t)) \in F^{(J)} \tag{3.4}$$

where $t \geq 0$ is equivalent of the the starting time of the attack. Consequently, the state space of the attack process $F^{(IATM)}$ is given by the Cartesian Product of the two spaces of the single nodes:

$$F^{(IATM)} = F^{(J)} \times F^{(T)} \tag{3.5}$$

and the probability that the attack process at the same time t is $\pi^{(IATM)}(t)_{[S_1,S_2]}$:

$$\pi^{(IATM)}(t)_{[S_1,S_2]} = P\{S^{(IATM)}(t) = (S_1, S_2)\} \tag{3.6}$$

and $\pi^{(IATM)}(t)$ can be computed as:

$$[\pi^{(IATM)}(t)] = \pi^{(IATM)}(0) \times e^{Q(IATM)*} \tag{3.7}$$

where $\pi^{(IATM)}(0)$ represents the state probabilities array at the time $t = t_0$ when the jamming node starts the attack.

By considering the two model independent and identical distributed $\pi^{(IATM)}(0)$ is equivalent to the Cartesian product of the steady-states probability of each process:

$$\pi^{(IATM)}(0) = \pi^{(J)} \times \pi^{(Tx)} \tag{3.8}$$

Consequently, the matrix rate of the state transitions rate $Q^{(IATM)}$ is a matrix 16×16, where the generic element can be computed as:

$$[Q^{(IATM)}]_{(S_1',S_2'),(S_1'',S_2')} =$$

$$\begin{cases} \frac{1}{T_{IDLE}} \\ if(S_1' = S_1''andS_2' = IDLE \neq S_2'') \\ or(S_1' = IDLE \neq S_1''andS_2' = S_2'') \\ \frac{1}{T_{SLEEP}} \\ if(S_1' = S_1''andS_2' = SLEEP \neq S_2'') \\ or(S_1' = SLEEP \neq S_1''andS_2' = S_2'') \\ \frac{1}{T_{RX}} \\ if(S_1' = S_1''andS_2' = RX \neq S_2'') \\ or(S_1' = RX \neq S_1''andS_2' = S_2'') \\ \frac{1}{T_{TX}} \\ if(S_1' = S_1''andS_2' = TX \neq S_2'') \\ or(S_1' = TX \neq S_1''andS_2' = S_2'') \\ -\sum_{(S_1'',S_2'') \in F^{(AP)}}^{(S_1'',S_2'') \neq (S_1',S_2'} [Q^{(AP)}]_{[(S_1',S_2'),(S_1,S_2)]}, \\ if(S_1', S_2' = (S_1)'', S_2'') \\ 0 \qquad otherwise \end{cases} \tag{3.9}$$

Based on the previous model, we can evaluate the effectiveness of an attacker node in respect of **costs constraints** and maximization of the **probability of success attack**.

For that we introduce the concept of cycle time as a time interval between two sleep states of the attacker mode. Consequently the cycle time is equivalent as the sum of the different time that a nodes spent in each state:

$$T_{CYCLE} = T_{IDLE} + T_{RX} + T_{TX} + T_{SLEEP} \tag{3.10}$$

With this previous equation, it is possible to compute the probability for each state:

$$\begin{cases} P_{IDLE} = \frac{T_{IDLE}}{T_{CYCLE}} \\ P_{RX} = \frac{T_{RX}}{T_{CYCLE}} \\ P_{TX} = \frac{T_{TX}}{T_{CYCLE}} \\ P_{SLEEP} = \frac{T_{SLEEP}}{T_{CYCLE}} \end{cases} \tag{3.11}$$

and

$$P_{IDLE} + P_{RX} + P_{TX} + PSLEEP = 1 \tag{3.12}$$

Energy cost

Each state of the duty cycle mechanism consumes a different amount of energy. Thereby, the sum of the time spent in each state for the jamming node represents the total energy expenditure of the attack:

$$\begin{aligned} cost = PW_{TX} \times [\pi^{(AN)}]_{[TX]} + PW_{RX} \times [\pi^{(AN)}]_{[RX]} + \\ PW_{IDLE} \times [\pi^{(AN)}]_{[IDLE]} \end{aligned} \tag{3.13}$$

where P_{mode} represents the power consumption of the each mode.

Attack Probability Distribution Function

For the analyse, we consider an active attack such as jamming or replay attacks. The victim and the attacker must be in the transmission mode at each time. If T is the time of the active attack to be successful, the probability distribution function can be computed as :

$$F_T(t) = 1 - k \times e^{-A(t)} \tag{3.14}$$

where

$$k = 1 - ([\pi^{AP}]_{(Mode,Mode)}) \tag{3.15}$$

and

$$A(t) = \int_0^t \rho_{T \geq t}(\tau) d\tau \tag{3.16}$$

where $\rho T \geq t(\tau)$ is the total frequency for the attack defined:

$$\rho_{T \geq t}(t) = \pi_{T \geq t}^{IATM} \times \Lambda^{(attack)*} \tag{3.17}$$

and $\Lambda^{(attack)}*$ is the transposed vector of $\Lambda^{(attack)}$ defined as the array rate of attack. The generic element $[\Lambda^{attack}]_{S_1,S2}$ representing the attack rate when the state of the attack process is (S_1, S_2) can be defined in respect of the number of slots M as:

$$\begin{cases} [\Lambda^{(attack)}]_{[S_1,S_2]} = \Sigma_{m=1}^{M}([Q^{IATM}])_{[(S_1,S_2),(Tx,Tx)]} \\ 0, otherwise \end{cases} \tag{3.18}$$

3.1.4 Theoretical evaluation

Before to implement this model in a real-test bed explained in Chapter 5, we evaluate our mathematical model with the Mathematica software [56]. We evaluate the effectiveness of the HARPAGON framework in respect of its two objectives: maximize the probability of attack success and minimize the energy cost.

Maximize the attack probability success

The first goal of HARPAGON framework is to maximize the attack probability of attack success in a certain interval time t, which corresponds to $max[F(t)]$, by giving an energy consumption lower than a certain threshold, meaning cost ≤ c.

In other term, if you based on equation 3.13, the cost function can be characterized as:

$$\begin{aligned} cost = PW_{RX} \times P_{RX} + PW_{TX} \times P_{TX} \\ + PW_{IDLE} \times P_{IDLE} \leq c \end{aligned} \tag{3.19}$$

where c is the optimal energy cost and the quadruple (P_{RX}, P_{TX}, P_{IDLE}, P_{SLEEP}) by:

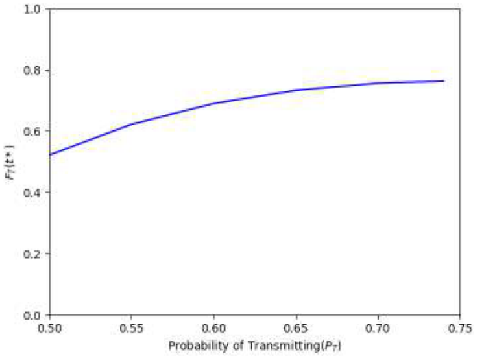
$$\begin{cases} 0 \leq P_{TX} \leq P_{TX}^{max} \\ P_{IDLE} = k \\ P_{RX} = \frac{cost - PW_{IDLE} \times P_{IDLE} - PW_{TX} \times P_{TX}}{PW_{RX}} \\ P_{SLEEP} = 1 - P_{RX} - P_{IDLE} - P_{TX} \end{cases} \tag{3.20}$$

without less of generality, we arbitrarily assign the value of P_{IDLE} as comprised in $0 < k \leq 1$. The value of P_{RX}^{max} can be calculated by the inequalities:

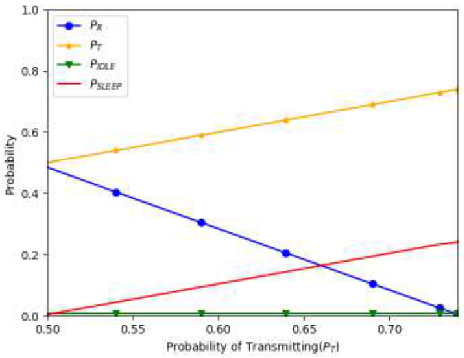
$$0 \leq P_{RX} \leq 1, 0 \leq P_{TX} \leq 1 \tag{3.21}$$

Based on real values of power consumption of an Atheros chipset wifi adapter found in [57]. We compute the maximum attack probability for several associated cost, namely $cost = 0.35(W)$, $cost = 0.4(W)$, $cost = 0.5(W)$ and $cost = 0.6(W)$.

Moreover, the probability distribution function $F(t)$ does not depend on the specific values of t and T, but rather on the t/T ratio. Consequently, in the following study of this scenario, the ratio will be considered as a system parameter instead of t and T separately. After, several simulations by varying the t/T ratio, we found that the optimal solution is reached when this latter is equal to $t/T = 2$.



(a) Values of $F(t)$ corresponding to energy cost equal to c= 0.5W and $t/T = 2$



(b) P_{RX}, P_{TX}, P_{IDLE} and P_{SLEEP} quadruples with $cost$ = 0.5.

Figure 3.6: Results for maximizing the probability of attack success

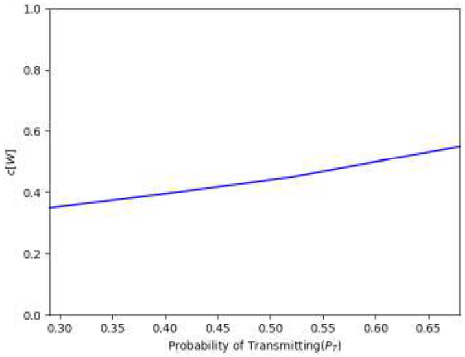
Fig 3.6 demonstrate the results obtained with a cost $c = 0.5$ and $t/T = 2$. With these values and the developed framework we evaluate the $F(t)$ values according to PT_x and the ratio $t/T = 2$, as demonstrated in Fig 3.6a. Consequently, we note that the maximum value of $F(t)_{max}$ for a cost at 0.5 W is $F(t)_{max}$ = 0763712 for a PT_x value at 0.74%. By reporting the PT_x value on the Fig 3.6b, we obtain the quadruple:

$$\begin{cases} P_{RX} = 0.005, P_{TX} = 0.74, P_{IDLE} = 0.01, P_{SLEEP} = 0.25 \\ \text{if } (c = 0.5W) \end{cases}$$

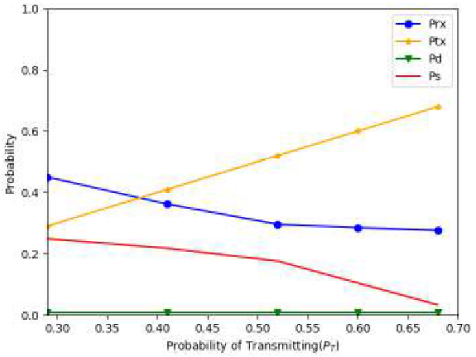
Minimization of the energy consumption

On the contrary, in the second objective, we try to perform an attack with a given probability that the attack is a success and calculate the equivalent energy cost. In other words, the goal is to minimize the energy consumption by taking into account that the attack must occur within a certain time interval, namely $F(t) \geq t$.

Fig 3.7a, show the variation of the cost according to the $F(t) = 0.7$ and $t/T = 2$ parameters. In this situation, the goal is to minimize the energy cost, consequently the C_{min} is equal to 0.35 W and the corresponding PT_x value is 0.29%. With the same process that for the first objective, we report



(a) Values of c corresponding to $F(t) = 0.7$ and t/T =2.



(b) P_{RX}, P_{TX}, P_{IDLE} and P_{SLEEP} quadruples with $F(t) = 0.7$ and t/T=2.

Figure 3.7: Results for minimizing the energy cost of the attack

the PT_x value on the Fig 3.7b to obtain the quadruple:

$$\begin{cases} P_{RX} = 0.45, P_{TX} = 0.29, P_{IDLE} = 0.01, P_{SLEEP} = 0.24 \\ \text{if (F(t)=0.7\%)} \end{cases}$$

Discussion

In these two sections, we prove that a tradeoff between the probability of the attack success and the energy consumption is present. However, we show that with HARPAGON, we can calculate the tradeoff between these two values. Indeed, we demonstrate that it is possible to maximize the probability of success of an attack according to a given energy cost. Or, on the contrary, minimize the energy consumption by imposing a success threshold in terms of the probability of the attack. This framework is general and can be adapted to several protocols relying on duty cycle mechanism. Finally, it is possible to combine this framework with several types of attack that depend on the interaction between the attacker and the victim such as jamming or replay attacks. In these previous section, the mathematical model is developed to maximize the time spend in the T_x state and consequently the effectiveness of active attack. However, it is possible to modify the framework slightly to optimize the attacker's listening time to minimize their energy cost during a passive attack. In Chapter4, we show the results obtained with the framework combining an experimental passive and active attack in a real testbed.

3.2 FOLPETTI: a framework to circumvent the channel hopping method.

Wireless communication protocols have continued to become more complex and specialized in recent years. The 802.11 protocol has seen one of these amendments dedicated to wireless communication between two autonomous vehicles [58]. Bluetooth now contains an improved and less power-hungry version for IoT networks. These specifications continue to increase and seriously complicate the issues of quality of service and security. Indeed, coordinating all these communication protocols in the same environment can quickly become problematic and improve anomalies. Studies have shown, for example, that 802.11 communication could interfere with Bluetooth communication [59]. In an attempt to solve performance degradation problems, new mechanisms have been developed in order to guarantee the coexistence of these protocols. One of the best known to avoid the impact of interference is frequency hopping. Frequency hopping allows a transmitting and receiving device to change their transmission frequency to avoid interference. This method has also proven to be effective against the different types of attacks that exist in wireless networks [60]. Indeed, for eavesdropping attack, the attacker must listen on the same transmitter channel as his victim. Similarly, if the attacker wants to be effective during an active attack, he must send data on the same communication frequency as the rest of the targeted network.

As in other areas, frequency hopping method strategies have also evolved over time. This ranges from basic algorithm like pre-defined frequency hopping mechanism to advanced machine learning methods using real-time network characteristics. This advance was provoked for one main reason: to prevent the more elaborated attacks. Indeed, in some works the authors managed to deduce frequency hopping patterns when these are based on simple implementation mechanisms such as pre-definied pattern. Indeed, in the case of a non-evolving pattern, the attacker was able to deduce the pattern by listening to the victim's transmissions beforehand. In [61], the authors demonstrate that an attacker knowing the channel hopping pattern in 802.15.4e has more efficiency than an attacker without degrees of knowledge. Indeed, this process is useful against environmental interference and simple basic attacks such as constant jamming attacks, but remains vulnerable to selective attack. The attacker's intelligence in this paper has been simulated, it is in this sense that we wanted to create a new smart framework having the ability to infer the frequency hopping pattern autonomously. In these following sections, we show that this new framework is effective even if the strategy of frequency hopping employed by the victim is elaborated.

3.2.1 Background of Frequency hopping

Frequency hopping is both an interference mitigation method and a reaction strategy against attacks. Several wireless communication protocols envision the use of different frequency bands for communications, and each frequency band is separated into channels. Exploiting this paradigm, channel hopping has been proposed to regularly or adaptive change the communication channel assigned to nodes in the network to limit both collisions and interference generated by neighboring networks. Channel hopping methods are integrated in several standard network protocols such as 802.15.4 protocol with the Time Slotted Channel Hopping (TSCH). The TSCH slotframe is represented as a matrix. Each cell corresponds to a timeslot and a channel offset which is translated into the radio frequency to use. The slotframe is repeated as long as the network is running and cells are assigned to communicating devices. In the Bluetooth protocol, several channel hopping methods have been implemented. The most basic is the Frequency Hopping Spread Spectrum (FHSS) that consists of following a pseudo-random pattern defined by the two communicating nodes in advance. Several years later, the Adaptive Frequency Hopping Spread Spectrum (AFH) was included in Bluetooth to adapt the choice of the channel according to the environment. Indeed, the term Adaptive is used to indicate, that the channel conditions are constantly monitored to identified the "bad" communication channel, i.e: occupied or low quality channels. Thus, the "bad" channels are included in a blacklist and become inaccessible during the next selection process until they become usable again. For the 802.11 protocol, the FHSS method was included from the beginning of its creation on the physical layer. Similar to WiFi protocol, we find Long Range- Frequency Hopping Spread Spectrum in LoraWan protocols. The nodes randomly distribute the packets over a defined frequency bandwidth which includes 137kHz, 336kHz and 1.523MHz depending on the region. We also observe the employment of channel hopping methods in cognitive radio to avoid noise effects and improve the control channel saturation problem. Finally, this solution is also employed in propriety protocols, such as that used for DJI drones where FHSS is one of several integrated methods.

As seen previously, many strategies to select the future communication channel have been elaborated in the literature. This ranges from basic algorithm like pseudo-random channel selection mechanism to advanced machine learning methods using real-time network characteristics. Consequently, we can categorize these different channel hopping methods according to two main classes, as follow :

- **Basic channel hopping methods**: In this class we can include all basic methods such as pre-defined pattern or pseudo-random pattern. In this class, the pattern

is not scalable and does not adapt to environmental conditions. Most of the time, the network nodes have previously established the list of future transmission channels before starting their transmission. In the case of random choice, the nodes are exchanged the seed allowing the generation of the numbers of the future channels.

- **Smart channel Hopping methods**: In this category, channel hopping method adjust their strategy according to their environment. The selection can be made by simple statistical or with more elaborate algorithms such as machine learning. For example, nodes can assess the level of network occupancy based on well-known metrics such as Received Signal Strength Indication (RSSI). In this situation, a threshold is defined upstream and if the value of the metric evaluated exceeds this threshold, reactions are applied. The reactions can be to blacklist channels that do not meet the criteria for a certain time, for example. In recent years, new smart channel hopping methods based on machine learning algorithms have been emerged. With this process, channel hopping methods become more autonomous in the development of their strategy and manage to respond to changing environments. In [62], authors evaluated several Multi-Armed Bandit algorithms to improve the Time-Slotted Channel Hopping (TSH) in IEEE 802.15.4 protocol. The evaluation suggests that this new process can significantly improve the packet delivery ratio compared to the default TSCH operation.

3.2.2 FOLPETTI: an overview

As for the previous framework, we will first give an overview of the workflow of the FOLPETTI framework before going into the mathematical details. In recent years, new work to automate anomaly detection has emerged. Indeed, due to the high heterogeneity and granularity of network components, the need for modular anomaly detection tools has increased. In this sense, the authors of [63] are developing an artificial intelligent framework for detecting anomalies and based on closed-loop automation. A closed-loop automation is used in several aera such as network softwarization domain and composed of several phases which are: "detect", "learn", "decide", "policy" and "act" [64]. Based on the automation loop, we develop a framework to independently infer and follow the channel hopping pattern of a network. Therefore, adapting this model to the attacker's logic, the process of this framework is separated into 4 steps as summarized in Fig 3.8.

The first step concerns the observation of the environment. In this case, the observations are captured by the attacker, and since our main goal is to infer how the channels are used,

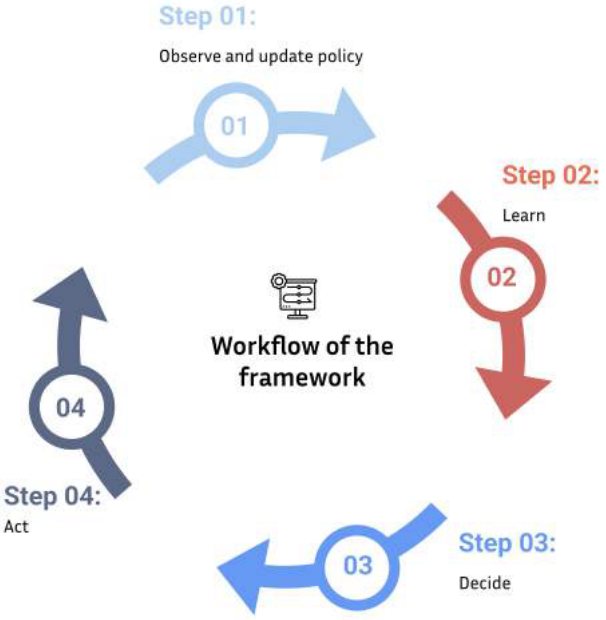


Figure 3.8: System flow of FOLPETTI framework

the attacker will base his observations on them. With these observations, the attacker can build or update its learning model which is represented on the diagram by the second step. This model is in turn used by the decision-making process to decide on the potential action to take based on possible moves and experience gained. Once the action is decided, the framework is in step 4 titled act. Applied to this situation, in this step the attacker conducts its attack on the channel determined by the previous steps. This framework was designed to be as extensible as possible, hence the attacker can decide at the beginning of the execution of this framework to choose between a passive or active attack. Finally, the framework observes its environment again to notice the potential consequences of its attack. With this information, the framework policy is updated. The main objective is to provide a policy that minimizes the regret, that is to say the quantity that expresses what the policy has caused to lose compared to the choice of the best machine.

3.2.3 FOLPETTI: the process

Efficient attacks against channel hopping has two fundamental requirements: i) it should not depend on specific assumptions on the victim's hopping pattern, and ii) it needs to be continuous in time. Moreover, to select the optimal channel, i.e., where the attack will have the most effect, the attacker needs to solve the Exploit-Explore dilemma. Indeed, the attacker must find the compromise between exploiting the channels that had the most effect before or exploring

other channels in order to find if among the unexploited there would not be one that would lead to a better strategy. In the literature, several Multi-Armed Bandit algorithms have been implemented on the channel hopping side in order to choose the optimal transmission channel (channel with the least interference) [65]. The main advantages of this algorithm are that it does not require any specific assumptions at its beginning and that the learning is done continuously and online. Finally, in MAB algorithms, at each iteration, an agent acts on the environment according to a predefined policy and receives a reward. Therefore, this type of algorithm seemed perfectly appropriate for our workflow framework and satisfied the fundamental characteristics of attack against channel hopping.

The Multi Armed Bandit algorithm MAB can be modeled as a Markov Decision Process (MDP). The MDP can be described via five tuples $\langle S, A, P, R, \gamma \rangle$, where:

- S is a finite set of states s;
- A is a finite set of actions a;
- $P_a(s^n, s^{n+1})$ is the probability that an action a in state s^n in time $n+1$;
- $R_a(s^n, s^{n+1})$ is the expected immediate reward received after transitioning state s^n to state s^{n+1} due to action a;
- $\gamma \in [0, 1]$ is a discount factor.

The main objective of a MDP is to find a policy π that associates an action to each state $\pi : S \rightarrow A$ to maximize the reward. Therefore, the agent tries to maximize his reward by selecting the optimal channel. At each time instant n, the attacker may stay in the previously selected state s_i, or move to another state s_j. Therefore, we define the possible actions as the set $W = s_1, s_2, ..., s_S$ of the available channels. We assume that the reward is one $R_a(s^n, s^{n+1}) = 1$ whenever the newly selected channel is used by a victim. Otherwise, the reward is zero $R_a(s^n, s^{n+1}) = 0$.

To solve this online decision problem, inspired by the work in [65], we apply the Thompson sampling algorithm as a policy. The prior distribution beta(α_j, β_j) for each access trial is a Beta distribution with parameters α_j and β_j. We denote as μ_j the event where the attack is successful ($R_a = 1$). For action s, the probability of success is given by:

$$p_j(\mu_j|S_j) = \frac{\Gamma(\alpha_j + \beta_j)}{\Gamma(\alpha_j)\Gamma(\beta_j)}(\mu_j)^{\mu_j - 1}(1 - \mu_j)^{\beta_j - 1}; \tag{3.22}$$

where $\Gamma(\cdot)$ is the gamma function. If the state s is selected in round t and returns a reward R_a, the prior distribution for the mean reward of arm s can be updated via the Bayes rule. By utilizing the conjugacy properties, the posterior distribution for the mean reward of each arm is also a beta distribution with parameters updated based on the following rules [**R-1**]:

$$(\alpha_s, \beta_s) \leftarrow \begin{cases} (\alpha_s, \beta_s) & \text{if } a_t \neq s; \\ (\alpha_s + R_a, \beta_s + 1 - R_a) & \text{if } a_t = s. \end{cases} \tag{3.23}$$

3.2.4 FOLPETTI combined with Jamming Attack

In this section, we show how FOLPETTI framework can be coupled with a jamming attack. We implement a constant jamming attack to continuously jam the victim's channels. Based on FOLPETTI model, the attacker follows the victim's channel selection and jams them as described in Algorithm 2. Notice that we assume that the attacker can jam a single channel per time instant.

Algorithm 1 FOLPETTI Algorithm

Require: j : channel index, c : total number of channel accesses, tj : number of successful transmissions so far

1: $\alpha_j = \beta_j = 1$
2: $t_j = c = 0$
3: **while** True **do**
4: **for** `all j` **do**
5: sample $r_j \sim$ beta $(\alpha_j + t_j, \beta_j + t_j)$
6: m = argmax $\{\bar{r}_j\}$;
7: c ++ ; JAM();
8: **if** channel is occupied **then** t_m+ = 1

We determine the success of the jamming attack based on the RSSI. This metric, unlike other metrics such as the PER or PDR, does not require the attacker to spend a lot of time in listening mode. Therefore, the attacker can remain active for the whole duration of the attack. Indeed, we have observed a drop in the RSSI when an attack takes place whether on the side of the transmitter or the attacker. Unlike other works, we do not rely on ACK packets to define the success of our attack. Consequently, the attacker must not wait to receive this packet to recover a reward. The update of the policy is done in a smaller time and the attacker will converge faster to the optimal solution.

3.3 Formal Security Analysis

In this last section, we present a formal security analysis of this two new frameworks. The main target being IoT networks, we relied on the formal IoTSaT framework developed in [66] for the security analysis of IoT. This framework is designed to automatically unveil a complex chain of attack vectors related to the pre-defined adversary's goals. It includes four components: i) IoT Topology Model, ii) The Attack

Constraint Model, iii) The Interaction Constraints and iv) Threat Model.

The first component aims to represent the composition of the targeted network. In this context, to simplify the analysis, we rely on a basic two-node network. We consider that a sensor (transmitter) periodically sends data concerning the temperature of a motor of a machine to a controller (receiver). The controller has the power to automatically stop the machine if the temperature becomes excessive. Therefore, the IoT topology model can be expressed as:

Let $Rep(XS, C)$ denote the reported temperature X, as perceived by the sensor S and further reported at the Controller C. If $Rep(XS, C)$ exceeds a threshold δ, the controller generates an action which is to stop the operation of a machine $Cmd(STOP)$. Consequently, the actuator's function for this example is:

$$R_{1,1} = \{(Rep(X_{S,C}) \vee X > \delta), Cmd(STOP)$$

The second element, the attacks constraints can be subdivided into two objectives: the adversary's capabilities and the adversary's goal. Consequently, the adversary's capabilities and adversary's goal for the HARPAGON framework is modelled as follow:

$$\begin{gathered} \textit{HARPAGON Goal} : \\ Act(N_1) \wedge Max(Impact) \wedge Min(Energy) \\ \textit{HARPAGON Capabilities} = CN = 1 \wedge CL = 1 \wedge EC \leq c \end{gathered}$$

where *Act()* corresponds to the Action of Attack, *Max()* the maximum function and *Min()* the minimum function. *CN* represents the maximum of victim node and *CL* the number of link. To finish, *EC* is equivalent to the energy consumption. Consequently, in this context, the goal of the attacker is to attack one node N and one link while maximising its impact and minimising its energy consumption. The energy consumption is lower or equal to the cost c.

The attack constraints for the FOLPETTI framework is more or less the same and is:

$$\begin{gathered} \textit{FOLPETTI Goal} : \\ Act(N_1) \wedge Max(Impact) \wedge Min(Learningtime) \\ \textit{FOLPETTI Capabilities} = CN = 1 \wedge CL = 1 \end{gathered}$$

The goal of the attacker who uses FOLPETTI framework is to attack one node N and one link while maximizing its impact and minimizing its learning time to infer the channel hopping strategy of the victim.

IoTSAT classifies IoT threats as interlinked threat vectors, where the injection of one vector by the attacker can trigger

a chain reaction, impacting multiple IoT entities. Indeed, IoTSAT framework includes five threats for the sensor context: denied, incomplete, inconsistent, tailored and fabricated contexts. Basing our analysis and the IoTSAT models, three main threat vectors can be exploited by the two new frameworks if sensor devices are used. Indeed, we saw in the previous section that these two types of framework can lead to several attacks and tries to intercept the maximum number of victim packets. Consequently, with these frameworks an attacker can intercept, delay, block or corrupt a packet. Thus, the context can be denied(DC) if the information is blocked at the node level or at the network level. Moreover, the context can be incomplete(IpC) if the information is delayed or not fully delivered (e.g., corrupted packet, fragmented packet but part is missing). Finally, the context can be tailored(TC) if the information has been modified directly at the level of the node or during the sending. Witht the IoTSAT framework, the threat can be expressed :

$$\begin{aligned} DC^t &= \neg Obs(X^t_{a,b}) \vee \neg Reach^t(S_b, C_d) \\ IpC^t &= Obs(X^{t+1}_{a,b}) \vee \neg Reach^t(S_b, C_d) \vee Reach^{t+1}(b, d) \\ TC^t &= Obs(X^t_{a,b}) \wedge Rep(\overline{X}_{b,d}) \wedge (X \neq \overline{X}) \end{aligned} \tag{3.24}$$

where $Obs(X^t_{a,b})$ corresponds to the observation of an event a effectuated by the Sensor b at the moment t with the X(int) value. The function $Reach^t(S_b, C_d)$ returns true, if a valid communication connection exists from Sensor b to a Controller d. Eventually, $Rep(X_{b,d})$ is the predicate returning true if the value X(int) of the event a is received by the Controller d and reported by the Sensor b.

If these two frameworks exploit the threats we have just defined above, the trigger function of the controller will also be impacted. Indeed, the controller process takes a decision made by the function $F()$, whose parameter a corresponds to the data transmitted by the sensor S. In case the context is incomplete, the received value is insufficient for the decision process and the trigger will also be incomplete (IT). If the context is denied, the trigger function will be blocked (BT). Finally, with a tailored context, the decision process is altered by the received value a, the trigger is false (FT). We define the threats for the triggers function as follows:

$$\begin{aligned} BT^t &= DC(S_b, C_d) \\ IT^t &= \neg MCF(C_d, F_a) \\ FT^t &= \sum_{\forall S_b \in S^{ad}} [TC(S_b, C_d)] \geq v_{ad} \end{aligned} \tag{3.25}$$

where *MCF* is the Minimal Context Fusion which corresponds to the sum of the Reach(S_b, C_d).

$$MCF(C_d, F_a) = \sum_{\forall S_b \in S^{ad}} [Reach(S_b, C_d)] \geq w_a d \tag{3.26}$$

Indeed for certain scenarios, several values of sensor may be needed to compute the return value of the trigger function F_a. Consequently, if the connection link between a Sensor S_b is lower than the number of data required $w_a d$ to compute the trigger function, the trigger is incomplete. If the trigger function can be computed, but returns a false result (FT), it means that the sum of the tailored context is greater than the number of ratio values needed to compromise the result v_{ad}.

In an IoT system, most of the trigger functions conduct to an action. In our example, the trigger function initiates the shutdown of the motor and therefore of our machine. Always based on the IoTSAT framework, we see that an incomplete or blocked trigger function conducts to the denied or delayed action (DA). Moreover, an False Trigger (FT) causes an incorrect actuation (IA). These two consequences are formulated as follows:

$$\begin{aligned} DA(A_f, E_i) &= IT(C_d, F_a) \wedge BT(C_d, F_a) \\ IA(A_f) &= FT(C_d, F_a) \vee Reach(C_d, A_f) \end{aligned} \tag{3.27}$$

where A_f is the f actuator and E_i is equivalent to the i Service.

We have seen that our frameworks can make it possible to launch several types of attacks such as eavesdropping, replay or jamming. If we combine this formal security analysis with our frameworks, the consequences can be a complete shutdown of the machine or, on the contrary, lead to a fire and much more serious real consequences. Indeed, if the data sent by the sensor is blocked or delayed, the controller never stops the machine in time, even if the temperature is higher than the warning threshold. On contrary, if the data is modified, the controller can take the decision to stop or no-stop the machine in inappropriate moment.

3.3.1 Conclusion

In this Section, two new frameworks to implement several types of denial-of-service attacks have been proposed. The first framework, HARPAGON, is based on the vulnerabilities created by the duty-cycle mechanism of IoT protocols.

Based on Markov Chain theory process, the main goal of HARPAGON is to "predict" the optimal moment of the attack and then increase the offender's performance. Moreover, the employment of this framework considerably reduces the energy expenditure of the attacker. We then presented FOLPETTI, a novel smart attack that operates in an unknown network in the absence of a-priori information about the network itself. Based on Multi Armed Bandit algorithm, FOLPETTI has the ability to understand and predict the behaviour of victims and more particularly their channel hopping strategy. The theoretical evaluation which was made in this chapter allowed us to confirm our models, however in the following chapters we evaluate these two frameworks through simulation and experimentation.

Evaluation of the FOLPETTI framework with a new jamming module on ns-3

4

In this thesis, we decided to evaluate our two frameworks presented in the previous chapter coupled with jamming attacks. Indeed, these types of attacks intentionally attempt to occupy a channel to prevent transmission between two nodes and lead in most cases to denial of services. In the IoT network, in addition to reducing communications between two devices, this type of attack also has consequences that are not visible at first view, such as an increase in energy consumption. Jamming attacks are not recent and therefore many methods of detection and countermeasures such as channel hopping have been created. However, we show thanks to simulations on ns-3 that it is possible to thwart this system due to the FOLPETTI framework and to make jamming attacks efficient again. We also reveal with these simulations the damage that FOLPETTI can cause coupled with a jamming attack on the batteries of IoT devices.

In this chapter, after providing the motivations for simulating jamming attacks, a brief state of the art of simulation tools existing in the literature is given. Realizing that no existing simulator corresponded to our needs, we decided to create our own jamming module on ns-3 which we detail in Section 4.4. Finally, after validating its behavior in Section 4.5, we analyze the performance of our FOLPETTI framework coupled with a jamming attack in section 4.6. A comparison with different attacks already presented in the literature is given as well as the consequences of FOLPETTI attack in terms of energy consumption.

4.1 Why simulate jamming attacks ?

Due to their nature, accurately analyzing the behavior of real-world jamming attacks can be cumbersome and time-consuming. Indeed, this type of attack is based on the vulnerabilities of the physical and data link layers and depends on a multitude of parameters. The effectiveness of a jamming attack is based on many parameters such as transmission properties (e.g. modulation, power), network characteristics (e.g. routing), or the strategy of the jammer as well as its position. The impact of several types of jammers is studied in [35] according to their distance from the victim nodes and the size of the packets. The authors conclude that the closer the attacker is to his victim, the more effective it is. Panim et al. wonders if random positioning of a jammer can be more effective than when the attacker's choice of position is strategically determined [67]. They deduce the aggressor has

more impact on the network when the jammer is situated next to a node where a lot of data transits.

Moreover, the composition of the environment can also distort the analysis of the effectiveness of these attacks. Indeed, the type of obstacle between the jammer and the victim can lead to different measures. In [68], the authors show that the signal received is altered depending on the placement of the jammer and therefore the type of obstacle. The number of different networks nearby is also a parameter that can make the success of the attack diverge. Indeed, several communication protocols can communicate in the same frequency band and generate unintentional interference. This is the case of the Bluetooth and WiFi protocols which communicate on the same 2.4 GHz band as demonstrated in [69]. Therefore, these unintentional interferences can be interpreted as the result of a jamming attack and skew the results. For all these reasons, it is very difficult to reproduce the same identical effects twice. Indeed, in most scientific articles, the environment is not completely described, and it is impossible to guess the number and type of neighboring networks. This is why an accurate study of jamming attacks requires expensive means such as an anechoic chamber or a Faraday cage as shown in Fig 4.1.

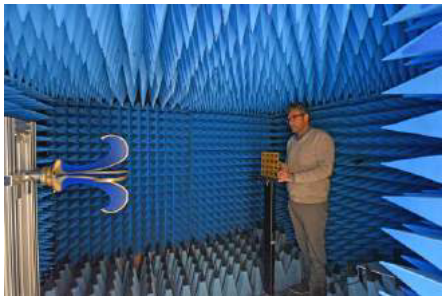


Figure 4.1: Anechoic Chamber [1]

In addition to the reproducibility aspect, simulation also makes it possible to evaluate parameters that are currently relevant and difficult to obtain. Indeed, as seen in chapter 3, the concept of *green attacks* is emerging in the literature. However, calculating the energy spent by the network card is extremely time-consuming and also depends on many parameters such as the hardware used. In several simulators, a precise estimate of the power consumption of the network card is already present. Therefore, the estimation of energy consumption can be done more easily with a simulator. Furthermore a simulator can be used to evaluate jamming attacks in a large network.

Finally, the increasing use of ML algorithms in the coming years will lead to a significant change in the threat landscape, as we explained in chapter 2. Jamming attacks exploiting this technology are becoming more adaptive, more resilient, more reactive, and less identifiable by existing detection methods. In most cases, ML algorithms are data-driven, and collecting them in real life takes time. Several experiments under identical conditions are needed to validate the data. Additionally, several ML frameworks can now be integrated into network simulators. For example, Matlab now includes a comprehensive machine learning tool to easily automatically and select hyperparameters for ML algorithms [71]. In 2019, a new module allowing communication between the TensorFlow framework, an open source machine learning tool developed by Google, and ns-3 was created [72]. In this situation, simulating jamming attacks could save researchers a lot of time and money.

For all these reasons, reproducing jamming attacks on a simulator can save researchers time by first giving an idea of their feasibility. Moreover, since the main purpose of creating a new type of attack is to study vulnerabilities, the simulation tool can also be used to evaluate new detection and mitigation methods.

4.2 State-of-the-art on jamming attack simulation tools

In this section, we focus on the tools and methods available to simulate a jamming attack on several wireless communication protocols. Jamming attacks in wireless networks is not a recent topic in the literature. Many works have already attempted to simulate jamming attacks in order to evaluate them and improve security systems. These works have distinct advantages and disadvantages that we have reported in Table 4.1.

OMNeT++ : OMNeT++ looks more like an emulator than a simulator. Composed of multiple modules and written in C++, it allows the creation of network simulators as explained in [73]. In [74], the authors proposed a new intrusion detection system (IDS), capable of differentiating jamming attacks in the wireless body area network (WBAN). This study is conducted in the OMNeT ++ simulator framework with the ZigBee (802.15.4) communication protocol. However, no source code is provided in this paper and no jamming module is officially found on the OMNeT++ platform.

Matlab : This tool is a numerical computing and programming platform for analyzing data, developing algorithms, and creating models. It is not a network simulator strictly speaking, but it is possible to simulate several communication protocols with it. Indeed, in [75], the authors used the Matlab simulator to design and implement a new countermeasure against Smart Jamming Attacks on Long Term Evolution (LTE) protocol Synchronization Signals. Another study was performed with Matlab to create a Smart Jamming Attack in [76]. This new type of attack is based on Deep Reinforcement Learning and aims to evade a mitigation system in the WiFi protocol (802.11). The main flaw of this tool is that it is not free and no official jamming module was found. However, it is recognized for its power and machine learning tools are already included.

NS-2: Network Simulator 2 (NS2) is a discrete event simulator specially designed for network research. It is based on two languages, C++ to write the simulator code and a Tcl interpreter to execute the command script. The behavior of the ZigBee protocol against jamming attacks is described and analyzed in [77]. Indeed, after this investigation, Sachin D. Babar et. al proposed a novel efficient and realistic defensive

mechanism against jamming attacks. One of the primary advantages of this simulator is that it is entirely free and open-source. However, it is no longer maintained by the community since the creation of its big brother ns-3. It is possible to find some code to simulate jamming attacks on ns-2, but since the simulator has been obsolete for 20 years, the communication protocol models are getting old.

NS-3: This simulator is the evolution of ns-2 and the first version appeared in 2006. Like ns-2, Network Simulator 3,(NS3) is free, open-source and maintained by a worldwide community. A recent module for ns-3 was proposed for jamming attacks for LoRaWan communication protocol in [78]. The main advantage of using this simulator is the integration of a module to assess the energy consumption of each node. Its open-source aspect remains a very significant point for the reproducibility of the search, however, this may lead to more implementation problems than on paid platforms. Modules already present on ns-2 have not been recreated on it for lack of time. As a result, some functionality is no longer available on this simulator.

Custom Simulators Several research laboratories have developed in-house their own network simulator for jamming attacks. Indeed, after an analysis of diverse types of attacks in [79], the authors propose an attacker model in the Wireless Sensor network and develop their own platform to validate estimates on the energy consumption of the victim node. Although the creation of own simulators can address the specific needs of creators, they stay private.

Based on this analysis, we decided to create a new jamming attack module for the 802.11 protocol in ns-3. module usable by the entire research community and allowing easy reproduction of the results. As a result, our choice fell on a free simulator known to the community: the discrete event simulator Network Simulator-3 (NS3). Moreover, jamming attacks have direct consequences on the energy consumption of their victim. The choice of a simulator already integrating an energy model was a key point in order to assess their consequences. As seen in the previous chapter, smart attacks have emerged in the literature. Therefore, our choice also fell on a simulator where the integration of ML algorithms has already been carried out. Indeed, since 2018 two modules allowing the use of ML algorithms are present in ns-3: ns3-gym and open-ai. Finally, we wanted to first focus our studies on jamming attacks on the WiFi protocol because it is widely utilized in many areas of wireless networks. Indeed, its constant evolution now makes it a possible protocol for IoT devices with the *802.11 ah* version or for vehicular networks with its *802.11 p* version. And these models have already been developed on the simulator by the active community of ns-3.

Table 4.1: Survey of jamming attack simulation tools

Network Simulator	Communication Protocol	Advantages	Disadvantages
OMNeT ++	ZigBee (802.15.4)	► Generate Real Traffic ► Energy consumption data	► Slow ► No real Simulator code
Matlab	Long Term Evolution (LTE) protocol - WiFi (802.11)	► High performance	► No-free
ns-2	ZigBee (802.15.4)	► Energy consumption data ► Modularity and Popular	► Not maintained ► Jamming Attack module not provided
ns-3	LoraWan	► Large Community - Free and Open-Source ► System modularized	► Active maintainers are required
Custom Simulator	WiFi (802.11), ZigBee (802.15.4)	► Support for many communication protocols ► Executes the same embedded software codes ► Calculation of software and hardware energy consumption	► Private simulator ► Not easily reproducible

4.3 NS-3 in detail

Ns-3 is the logical continuation of ns-2 and won over the community already present on ns-2 as soon as it was released. It was created to improve the realism of the models as mentioned by its authors and, its use has been simplified. Indeed, compared to its predecessor, it is entirely coded in C++ and python biding, it is no longer necessary to juggle between a mixture of TCl and C++. Ns-3 is a discrete event simulator, which means that the simulated system is modeled as a series of discrete events that change the state of the system. Moreover, it is light to use because it is composed of a set of modules that the user can import according to his needs. Its lightness is also due to the fact that the user calls its code via command lines (CLI), without using a graphical interface. A graphical interface, named NetAnim, also exists even if its applications are limited to simple visualization purposes as shown in Fig 4.2.

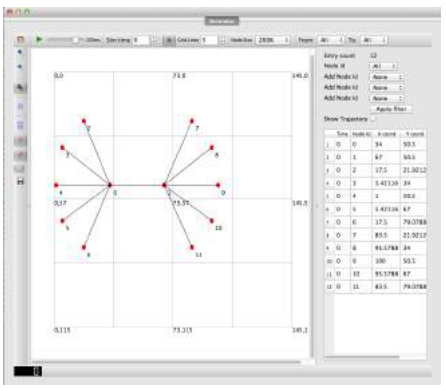


Figure 4.2: Graphical Interface of NS-3

All physical devices such as a computer or an IoT device are

abstractly represented as a node under Ns-3. As in reality, the node must integrate a network interface card (NIC) to have connectivity. This function is imitated with the NetDevice object on NS-3 and makes it possible to simulate the software and hardware behavior of a card. Therefore, a NetDevice object is "installed" in a node to allow it to communicate with other nodes. When installing the NetDevice element, the developer can specify the type of protocol he wants to use, such as WiFi, cellular, or Ethernet communication protocol. Each networking protocol comes with its own communication channel models.

Regarding the WIFI protocol, the behavior of the PHY and MAC layers is modeled on ns-3 for several types of topology such as ad-hoc, infrastructure and mesh modes. Since the last version (ns-3-36), it is now possible to find most of the existing WIFI amendments: from the oldest, *802.11 a*, to one of the most recent, *802.11 ax*. In addition, supports have also been developed for specific use cases of certain wifi amendments, such as the WAWE model which allows the creation of system architectures for vehicular wireless communications with the *802.11 p* protocol [80]. More generally, it is also possible to integrate higher-level communication protocols like routing protocols available on the network layer like OSLR or TCP.

NS-3 supports different mobility models to simulate a mobile attacker, such as constant or random mobility models. This functionality can be very advantageous when one wishes to estimate the effectiveness of the attacks according to their distance from their victim. In addition to these different mobility models, a framework for evaluating the energy consumption of a node has been developed in recent years [81] for the WiFi protocol. The power supply on each node can be represented by the class: EnergyModel and like in reality a node can possess one or more energy sources. The calculation of the energy consumption is performed by taking into account the different states of the PHY layer presented in the WIFI protocol: Idle, Tx, Rx, ChannelSwitch, Sleep, and Off. At the start of the simulation, each node is assigned an energy source with a certain number of energies and each of these states is associated with a value (in Ampere). At each transmission, the energy consumed in the previous state is calculated, and the energy source is notified in order to update the remaining energy of the node. Finally, the ns-3 community is active and develops modules to simulate recent technologies such as machine learning algorithms with ns3-ai or ns3-gym.

4.4 A new module to simulate jamming attacks

After much research to find a complete jamming attack module on ns-3, we found a beginning of code corresponding to this need [82]. However, the latter has not been updated

for over twelve years and is based on an older version of ns-3. In addition, during its installation, we had to modify certain classes belonging to other modules such as the "WiFi-Phy" or "InterferenceHelper" classes. These changes were therefore risky and time-consuming. To finish, the module proposed in this paper is based on the "WiFi-Phy" class, and in the last releases of ns-3, the main functions of this class have been modified. Consequently, the previous work with the new WIFI module is not compatible. In our new version, we update the previous work in order to improve some elements. The complete code is available in [83] and the documentation in [84].

Besides, the compatibility aspect, we have added other features. Indeed, in addition to the two metrics already available in the previous version (PDR and RSSI), we have added other essential metrics to assess jamming attacks. The first is the energy expenditure of the victim. Indeed, we have connected the energy model of ns-3 "EneryModel" with our module to be able to evaluate the energy consumption of both the attacker and the victims. Moreover, the Inter Arrival time (IAT) metric was added to this module. It corresponds to the time interval that elapses between the reception of two packets. The higher the IAT, the more likely the network is to be jammed. The number of hops has also been included as a metric, as well as the total number of packet jamming. Finally, it is now possible to calculate the bit error ratio (BER) which is equal to the number of bit errors divided by the total number of transferred bits during a time interval. One of the main contributions is the integration of tools into the module to create smart jamming attacks and smart mitigation methods. Indeed, we integrated "ns-3-gym" in order to create new Subclasses of Jammer and Mitigation. The "ns-3 gym" module created by Piotr Gawłowicz and Anatolij Zubow provides an interface between ns-3 and OpenAI-Gym. OpenAI-Gym is a toolkit for developing and comparing reinforcement learning algorithms. Consequently, the integration of this module allowed us to easily and quickly simulate jammers and mitigation methods using reinforcement learning algorithms.

4.4.1 Architecture of the new module

In this section, we present the architectural model of the new jamming module. This module is integrated between the PHY and MAC layers of the communication protocol. Therefore, no modification of the different layers is necessary for its execution. As we see in Fig 4.3, the jamming module is composed of four main components and provides a set of essential functions (called APIs) to exploit them.

The APIs allow direct control of the Jammer or Node having a Mitigation System. The components of the new jamming

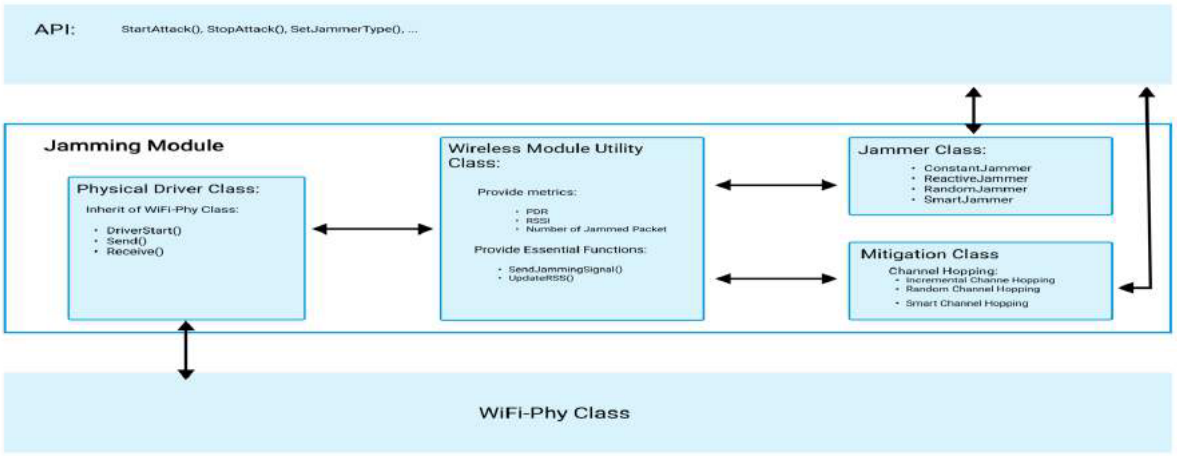


Figure 4.3: NS-3 Jamming Module Architecture

module are:

- **Physical Layer Driver Class:** This class establishes the link between the Wifi-Phy class present in the WIFI module and the Jamming module. Indeed, the behavior of the PHY and MAC layers of a jamming node is not the same as that of a conventional device. For example, the CSMA/CA process should be disabled and packet creation is simplified. Indeed, the jammer does not aim for its transmitted packets to be processed by the receiver. The only requirement is that the packet from the jammer is in the form of a WIFI packet. Therefore, the preamble or the data can be set randomly. Additionally, in order to implement the new mitigation methods for legitimate network nodes, we had to modify the physical behavior of the WIFI protocol. Indeed, the multichannel not being present in the WIFI module, we had to adjust part of the logic of the physical layer. These changes have been implemented in the Physical Layer Driver class which inherits basic functions from the WiFi-Phy class of the WiFi module. Therefore, no modification of the ns-3 WiFi module is required. This class essentially improves the portability of the module.
- **Jammer Class:** This part represents the heart of the module and includes all the functions representing the behavior of a jammer. Its implementation was designed to be as extensible as possible. Indeed, with a heritage system, it is feasible to implement your own jamming strategy by creating a new subClass based on the Jammer Class, as depicted in Fig 4.4. Therefore, four basic jamming attack subclasses are implemented in this release: Eavesdropping-Jammer Class, Constant-Jammer Class, Reactive-Jammer Class, Random-Jammer Class. Moreover, we have added another subclass, the Smart-Jammer class, allowing to implement a jammer based on reinforcement learning algorithms with the help of

the ns-3-gym module.

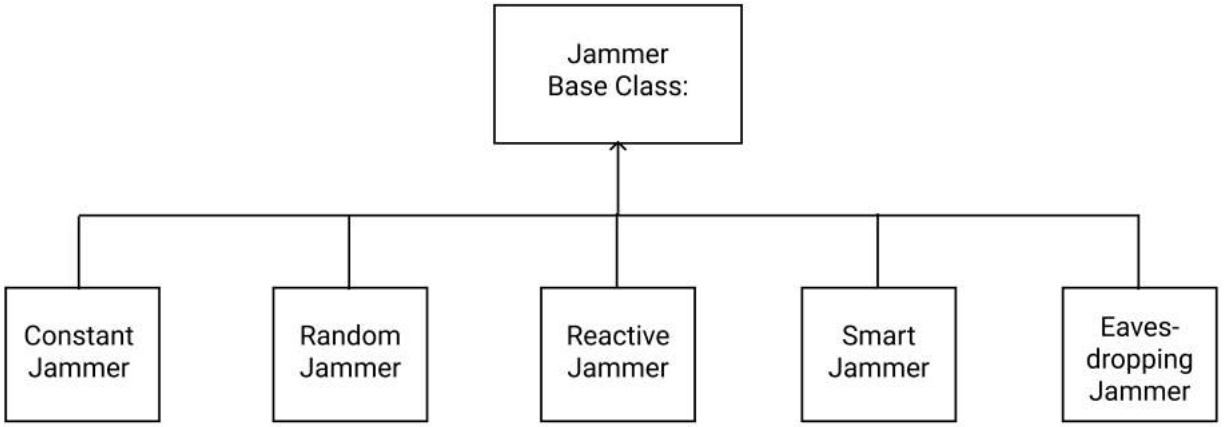


Figure 4.4: Jammer Component UML

- **Jamming Mitigation Class:** With the similar logic as the Jammer class, this component allows you to set up mitigation methods. Based on abstraction, all mitigation approaches are imaginable. In this release, a mitigation method has been implemented in a subclass: Mitigation-channel-hopping. However, as with the jammer component, several strategies have been devised. We find basic strategies such as the scheduled or random channel hopping methods, but also smart channel hopping methods.
- **Wireless Utility Class:**This class is located between the different classes of the module and ensures their connection. It is in this component that all the various metrics such as PDR or RSSI are stored.

All of these components are interdependent. It is possible to create a network with a mitigation method without an jamming node and vice versa.

4.5 Validation of the module

Before implementing the FOLPETTI framework on this new jamming module, we wanted to verify that the models implemented represent the behavior obtained with the real world. We started validating this new module with simple models before moving on to more complex models.

4.5.1 Basic method

This section provides a comparison between the results obtained with several basic jamming attacks on the simulator and the testbed. The complete description of the development of the test-bed is presented in chapter 5.

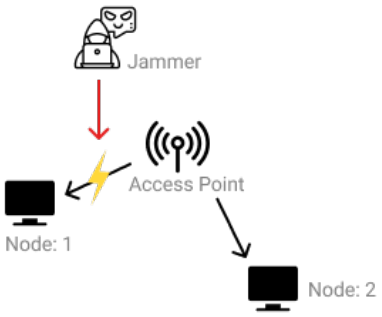


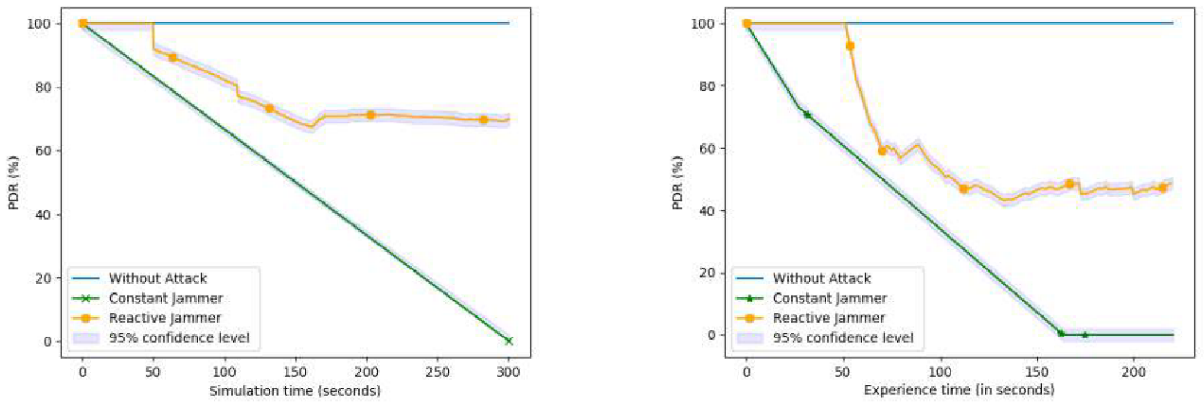
Figure 4.5: Network model for validation assessment

At first, we chose a simple configuration for the network to implement both on the simulator and the test bench. Indeed, the network is composed of four elements: a transmitter, a receiver, an access point, and an attacker. The configuration of this network is illustrated in Fig 4.5. The transmitter and the receiver are connected via the access point and use the 802.11n protocol with the Transmission Control Protocol(TCP). All simulation and experiment parameters are summarized in Table 4.2.

Table 4.2: Simulation and TestBed parameters

Parameter Name	Setting Used
Energy Model	EnergyBasicModel
Simulation Time (seconds)	300
Interval Packet (seconds)	0.2
Distance of legitimate nodes (meters)	1
Distance of attacker node (meters)	5
PDR Threshold (%)	60

Validation experiments are performed under three different conditions a) communication without attack, b) communication under constant jamming attack, and c) communication under reactive attack. In Chapter 2, we explained the different jamming strategies in detail. A constant jamming attack aims to continuously emit a signal on a channel. Conversely, a reactive attack is active only when a communication is present on a channel. The result presented just below is an average of the results obtained with 5 runs and 95% of confidence interval.



(a) (b)

Figure 4.6: PDR for different type of jamming attacks in a a) simulation environment and b) real-environment

Fig 4.6 demonstrates the behavior of the constant and reactive jamming attacks in terms of Packet Delivery Ratio (PDR) in both environments. In this case, the closer the PDR is to zero, the more effect the attack has. For the constant attack, in the simulator environment, we can see in Fig 4.6a that the PDR constantly decreases after the start of the attack. The same behavior is observed in the real environment in Fig 4.6b. We also observe that the constant attack regularly degrades the PDR and at the end of the simulation the PDR is equal to 0%. In the simulated environment, the reactive jammer has a strong effect at the start of the attack. However after 150 seconds of the simulation, the PDR varies between 60% and 70%. In the real case, after few seconds of the experimentation the PDR significantly drops. However, the latter increases again to stabilize around 50-60%. The behavior is more or less the same for the reactive attack in the both environments.

It is important to note that for the reactive attack to be successful, the reaction and attack time must be less than the legitimate transmission time. However, in these two scenarios, the packet size varies randomly and the transmission time depends on the packet size. Therefore, the PDR of the reactive attack converges to a threshold. In addition, not having an anechoic chamber it is possible that the PDR in the real environment is also impacted by the interference generated by nearby networks. This is how we partly explain the difference between the results obtained with the real and simulated environment.

4.5.2 Validation with the smart method

After validating this new module with two classic jamming attacks, we tried to validate this module with a more elaborate process. Indeed, our second analysis is carried out on the creation of smart mitigation method: channel hopping. Channel hopping represents a real dilemma for the victim during a jamming attack. Indeed, a legitimate node has the choice of hopping onto a channel whose performance is known and therefore satisfying a certain quality or switching to an unknown channel without knowing its performance but which could possibly have a higher performance. To evaluate our module, we implement the same algorithms presented in [65] on the simulator side and on the experiment side. In this paper, the authors implement a channel hopping strategy based on Multi Armed Bandit algorithm with a Thompson Sampling policy. The goal of this algorithm is to converge to the best possible choice in order to maximize the sum of the rewards. In this case, the sender receives a positive reward when the channel is available and the communication completes without a hitch. The authors employed a Thompson Sampling policy and prove that their method converges faster to the best channel than the existing algorithms and achieves higher average throughput. The main objective of

this method is to predict the optimal future communication channel, i.e. the one with the least possible interference.

We have adapted this method to the 802.11 protocol with a number of channels equal to 12. In this situation, the access point employed the smart mitigation method and based its reward on the PDR metrics. Indeed, if the PDR is below a certain Threshold, the access point can deduce that a problem occurs in the channel. In the same logic, if the access point loses the connection with a certain number of nodes, it can deduce that a possible attack is taking place in the network. Therefore, we based the reward on these two metrics and we set the PDR threshold to 60%. Thereby, the reward is negative if the PDR is less than 60% or if the access point loses connection with at least one node in the network. The jammer also has the ability to change the transmission channel with a frequency hopping method. Indeed, at regular time intervals, the jammer jumps to the next communication channel. In this scenario we fixed the interval hopping for the attacker at 1 seconds. Therefore, every 1 second a new channel is jammed and its performance is drastically reduced.

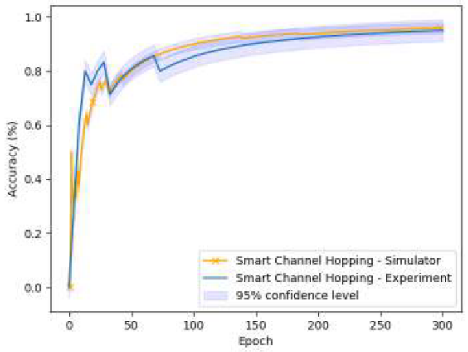


Figure 4.7: Accuracy of Smart Channel Hopping Model with 12 accessible channels

Fig 4.7, demonstrates the accuracy of the smart channel methods according to the number of epoch for the experiment and simulation time. Accuracy is calculated as the ratio of the sum of correct rewards to the total number of rewards. In this case, a correct reward corresponds to the choice of the available channel with the least possible interference. We observe that the Multi-Armed Bandit algorithm in both environments converges to 80% after 70 epochs and achieves 93% after 300 epochs. We observe the same behavior of smart attenuation channel hopping when simulated or experimented.

4.6 Assessment of the FOLPETTI framework in two scenario

After validating our module with several examples, we implemented FOLPETTI on it in order to understand its effec-

tiveness. We combined FOLPETTI framework described in chapter3 with a jamming attack. In this section, we describe in first the network model implemeted to test this framework on the simulator. Then, after briefly describing the behavior of the FOLPETTI attack, we compare the performance of this framework with those of other relevant attacks in two different scenario.

4.6.1 Network model for the assessment

We consider a wireless communication network composed by three legitimate nodes connected via an access point with the IEEE 802.11 protocol and capable of transmitting on 12 different channels. We assume that the attacker has the same configuration as the legitimate nodes to reduce the probability of being detected. The attacker jams the communication between the access point and node 1 as depicted in Fig. 4.8.

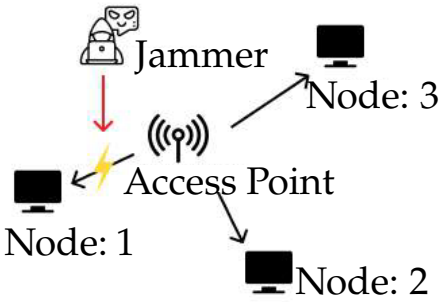


Figure 4.8: Network model for the FOLPETTI assessment

We assume that legitimate nodes in the network may be able to detect an attack if their performance fall behind a certain value. Therefore, a stealthy attack is possible only if the attacker is able to keep its success rate below the identifiability threshold. To follow the effects of the jamming attacks and compute the PDR, the three considered devices communicate thanks to the Transmission Control Protocol (TCP). Indeed, TCP provides an acknowledgement (ACK packet) for each correctly received packet. Therefore, based on these ACK packets, the transmitter can judge whether the transmission is successful and consequently update the PDR metric. We assume that the access point constantly transmits packets every 0.1 s and begins its transmission at the start of the simulation (t = 0). After 10 s, the attacker starts its attack. The legitimate nodes and the attacker start their communication on the same channel. Table 1 summarizes the simulation parameters.

Victim channel hopping:

Legitimate network nodes have the ability to respond to a jamming attack with a channel hopping mitigation method. Two methods of channel hopping have been implemented in this new simulator module. We describe each of these strategies below:

- **Random Channel Hopping**($T_{xRandom}$): This strategy is very simple and consists to randomly hop into a new channel when a detection takes place.To guarantee the quality of service, after the selection of the new channel, the nodes proceed to a verification of the availability of the channel. Therefore, if the new channel has a low signal quality, it has the possibility to re-select a new one. The transmitter has a probability $M_{available}/M$ of selecting a free channel, where $M_{available}$ represents the

Table 4.3: Simulation parameters

Parameter Name	Setting Used
Simulation Time	1790 seconds
Size of Legitimate Packet(octets)	1000
Start of jamming(seconds)	10
Start of channel hopping(seconds)	1
Energy Model	EnergyBasicModel
Distance Node - Access Point(m)	5
Threshold Detection (%)	80

number of channel available at time instant t and M the total of channel.

- **MAB Channel Hopping:(T_{xSmart})** This is the method already described and evaluated in the previous section 4.5.2. The main objective of this method is to predict the future communication channel with the least possible interference.

Moreover, we have compared this new type of attack against four kinds of attack already presented in the literature. Each attack differs in its strategy, which we outline below:

- **Random Channel Hopping**(A_{Random}): At each iteration, the jammer randomly selects a new channel to attack. Consequently, the jammer has a probability to attack the "good" channel, i.e the channel where the transission takes place, $1/M$ where M represents the total of channel used by the victim.
- **Reactive Channel Hopping**($A_{Reactive}$): At each iteration, the jammer listens and records the occupation of each channel used by the transmitter one by one. If during its listening, it locates a communication corresponding to the victim, it reacts and jam the packet. Consequently, the reaction time must be shorter than the packet transmission time for the latter to produce an effect.
- **DRL based Channel Hopping**(A_{DRL}): The attacker employs the strategy developed in [31] and uses a DRL algorithm consisting of an actor and a critic to make decisions on the channel to attack. In this method, the actor observes its environment to choose the action to optimize his policy. During this time, the critic evaluates the actions performed by the actor by calculating the difference between the expected outcome and the actual one and informs it of the quality of its choice.

The temporal difference value is then used to update the actor and critic model. Moreover, to optimize this model and reduce the probability of being detected, the attacker alternates between two modes: attacking phase, and listening mode. Indeed, the two agents (actor and critic) are based on neural networks that must be trained beforehand. This is why, during the training phase, the attacker is in listening mode. Once the model is trained, the attacker switches to attack mode. In this phase, the attacker jams the channel and can decide to switch to listening mode to re-train its neural network when performance decrease.

- **Optimal based Channel Hopping**($A_{Optimal}$): It corresponds to a genie attack. This approach represents the optimal solution, and we considered it as baseline in our simulations. We assume that the attacker is omniscient and know in advance the pattern of the channel hopping employed by the victim.

4.6.2 Performance of FOLPETTI-Based Jamming

As we saw in chapter 3, FOLPETTI is a framework that we can couple with several type of attack. In this simulation we coupled this framework with a jamming attack. In this case, the action is to jam the selected channel corresponding to the output of the Multi-Armed Bandit algorithm. If the attack is a success, the reward obtained is equals to 1, 0 overthise. To obtain if the attack is an success, we base on the Received Signal Strength Indicator (RSSI) metric. Indeed, we observe a variation of this metric when an attack occurs in the channel. Indeed, we have observed a drop in the RSSI when an attack takes place whether on the side of the receiver or the attacker. Fig. 4.9 shows the variation of the RSSI of the receiver (victim node) during a jamming attack. The evaluation lasts for 100 seconds. The attack begins at second 30, and lasts for 10 seconds.

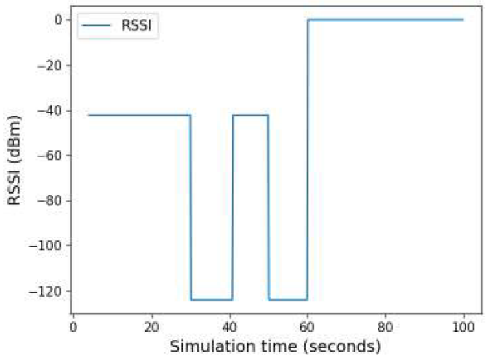


Figure 4.9: RSSI metric of the Receiver during jamming attack

Based on Fig. 4.9, we observe that, when the jammer is active the RSSI of the receiver is lower (−120 dBm) compared to

when the jammer is inactive (−40 dBm). Moreover, after the first 60 seconds, the connection between the legitimate nodes of the network is lost. This is confirmed by the 0 valued RSSI, that denotes the absence of signal. Consequently, we can deduce that the RSSI of the jammer and the receiver can be considered as a metric to estimate the success of an attack. Indeed, the jammer is effective when its RSSI is lower than a predefined threshold. This metric, unlike other metrics such as the Packet Error Rate (PER) or PDR, does not require the attacker to spend a lot of time in listening mode. Therefore, the attacker can remain active for the whole duration of the attack. Finally, to better understand the advantage of FOLPETTI, we initially describe its behavior and compare it with that of a DRL-based attack. In this part, we assume that the victim's channel hopping strategy follows a random choice. The Fig 4.10 represents the effectiveness of these two type of attack in therms of PDR.

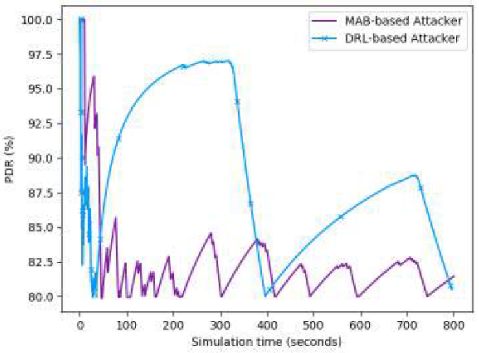


Figure 4.10: PDR of the attacker.

The closer the PDR is to detection threshold, the more effect the attack has on the network. We fix the attack detection threshold to 80% PDR. Indeed, based on [35], and the distance between the access point and the victim that we defined above, the percentage of the PDR observed in real-life in a network without attack varies between 82%- 100% in a normal behavior. Consequently, when the PDR drops at 80% a jamming attack occurs and the transmitter randomly hops into another channel. As we can see in the figure, the victim's PDR when undergoing a Folpetti attack remains around 80-85%. On the contrary, with the DRL attack, the PDR tends to vary much more from 80% to 97%.

Fig. 4.11 shows the transition channel pattern for the transmitter and the attacker for the case of FOLPETTI attack and DRL-based attack. For reasons of readability, we report the attacker's channel only when it is simultaneously used by the transmitter.

With the combined Fig 4.10 and Fig 4.11, we observe that the FOLPETTI tracks the channel hopping mitigation method. Indeed, at $t = 10$ the PDR drops from 100% to 89%. This is explained by the fact that the attacker and the transmitter are

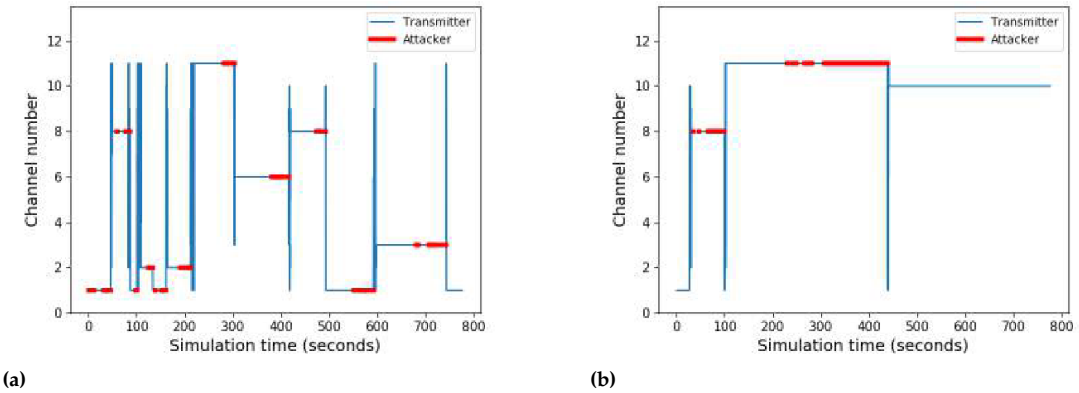


Figure 4.11: Transition channel pattern for transmitter and attacker a) Channel Behavior with FOLPETTI attack, b)Channel Behavior with DRL-based attack

positioned in the same channel. Then the attacker is in the exploration step and examines the availability of the other channels, hence increasing the PDR again to 95%. At the end of this phase, the attacker decides to turn to the exploitation period and jams channel 1 causing the PDR to drop to 80%. At this point, the transmitter detects a potential attack and changes the communication medium. The victim randomly chooses a channel and verifies in a second time if this is occupied. Therefore at time $t = 49$, the transmitter switches to channel 11. However, as this has a low RSSI value, the access point will re-select a new channel (in this case, 8). Simultaneously, the attacker receives negative rewards and decides to change the jamming channel. Consequently, at $t = 54$ the attacker chooses to jam channel 8 which will again cause the PDR to drop to 80%. We observe that this behavior pattern remains until the end of the simulation.

On the contrary, for the DRL-based attack, we notice in Fig. 4.10 that at the beginning of the simulation, the PDR rapidly drops to 80%. Consequently, the legitimate nodes of the networks react and change their transmission channel from 8 to 11, as we see in Fig. 4.11b. This is explain by the fact that the DRL attack was already train before the begining of the attack. Consequently, the attacker already knows the pattern and the optimal channel to jam. However, the attacker continues to jam channel 8 for a short time until it finds that it has no effect on it. As a result, it switches to listening mode to re-train its neural network with new observations. This period appears on the figure starting from second 150 and lasts about 100 s. During this time, the attacker does not impact the network, and the values of the PDR increases. At second 300, the attacker finds its new policy and jams the channel where the transmission is taking place. Contrary to Folpetti attack, the DRL attack has a long learning phase, where the attacker is inactive. Therefore, during this phase, the attack has no effect and the PDR increases.

By analyzing the behavior of the two attacks, we see that the attack based on the MAB algorithm, i.e., FOLPETTI, reacts faster to policy updates than the one based on the DRL. After this brief analysis, we evaluate the performance of FOLPETTI attack in two scenarios in the next section.

4.6.3 Scenario 1: FOLPETTI Against Random Channel Hopping

Our first analysis was performed against a basic strategy, a channel hopping method. For this scenario and the one described in the next section, we evaluate four metrics a) PDR, b) success rate of the attack, c) number of retransmissions and d) number of detections. The success rate of the attack corresponds to the number of successfully jammed packets by the attacker over the total number of transmitted packets by the victim. In order to obtain these results, 1000 simulations of each attack were carried out.

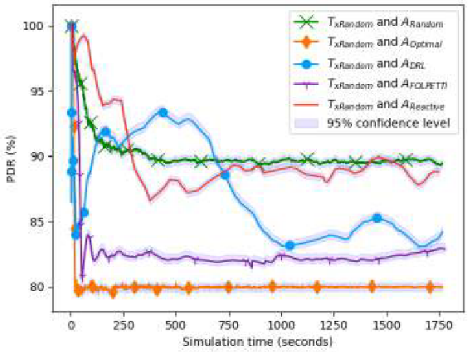


Figure 4.12: PDR for different strategies of attack against Random Channel Hopping Method

The Fig 4.12 demonstrates the behavior of the PDR under each type of attack. We observe that the random and reactive attacks have limited effects. Indeed, we observe that the PDR is around 90% throughout the simulation. The DRL attack is effective at the beginning of the simulation. As we are already explained, the attacker is pre-trained and know the optimal channel to jam. However after a few seconds, the victim changes its channel transmission and the attacker must re-train its strategies. Consequently, the PDR takes more time to the FOLPETTI attack to converge on the right strategy. Indeed, the results obtained with the FOLPETTI attack approach those obtained with the optimal attack. Indeed, as said before it is an ideal approach, where we assume that the attacker knows in advance the right channel to attack. Hovewer for the FOLPETTI attack the PDR is close to 82%, i.e 2% more than for the optimal attack.

These results are also confirmed with the success rate reported in Fig 4.13. Indeed, we notice that the sucess rate for the FOLPETTI attack is close to the optimal attack and twice

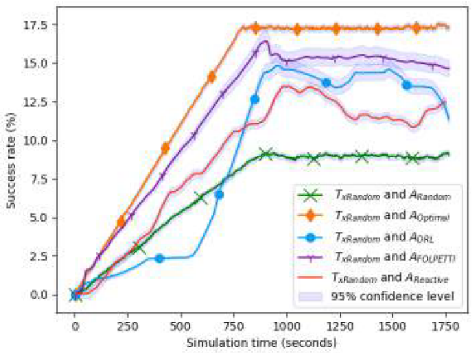


Figure 4.13: Success rate for different strategies of attack against Random Channel Hopping Method

more effective that the randon attack. It is important to note that the success rate for the reactive attack is variable and does not remain constant over time. Indeed, the success of this attack essentially depends to the react time of the attack. Indeed for the attack be a success, the attacker must satisfy this equation:

$$listen_{time} + channelSwitch_{time} + attack_{time} < transmission_{time} \tag{4.1}$$

The transmission time essentially depends to the size of the packet and the distance between the transmitter and the receiver. In these simulations, the two nodes are fixed and remain at the same distance. However with the WIFI protocol, different size of packets are transmitted according to the type of packet. For example, an ACK packet has a approximately defined size at 8 bytes and the size of a beacon packet can vary from 60 bytes to even 450 bytes. Moreover, the size of packet depends on the data transmitted. Consequently, we randomly varied the size of the packet. This is why, in some case, the reactive attack is more effective and have better performance.

By comparing the number of retransmitted packets in Table 4.4, we can discern that the strategy based on MAB with Thompson policy has an impact on the behavior of the network. On the contrary, the random channel hopping strategy has no effect on the number of retransmissions. Indeed, the number of retransmissions is equal to 74 against 738 with the strategy based on the MAB and 1070 for the optimal solution. Moreover, the number of retransmissions for the DRL-based solution is 641, 13% less than for the FOLPETTI attack. The number of detection and consequently the number of channel hops at the transmitter side confirm these results. For the random solution, the PDR never drops below 80%, therefore no attack is detected and no channel hopping is performed. For FOLPETTI, the transmitter detects an attack 1483 times against 1729 times for the optimal solution. Therefore, this new type of approach produces a significant

Table 4.4: Number of retransmissions and detection for different attack strategies against random channel hopping method

Attack	Number of Retransmission	Number of Detection
Random	74	0
Reactive	419	514
DRL-based Attack	641	1189
FOLPETTI	738	1483
Optimal	1070	1729

effect in terms of retransmissions and disturbance on the channel, hence decreasing the energy efficiency of the victim network. As confirmed by the results obtained previously, the attack based on DRL is less efficient than the optimal and FOLPETTI attacks and therefore leads to a lower number of detection. Indeed, this approach has a detection number of 1189, 294 detections less than with our approach.

Consequently, when the mitigation method is based on a random strategy, the proposed solution is effective and the results obtained are almost similar to those provided by an optimal solution.

4.6.4 Scenario 2: FOLPETTI against Smart Channel Hopping

We also evaluated the FOLPETTI attack against a more elaborated channel hopping method: a channel hopping based on MAB as already mentioned in previous section. Same evaluations have been conducted in this scenario as in the previous case.

As confirmed in the Fig 4.14, the FOLPETTI attack is more effective than other type of attack even if the channel hopping method is elaborated. Even if smart channel hopping has better performance than the random channel hopping, the FOLPETTI attack cause damages. Indeed, the PDR for the reactive and random attacks is around 98% contrary to 92.5% percent for FOLPETTI attack. The DRL attack takes longer to converge as we have already explained and reaches at the end of the simulation a PDR rate of 95%.

The performance of the different attack strategies evaluated are also confirmed with the success rate presented in Fig. 4.15. FOLPETTI attack has better performance than other types of attack and particularly 7.5% more effective than based on a random strategy. Indeed, the success rate for the FOLPETTI

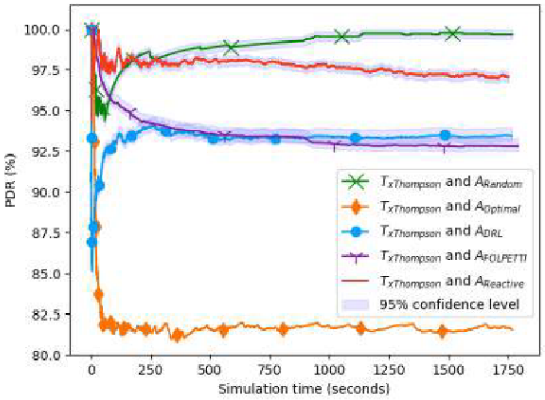


Figure 4.14: PDR for different strategies of attack against Smart Channel Hopping Method

attack at the end of the simulation is equals to 7.5% and random attack 0% and DRL 2%.

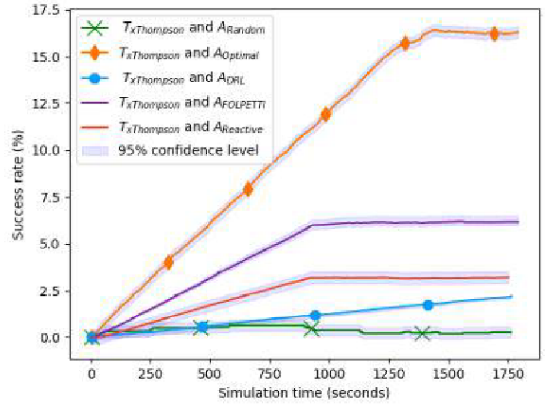


Figure 4.15: Success rate for different strategies of attack against Smart Channel Hopping Method

Finally, this conclusion is also confirmed with the number of retransmissions reported in Table 4.5. The attack based on MAB algorithm has an important effect on the network as it allows to generate 280 retransmissions.

Table 4.5: Number of retransmission and detection for different attack strategies against smart Channel Hopping method

Attack	Number of Retransmission	Number of detection
Random	38	0
Reactive	87	124
DRL-based Attack	212	485
FOLPETTI	280	513
Optimal	470	751

4.7 Conclusion

In this chapter, we introduced a new jamming module for the ns-3 simulator. We have set up a system that is not only extensible but also includes new methods of mitigation and jammer strategies. Indeed, it is now possible to create smarter attacks based on more advanced algorithms such as reinforcement learning. We prove its scalability by developing an intelligent channel hopping method and jamming attacks based on existing works. In addition, we have validated the behavior of this module including the results of different types of attacks obtained with it and real experiments. As with any tool under development, there are still improvements to be made to this module. For example, in the future we will want to extend this module with additional functionality such as another mitigation method. Several more advanced detection algorithms can also be added later, such as a detection algorithm based on Machine Learning for example.

However, this new tool allowed us to confirm the FOLPETTI framework can be coupled to the jamming attack and that no prior knowledge is required for the attacker. These results show FOLPETTI has the ability to understand and predict the behaviour of a victim and more particularly their channel hopping strategy. In this simulator, we evaluated FOLPETTI against two defence strategies, one more classic based on random channel hopping and one more advanced based upon MAB algorithm. In the first situation, FOLPETTI approaches an optimal attack solution in terms of PDR, success rate, and number of retransmissions. In the second case, our new attack is still able to enhance the success rate to 5% against 2.5% for the other smart attack. However, these first analyzes were carried out in an ideal situation. Indeed, for the moment a jamming attack carried out on a specific channel only affects the latter in the simulator. This behavior does not entirely

reflect reality because sometimes a jamming attack on one channel causes effects on neighboring channels, that is not accounted in simulator. That's why, now that we saw that the behavior of the attack could work thanks to the simulator, we wanted to investigate it in reality as explained in the next chapter.

Evaluation of HARPAGON and FOLPETTI frameworks in real environments

5

As mentioned in [85], the simulation on the one hand, and experimentation on the other hand should not be considered competing but complementary activities. Indeed, as already said in the previous chapter, simulators are an ideal tool for predicting and quickly estimating the effectiveness of models. They offer a high degree of flexibility and permit the isolation of particular physical effects. However, it can be very complicate to account for all the effects occurring in a real environment in a simulator, and sometimes the results obtained are not representative of reality. This is why, in the previous chapter, we simulate the FOLPETTI framework in order to have a potential representation of the results that we will have in real life. This allowed us to validate this new type of attack in a theoretical way. In this chapter, we set up a testbed to demonstrate the effectiveness of these two frameworks created during this thesis HARPAGON and FOLPETTI attacks. After fully describing the testbed, we detail the effectiveness of HARPAGON and FOLPETTI attacks in terms of PDR and the loss of energy it causes for the victims when coupled with jamming attacks..

5.1 Description of the implementation of the testbed

In this section, we describe the implementation of the testbed for the two parts: the legitimate network and the attacker.

5.1.1 Implementation of the legitimate network

In order to reduce the probability of side effects during our measurements such as various behaviors generated by two amendments to the communication protocol, we have chosen to work with homogeneous nodes. Consequently, the legitimate network is composed of three same classic and identical laptops as shown in Fig 5.1, using the same type of network adapter: a Qualcomm Atheros AR9485 Wireless Network Adapter. In order to stay on the same type of network as that described in the previous chapter, the receiver and the transmitter communicate via a WiFi network (802.11) in infrastructure mode on the 2.4Ghz Band. Moreover, to simplify the computation of the Packet Delivery Ratio, the Transmission Control Protocol (TCP) is employed. Indeed as demonstrated in Fig 5.2, with this type of protocol each data sent has an associated sequence number named sequence number field. At

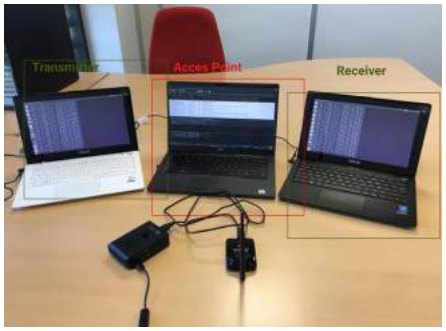


Figure 5.1: Test-bed composed of three legitimate nodes: one transmitter, one receiver and one access point

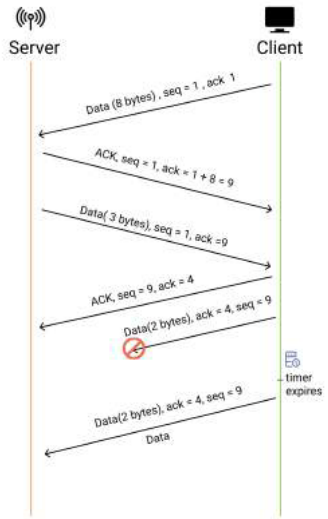


Figure 5.2: Illustration of TCP retransmission after a packet loss

Parameter Name	**Parameter Used**
Size of transmitted packets	Between 50 and 1400 bytes
Number of retransmission allowed	1
Waiting time for acknowledgment	1 second

Table 5.1: Parameters of legitimate nodes

each reception, the receiver transmits an acknowledgment of receipt (ACK frame) allowing the transmitter to be informed that the received frame is readable.

In the case of a corrupt or never-received frame, the receiver will not send an acknowledgment. During this time, the transmitter waits during a determined time the ACK frame. If the emitter, does not receive an acknowledgment before the timer expires, the sender will assume the segment has been lost and will re-transmit it. Therefore, we determine the PDF metric on the sender side through the ACK packets. If the ACK is received, the last emitted packet will be marked as a transmission success.

In this testbed, to generate regular traffic, we used the SCAPY tool with a PYTHON script on the receiver and transmitter side [86]. SCAPY is a powerful interactive packet manipulation program that makes it easy to create or break up packets of many communications protocols. It is thus possible to configure a large number of parameters such as the number of possible retransmissions and the waiting time for an ACK. We defined the number of retransmission at 1 and the waiting time for each acknowledgement a 1 second. All the different parameters are summarized in Table 5.1. Moreover, the transmitter scans the channel before each transmission to detect the status of the latter. Indeed, if the channel is perceived as busy, it reports the sending of packets. Otherwise, if the channel is detected as unoccupied, data will be transmitted. Moreover, as we already mentioned in chapter 4 with the equation 4.1, the success of reactive attacks depends essentially on the size of the jammed packet. Therefore, to verify this assumption and ensure that our testbed is closest to reality, we have considered different sizes of packets emitted by the transmitting node.

The access point possesses a mitigation approach against jamming attacks, i.e., a channel hopping method. Indeed, we modify the *hostapd* open source code on the AP side to encode several channel hopping methods not available in the basic protocol [87]. Hostapd (host access point daemon) is a user space daemon software enabling a network interface card to act as an access point and authentication server. In the 802.11 protocol, the legitimate nodes have a predefined set of M channels to use, which can be identified via integer numbers. This number is variable according to the country. For example, in France, the 802.11 protocol includes 13 available channels. Three different channel hopping implementations: incremental channel hopping, random channel hopping, and a smart channel hopping have been developed with the *hostapd* tool:

- **Incremental Channel Hopping** ($T_{Incremental}$): At each defined time, the legitimate nodes on the network increment their channel number by 1. Therefore, the legitimate network nodes follow a well-defined pattern.

- **Random Channel Hopping** (T_{Random}): In order to reduce the chances of inferring the predefined channel hopping pattern, the second method we implemented is a randomness-based method. The master node randomly chooses a new channel among the M available and informs the other nodes.
- **Smart Channel Hopping** (T_{Smart}) The method follows the model proposed by the authors in [65] and employs a MAB approach. The objective is to converge to the best channel available in as few steps as possible by employing a Thompson sampling formulation.

Finally, an interface was created for each node of this network in order to facilitate the configuration of the parameters during each experiment such as the experiment time. To create this desktop application, we used the well-known Electron framework to enable the creation of native applications with web technologies such as JavaScript, HTML and CSS [88]. The react library has been used to write the Javascript code of the different forms more quickly and to allow the call of python scripts. An example of this interface can be seen in Fig 5.3.

To conclude with this test bench, as we have control over the traffic we have the possibility of calculating, on the transmitter side, the PDR, the number of packets sent, the number of retransmissions and the Received Signal Strength (RSS). In the same way, for the receiver side, we are able to compute the Inter Arrival Time (IAT) of each packet, the number of received packets, and also the Received Signal Strength. With the access point, it is possible to keep an history of the different used channels and the accuracy of the channel hopping method based on Machine Learning.

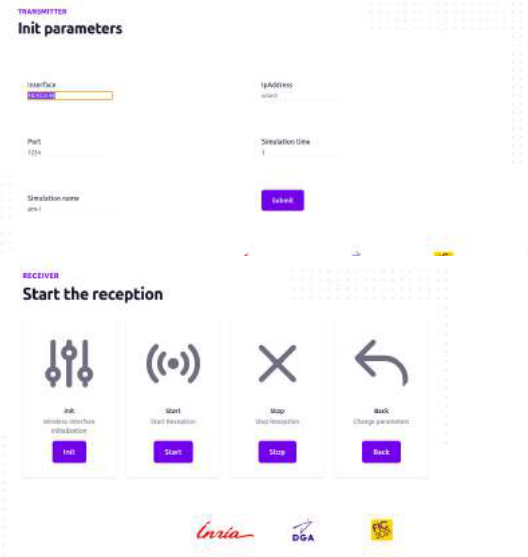


Figure 5.3: Examples of interfaces for a) transmitter, b) receiver

5.1.2 Description of the attacker's implementation

In order to possess a mobile attacker, we implemented our attacker code in a Raspberry-Pi equipped with an Alfa AWUS036h and Realtek RTL8187L device including the wireless chip ath9k. We voluntarily chose this equipment because the driver and firmware are open-sources. We have modified the driver of the wireless chip to get direct control over MAC layer parameters, following the work of [89]. In addition, with this device, we can easily have our hands on the four operating modes of the wireless card, that is, the modes of reception, transmission, idle and sleep. Each state of the attacker wireless card has a different energy consumption which can be found in the documentation of the wireless chip in [90], and reported in Table 5.2. In addition, it is also possible to influence the channel jump behavior in the but to implement our own strategies. The Carrier-sense multiple access with collision avoidance (CSMA/CA) mechanism to

this device has been disabled to be able to send packets without conditions. The size of the packet allowing to scramble the communications has been defined at 50 bytes.

Constant and reactive jamming attacks, two old jamming attacks have been implemented in the driver to allow comparison of our results between the latter and our new frameworks. Also, since we have to listen to the communication during the reactive attack, based on this logic, another type of attack: a passive eavesdropping attack has been implemented. Finally, as for the legitimate network nodes, an interface was created, as shown in Fig 5.4. Thanks to this one it is possible to specify which type of jamming attack we wish to perform as well as its duration.

Operating Mode	P(W)
Sleep	0.001
Idle	0.30
Tx	0.67
Rx	0.34

Table 5.2: Power Consumption for 2.4 GHz Operation

Figure 5.4: Example attacker interface

5.2 Results obtained with HARPAGON framework

In this section, we tested our HARPAGON framework in the developed testbed. With the help of the framework, we initially perform a jamming attack to disrupts the behavior of the network. Furthermore, as we have already developed the eavesdropping attack, we evaluated also the performance of HARPAGON coupled with this type of passive attack.

5.2.1 HARPAGON framework coupled with jamming attack

We evaluate the performance of HARPAGON framework coupled with a jamming attack, with the theoretical values obtained in the section 3.1.2 on the chapter 3. Therefore, F(t), representing the success probability of the attack is fixed to 0.7 and the value of the maximum energy cost is equal to 0.35. As a reminder, the different values obtained for each state were:

$$\begin{cases} P_{RX} = 0.45, P_{TX} = 0.29, P_{IDLE} = 0.01, P_{SLEEP} = 0.24 \\ \text{if (F(t)=0.7\%)} \end{cases}$$

In the experiments below, the duration of each experience equals 4 minutes. After 30 seconds of experimentation, the attacker begins to operate for a duration of 2 minutes. Several metrics that were used to evaluate the different attacks:

Packet Delivery Ratio and Detection Time:

One of the first metrics we used to estimate the effectiveness of the attack is the Packet Delivery Ratio (PDR). During our

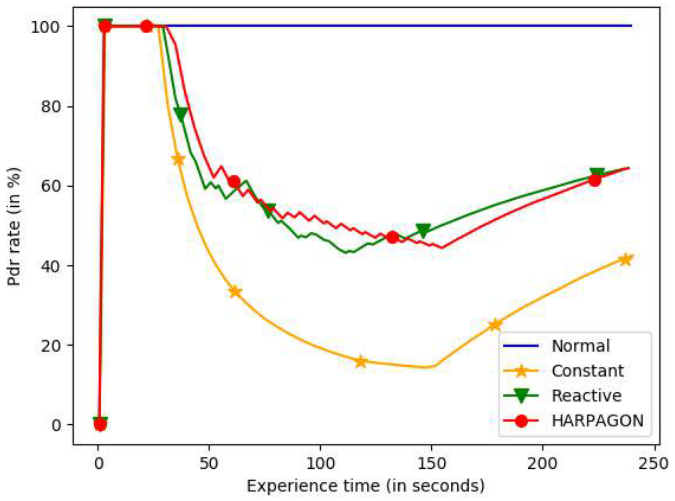


Figure 5.5: Packet Delivery ratio for each type of attack

experiment, the global PDR of the network is refreshed after each transmission. Based on this metric, statistical detection was implemented. This solution is based on the behaviour of the network without attack and has been fully explained in [91]. Indeed, on the basis of a network, we can determine the average of the PDR and hence define a detection threshold. Consequently, an attack is identified, if the PDR decreases below the detection threshold. We defined a detection threshold at 70%, and in order to avoid false positive we set-up a number of observations at 5.

Fig. 5.5 represents the PDR measurements for each type of offensive. As we can discern, the PDR for a constant attack decreases significantly from the start of the attack. This is justified to the fact that the sender and the receiver are disconnected with the access point a few seconds after the start of the attack. Indeed, the constant attack occupies the entire communication channel, consequently the access point is no longer able to send management frames (beacon) to ensure synchronisation with these members.

In addition, if we contrast the measurement of the PDR of the HARPAGON attack to that of the reactive attack, we can notice that the latter decreases gradually. Therefore, if we compare the detection-time of each attack, summarised in table 5.3, the HARPAGON attack is less detectable than the other type of jamming attack. This type of attack is detected 6.36 seconds later compared to a constant attack and 5.67 seconds compared to a reactive attack.

Type of Attack	**Detection Time (in seconds)**
Constant	18.80s
Reactive	19.49s
HARPAGON with jamming	25.16s

Table 5.3: Table of detection time for each attack

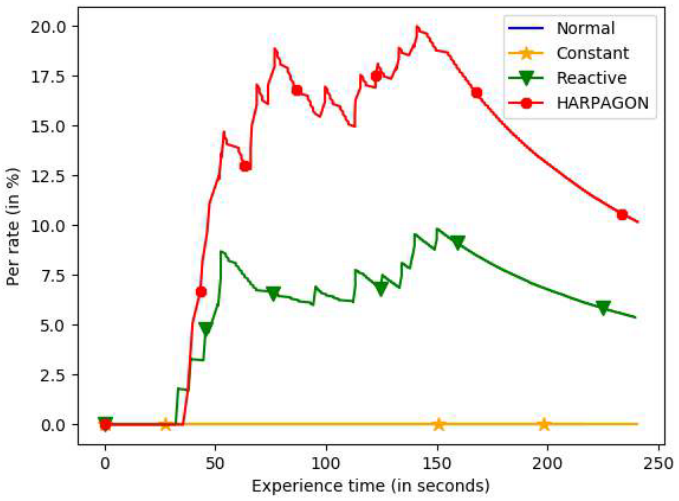


Figure 5.6: Packet Error ratio for each type of attack

Packet Error Rate and Number of Re-transmission:

In order to assess the number of corrupted data packets, we also calculated the packet error rate (PER) on the receiver side. The PER metric corresponds to the number of packets received with error divided by the total number of packets received. The Fig 5.6 presents the PER measurements for each type of attack. One of the first observations we can perform is that the HARPAGON attack produces a more important error rate than the other type of attacks. Indeed at the end of the attack, the PER for the attack based on our framework is around 20% against 9.5% for the reactive attack and 0% for the constant attack.

This can be explained by the fact that reactive attack is more likely to occupy the channel than to corrupt a packet when the latter is small. Indeed, as formulated in 4.1, if the transmission time is inferior to the reaction time added to the jamming time, the reactive attack will not be able to corrupt the packet. Therefore, the jamming signal will be transmitted but will result in partial occupation of the channel.

The receipt of corrupted packets results in re-transmission in most cases. This is why, we also report the number of each re-transmission for each type of attack in the Table 5.4. We can observe that the HARPAGON attack leads to a higher number of re-transmissions than the other two types of attack. Indeed, for a network without a detection system, the number of re-transmissions for the attack based on the framework is 48 against 29 for reactive and 0 for constant attack. In addition, if we base our analysis on a network including a detection method described just above, the number of re-transmissions for the attack created is greater before detection than the other type of attack. Indeed, the number of re-transmissions

before detection for the HARPAGON attack is 11 i.e 1.8 times more than for the reactive attack.

Table 5.4: Table of number of re-transmission for each type of attack

Type of Attack	**Number of re-transmission without detection system**	**Number of re-transmission with detection system**
Constant	0	0
Reactive	26	6
HARPAGON	48	11

Received Signal Strength:

The received signal strength is also a well-known metric for detecting jamming attack. However, as the authors of [92] have already mentioned depending on the type of jamming, RSS metric cannot be used to prove that an attack has taken place. We have evaluated this metric, shown in the Fig 5.7 and we can observe that the RSS for the HARPAGON attack fluctuates less than the other attacks and approaches the normal behaviour of the network. Therefore for this type of attack, based on the RSS metric, detection will not be feasible.

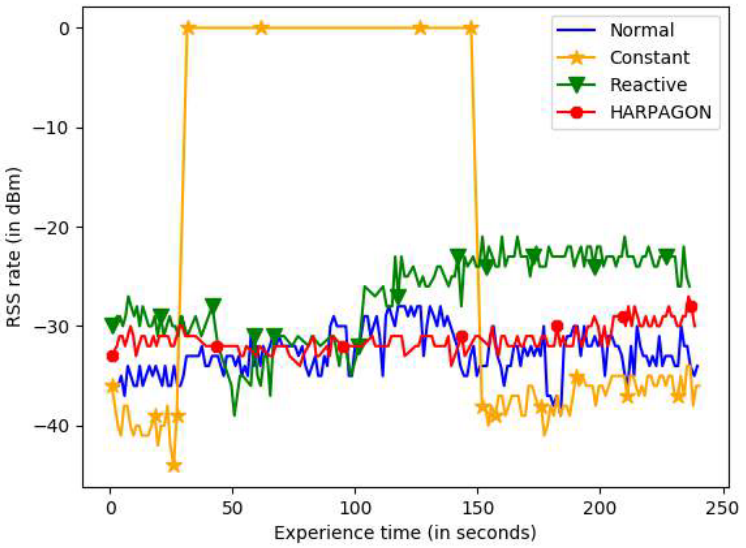


Figure 5.7: Received Signal Strength for each type of attack

Attacker Energy Efficiency:

One of the first goals of the model is to reduce the power consumption of the attacker by switching the different states of the attacker. To evaluate the energy consumption we used the formula 3.13 in Chapter 4. The results are given in Table 5.5. The HARPAGON attack spends less energy than the other two attacks. In fact, this new type of attack consumes 12.85 J less than the reactive and 41.4J less than the constant. Including the two states idle and sleep in addition to those already used in the reactive attack allows to consume less energy. In [93], authors develops a new metrics: Attacker Energy efficiency (AEE) to measure the performance of a "green" attack. This metrics is defined as a ratio of the impact of the network to the total power consumption of the attacker. In the case of jamming attack as demonstrated above, the impact of the network can be measured by packet error rate. Consequently, the HARPAGON attack has an AEE equal to 51 and the reactive attack to 18. In terms of AEE the HARPAGON attack is twice as effective.

Table 5.5: Table of energy consumption for each type of attack

Type of Attack	Tx	Rx	Idle	Sleep	Total
Constant	80.4J	0	0	0	80.4J
Reactive	22.4383J	29.4134J	0	0	51,8517 J
HARPAGON	16.2J	23.31J	0.003J	0.288J	39J

5.2.2 HARPAGON framework coupled with eavesdropping attack

We have demonstrated the performance of the framework used during an active attack. In this section we utilize it to improve a passive attack: an eavesdropping attack. The goal for the attacker is to record the maximum of packets without intervening on the network. Therefore in this case, the attacker must be in R_x mode while the transmitter is in T_x mode. Based on the HARPAGON framework, the attacker can measure his time probability of being in each state in order to maximize the attack success while minimizing the energy cost. The ideal energy cost for this objective is when $c = 0.5$.

$$\begin{cases} P_{RX} = 0.49, P_{TX} = 0.0035, P_{IDLE} = 0.01, P_{SLEEP} = 0.249 \\ \text{if (c=0.5)} \end{cases}$$

Fig 5.8 shows the energy consumption from the attacker's point of view. We can notice that HARPAGON consumes two times less energy than the basic eavesdropping attack. Indeed, the energy consumption for the eavesdropping attack is 81.6 Joules against 39.98 Joules for the attack based on the intelligent process. With the basic eavesdropping attack, all the packets are recorded by the attacker. In fact, during 4 minutes of transmission, 200 packets on average transits through the network. However, the smart attack listens on average 176 packets for an attack of the same duration. Consequently, the HARPAGON coupled with eavesdropping attack consumes 50 percent less energy but has a success rate of 88 %. If we report these values in terms of AEE, the HARPAGON attack has an AEE score of 4.4 and the basic attack of 2.4, i.e almost double.

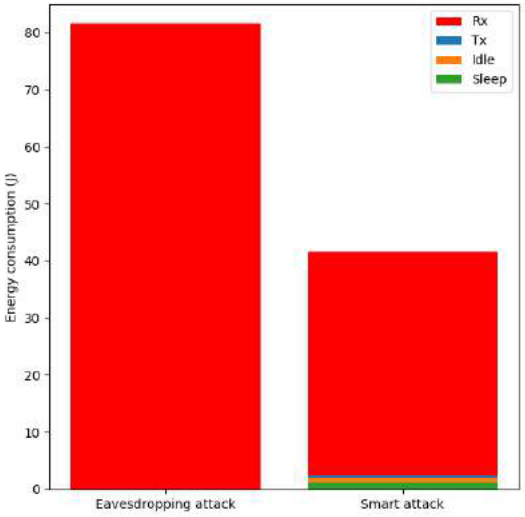


Figure 5.8: Comparison of energy consumption between basic eavesdropping and HARPAGON attack coupled with eavesdropping attack

5.2.3 Evaluation of HARPAGON framework

Like any new approach, it is necessary to evaluate the performance with other existing ones. In this section, we compare our framework with other methods for generating green attacks, i.e attacks that consume low energy. In [94], authors implement a new model to create green jamming attack named "LearnJam". In this solution, the attacker alternates between two phases: learning and attacking phase. During learning phase, the jammer keeps listening to the communication between two nodes. It records the time instances of incoming pulses over the wireless channel, which indicate the transmission instances. Based on the information the attacker obtained during its listening time, a time interval is calculated. This interval corresponds to the time between two

transmissions. With the help of this metric and the energy budget available, the jammer computes with an optimisation problem its active period time. Consequently, with a timer system the attacker alternates between two operating modes: sleep and transmission. The jammer wakes up at the beginning of the time transmission and jams the channel during its active period time. After, the jammer remains asleep until the end of the time interval between two transmissions. In [93], the authors also focus on creating a green jamming attack. First, the authors decompose this problem into three sub-problems and then obtain the jointly global-optimal solution by using outcomes of these three sub-problems. The first sub-problem concerns the optimal listening rate, which tries to find the optimal listening time to retrieve a maximum of useful information. The second, the optimal jamming power sub-problem, has been defined to find the optimal power of the sending signal to perform the attack. To finish, the last sub-problem concerns the optimal mode selection. The goal is to define the optimal time of jamming and eavesdropping. The jointly global-optimal solution combines the three results and tries to maximise the Attacker Energy Efficiency metric with a power constraint.

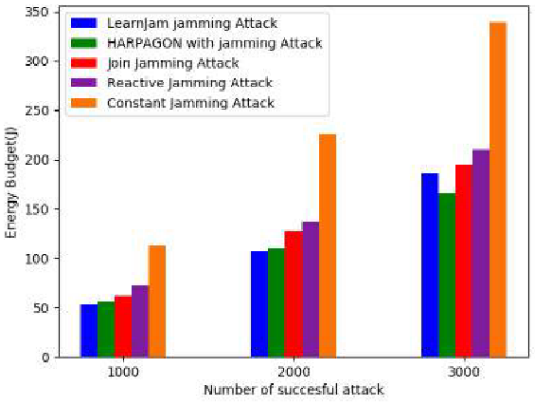


Figure 5.9: Performances of different Jamming strategies with different energy budgets

Fig 5.9 represents the comparison of these two previous methods and HARPAGON combined with jamming attack according to the energy budget. The values of the different energy consumptions for LearJam and Join Jamming attacks can be found in the articles [93, 94]. We also compare this performance with the two classic jamming attacks: constant and reactive. For 50J of energy budget, the LearnJam attack performs 940 successful attacks against 826 for the jointly method and 909 for the HARPAGON framework. In addition, an attacker with 250J of energy can successfully jam a network 3200 times with the LearnJam jammer, 3076 times with the Join method jammer and 3636 times with the HARPAGON jammer. Therefore, for a low energy budget, the LearnJam attack has a slightly better performance than HARPAGON attack, although the gap remains small (i.e 31

successful attacks). For the HARPAGON attack, the number of successful attacks is almost linear with the energy budget, as the probability of accomplishing a jamming attack is basically invariant with the elapse of time. Consequently, the higher the attacker's energy budget, the more effective the HARPAGON attack will be compared to other attacks. Indeed, in the LearnJam model, the higher the energy budget, the longer the learning phase will be. Moreover, our framework compared to the other two models has the possibility of maximising the listening or transmission time depending on the type of attack desired. Our framework is adaptable for different types of attacks and takes into account the idle state which is a mandatory transition state between sleep mode and receive/transmit mode in communication protocols.

5.3 Results obtained with FOLPETTI framework

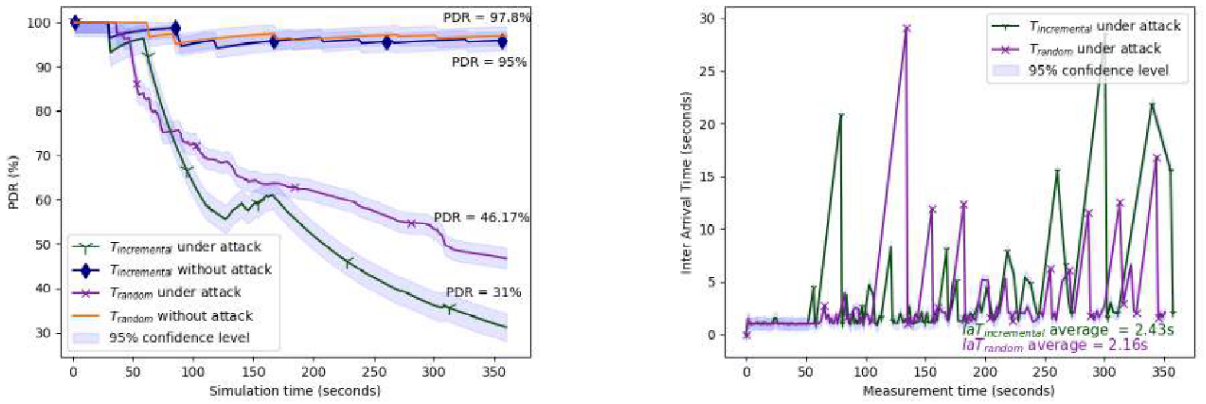
In this section, we test our FOLPETTI framework on the test-bed developed. As the distance between the elements of the network plays an important role in the transmission of a packet, this parameter remains fixed throughout the experimentation. The transmitter and the receiver are placed at 1 meter each from the access point and the attacker at 5 meters, as shown in Fig 5.10. The duration of the experiments below is fixed at 6 minutes. At the 30th second, the mitigation method begins to operate, then 30 seconds later the attacker starts to jam. Each experiment was repeated 5 times and the results presented below are an average of these with a 95% confidence interval.

Figure 5.10: Testbed implementation for the evaluation of FOLPETTI framework

We evaluate our new framework in two scenarios. In section 5.3.1, we consider a victim using basic channel hopping, i.e. incremental or random channel hopping. Then, in Section 5.3.2 we push our evaluation further by taking into account a victim exploiting the aforementioned smart channel-hopping strategy. For these two cases, we compare the different strategies in terms of a) Packet Delivery Ratio (PDR), b) number of retransmissions, c) packet Inter arrival Time (IaT).

5.3.1 Basic Channel Hopping Methods

Fig.5.11a shows the impact of our attack on the network considering the PDR evolution over time. We notice that the PDR remains high when no attack takes place. Indeed, after 6 minutes of experiments, the PDR is around 97.5% using the random channel hopping strategy. The same behavior is also achieved via incremental channel hopping, where the PDR remains above 95%. The experiments were carried out in a real environment, that is to say in the presence of



(a) (b)

Figure 5.11: a)PDR for different basic channel hopping defense strategies against smart attacks b)IaT for different basic channel hopping defense strategies against FOLPETTI framework

other neighboring networks. Indeed, we noted the presence of 14 other APs present in the building and some channel presented a high occupancy rate. The drop in PDR is therefore explained by the natural interference caused between the occupants of the channels. When the attack takes place, the PDR drops dramatically when considering incremental channel hopping. Indeed, due to the victim's periodical pattern, our attacker can efficiently predict the victim's future channel. Consequently, at the end of the experimentation, the PDR is around 31%. On the other hand, when the victim is less predictable thanks to random channel hopping, we notice that our attack still impacts the PDR. Indeed, our smart attacker manages to drop the PDR by 51.62%, being hence effective in the victim's channel prediction. One of the visible effects of jamming attacks is also an increase in the number of retransmissions. Indeed, with certain protocols such as TCP-IP, packets that are malformed or that are not acknowledged have the possibility of being retransmitted.

Tab 5.6 shows the number of victim's retransmissions both in presence and absence of the attack. Without attack, the number of transmitted packets is equal to 647.5 and 648.75 for incremental and random channel hopping, respectively. The number of retransmission is low for both cases, around 10. However, when the attack is executed, the number of retransmissions increases dramatically. Indeed, for both cases, the number of retransmitted packets undergoes a ten-fold increase compared to the cases without the attack. As seen previously, the jamming attack impacts communication also in terms of total number of packets sent. Indeed, the number of packets sent is halved when victim follows the incremental channel hopping strategy.

The effect of FOLPETTI attack coupled with jamming on basic channel hopping strategies is also visible in terms of

Table 5.6: Number of retransmissions for different channel hopping strategies without and under attack.

	Without Attack		**Under Attack**	
Type	Total number of packets	Number of retransmissions	Total number of packets	Number of retransmissions
Incremental Channel Hopping	647.5	10.5	365.75	114.75
Random Channel Hopping	648.75	9.75	403	107.66

packet IaT. Fig. 5.11b shows the IaT for the two basic channel hopping methods. As we observe in the figure, the average packet inter arrival time is equal to 2.43 seconds with the incremental channel hopping method and 2.16 seconds with the random channel hopping method. In both cases, at a given time, the arrival time between two packets can exceed 5 seconds, which corresponds to the effect of a successful jamming attack. It can be hence deduced that the attacker occupies the same channel as the victim at the same time.

Tab.5.7, summarizes the accuracy of the FOLPETTI jammer model according to several strategies of channel hopping. The computation of the accuracy. The calculation to calculate the accuracy of a machine learning model has already been discussed in chapter2. Briefly, the accuracy of a model corresponds to the number of good actions taken out of the total number of actions.

Table 5.7: Accuracy of the jamming model against different channel hopping methods.

Type	**Accuracy(%)**
Incremental Channel Hopping	82% (+/- 4 %)
Random Channel Hopping	75.6% (+/- 2 %)
Smart Channel Hopping	65.2% (+/- 7.2 %)

The results for the incremental and random channel hopping methods confirm the inefficiency of these solutions. Indeed, after 5 minutes of attack, the model has an accuracy of 82% and 75.6% for the incremental and random channel hopping, respectively.

5.3.2 Smart Channel Hopping Methods

To further validate the effectiveness of our attack, we test it against a victim having the ability to react according to his

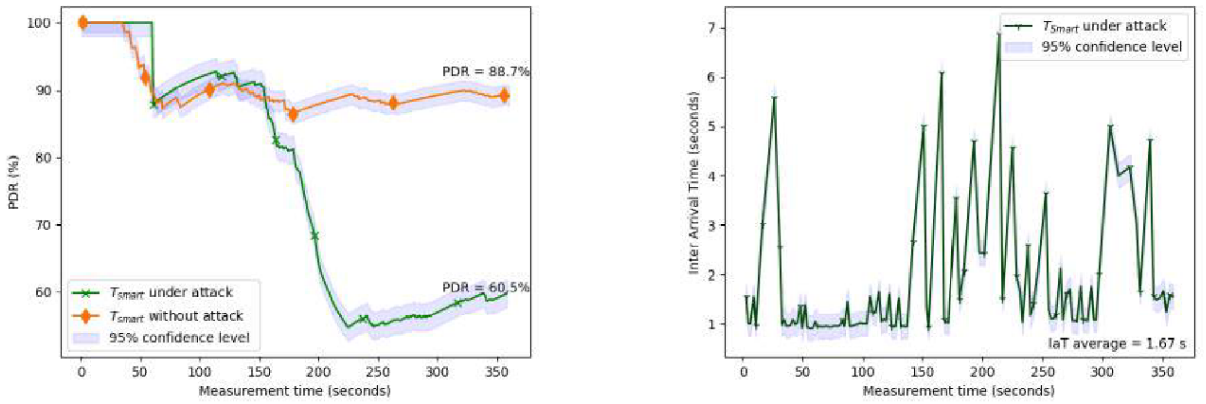


Figure 5.12: a)PDR for smart channel hopping defense strategy against FOLPETTI attacks b)IaT for smart channel hopping defense strategy against FOLPETTI attacks

environment. The channel hopping strategy also employs a MAB algorithm with a Thompson Sampling method as described in [65]. The victim's reward is calculated on the access point side and it is based on two metrics: the PDR, and the number of nodes lost after the channel change. As previously discussed, both the presence of other networks and the jamming attack impact on the PDR. Additionally, if a jamming attack has already taken place on another channel, and the access point makes the wrong decision and orders other network nodes to go to that channel, communication cannot be reestablished because the nodes will be under attack. Therefore, as we are in a real environment and other communications take place in parallel, we set the detection threshold of the PDR to 60%. As our experienced network is composed by two nodes, the threshold on the number of connected nodes is defined at 1. Consequently, if the PDR is less than 60% and the number of connected nodes less than 1, the environment returns 0 as a reward to our AP, which updates its policy with this information.

The behavior of the PDR obtained after 6 minutes of experimentation with and without attack is shown in the Fig 5.12a. Surprisingly, without attack the PDR at the end of the experimentation is around 88.7%, which is 6.3% less than with basic channel hopping methods. This can be explained by the fact that our victim initially explores all the channels several times before converging on the optimal solution. This convergence time causes the PDR to drop during the first stage which is represented on the curve between the timestamps 30 and 80 seconds. Then, the algorithm converges towards the optimal solution and manages to maintain a stable PDR approximately equal to 88.7%. In case of attack, we observe that at the end of the simulation the PDR is around 60.5%, i.e. the detection threshold. Indeed, we have noticed after several experiments that the attacker has a significant effect after a

Table 5.8: Number of retransmissions for different channel hopping strategies without and under attack.

	Without Attack		**Under Attack**	
Type	Total number of packets	Number of retransmissions	Total number of packets	Number of retransmissions
Smart Channel Hopping	552.6	47.33	480.4	89.85

few minutes of experimentation, which is reflected here by a significant drop in PDR from 81% to 58% at times 180 and 225. Then, the MAB-based channel hopping method changes its strategy because it receives rewards fixed to 0 for a certain time (the PDR being below the detection threshold). As the strategy changes, the attacker must adapt to it. However, from 250 seconds, as seen in the figure, the PDR remains close to the detection threshold. i.e. 60%.

The effectiveness of our attacks is also confirmed by the total number of retransmissions showed in Tab 5.8. Although the total number of retransmissions generated by the attacker is smaller than when the victim uses basic channel hopping methods, our attack still causes a twofold increase.

Finally, for the IaT metric, we observe in Fig 5.12b that the MAB algorithm manages to limit the effects of jamming attacks. This method has higher performance than basic channel hopping strategies. Indeed, the IaT average for random channel hopping is equal to 2.16 seconds against 1.67 seconds for the smart strategy. Nevertheless, the attacker causes certain transmissions take a long time to arrive at their destination (more than 2 seconds at a distance of 1 meter), therefore impacting the performance of the network. To conclude, we observe in Tab 5.7 that our new jammer has a high accuracy even if the detection method is based on a smart approach. Indeed, the obtained percentage accuracy after 5 minutes of experimentation is 62.5%.

5.4 Discussion on the victim's lifetime

The goal of this thesis is to create smart denial-of-sleep attacks. In this section, based on an example use case, we demonstrate the impact of these two new frameworks on the victim's life. We based our example on a use case of a sensor in an industrial place. Indeed, in some professional sectors, IoT networks are utilised for operational applications, often associated with maintenance and elementary denial of service attacks can produce immediate consequences. In some cases like in the chemistry industry, it is necessary to monitor the temperature and vibrations of industrial motors and detect the irregular operation in it. The sensors installed

on these machines will ensure industrial maintenance by alerting to the slightest problem. Based on this example, below we show the consequences of these two frameworks combined with a jamming attack.

5.4.1 HARPAGON

In this section, we suppose that the sensor has the same behaviour that the transmitter of the testbed. It sends its data once a day, which in a typical situation is an average of 204 packets sent in four minutes. Moreover, the sensor is equipped with a battery with a capacity of 133,200 J (10,000 mAh) and a Realtek RTL8187L device. The power consumption for each state of this wireless network device has already been mentioned in the previous Section 5.1. In this case study, the energy expended by the sensor to carry out these measurements/calculations and when it is inactive, is not taken into account. Only the energy expended while sending this data is calculated. Based on these data, we are able to estimate the approximate life time of the sensor in a case without attack. An example calculation is listed in Table 5.9.

Table 5.9: Battery life for an IEEE 802.11 sensor device.

Process	Value
Number of packets per day	204
Battery Capacity	133,200 J (10,000 mAh)
Time of Tx mode for one transmission	0.45 seconds
Time of Rx mode for one transmission	0.40 seconds
Energy Spent in Tx mode for one transmission	0.3015 J
Energy Spent in Rx mode for one transmission	0.136 J
Energy Spent total for one transmission	0.4375 J
Total energy spent in 204 transmission per day	89.25 J
Expected device life time	4 Years

First of all, we assume an attacker employs the constant jamming strategy and targets the sensor. As we noticed in Section 5.2, from the beginning of the attack, the attacker has an impact on the transmission. Indeed, the packets are no longer transmitted, and the sensor will stay active. However, as we are reported during experimentation, the device will disconnect from the access point after a few seconds. Consequently, the administrator will immediately notice there is a

problem with the sensor. This attack is effective in restricting communication, but it is clearly identified without a detection system. As seen in the previous section, if the attacker uses a reactive jamming attack and no detection system is in place, it will generate an average of 26 re-transmissions. This represents an extra power consumption of 11.375 J per day to transmit 204 packets. Reported in terms of lifetime and taking into account only the energy consumption generated by the transmissions, the sensor will be operational for 3.62 years. However, with the HARPAGON attack, the attacker causes 48 supplementary transmissions, which represents an additional energy expenditure of 21 J. In this case, the existence of the sensor is equivalent to 3.31 years.

It is now assumed that the network has a detection system based on the threshold like explained above. Consequently, when the network detects an attack, the nodes change channel frequency to continue transmitting. Before the system detects a reactive attack, the device will send 6 extra packets which represents an added cost of 2.625 J. The lifetime of the sensor is therefore reduced by 0.05 years. However, if the attacker uses a HARPAGON attack, as we saw in Section 5.2, the PDR decreases less quickly and this type of attack takes longer to be identified. As a result, the network will send 11 additional packets and consumes 4.8125 J more per day. Its lifespan will suffer a 14% reduction, decreasing from 4 to 3.43 years.

In conclusion, a HARPAGON attack increases the energy consumption of its victim by 15% when the network possesses a detection system. The reactive attack decreases the lifespan of its target by 1.25%, which is 13.75% less than HARPAGON attack. Moreover, if we assume our attacker is a node with the same capacity of the battery and a wireless chip as its victim, it is also possible to calculate its lifetime. Based on Table 5.4, for one reactive attack, the attacker consumes 51,8517 J and for the HARPAGON Attack 39 J. Consequently, a HARPAGON attack can be executed once a day for 9.35 years on this type of node against 7.03 years for the reactive attack. For the same duration of attack, HARPAGON attack allows to save 24.82% of energy compared to reactive attack while having a more significant impact of 15% on the lifespan of its victim.

5.4.2 FOLPETTI

Using the same logic used in the previous section, we evaluated the victim power consumption overhead generated by the FOLPETTI attack under the different channel hopping strategies. Based on the table 5.6 and table 5.8, it is possible to assess the power consumption when the victim uses a basic or smart channel hopping method. Let us notice here that in our calculations only the energy expended by the transmissions is taken into account. In this case, we assume that your sensor sends data once a day for 6 min, which is an average

of 647 transmissions with the basic channel hopping method and 552 with the smart approach. Lifetime calculation values for each strategy are given in the table 5.10.

Table 5.10: Battery life for an IEEE 802.11 sensor device with different channel hopping methods.

Process	Basic Channel Hopping Values	Smart Channel Hopping Values
Number of packets per day	647	552
Battery Capacity	133,200 J (10,000 mAh)	133,200 J (10,000 mAh)
Time of Tx mode for one transmission	0.45 seconds	0.45 seconds
Time of Rx mode for one transmission	0.40 seconds	0.40 seconds
Energy Spent in Tx mode for one transmission	0.3015 J	0.3015 J
Energy Spent in Rx mode for one transmission	0.136 J	0.136 J
Energy Spent total for one transmission	0.4375 J	0.4375 J
Total energy spent per day	283.06 J	241 J
Expected device life time	1.28 year	1.5 year

As observed in the previous section, if the attacker uses the FOLPETTI attack against a basic channel hopping method, an average of 104 retransmissions will occur. The number of retransmissions will then lead to an energy expenditure of 45.55J more per day. Reported in terms of years, the device will therefore have a lifespan of 1.11 years, i.e 0.17 less than without attack. In the case of Smart Mitigation method, the normal lifetime of the IoT device is around 1.5 years. However, under the influence of the FOLPETTI attack, the lifetime of this IoT object will also be reduced from 1.5 to 1.4 years.

5.5 Conclusion

In this chapter, we have confirmed the feasibility of these attacks in real life through experimentation. First we have combined HARPAGON framework with a jamming attack and demonstrate the impacts of the latter in a real testbed.

We show this type of attack can reduce the attacker's energy consumption by 24.82% and improve this impact by 13.75% compared to a classic jamming attack. Then in the same testbed we also implemented the FOLPETTI framework and also associated it with a jamming attack. We were able to evaluate different frequency hopping methods, from the simplest strategies to a more advanced strategy based on Multi Armed Bandit. The results obtained show that frequency hopping can be useful in mitigating irregular interference from that do not have the primary objective of creating attacks. In other words, splitting in the form of frequency is effective in allowing multiple networks to communicate on the same frequencies. On the other hand, frequency hopping seems to be ineffective against a smart jamming attack. Indeed, when the channel hopping used is a basic method, the PDR drops from 95% to 31% after five minutes under an intelligent jamming attack such as the one exposed in this thesis. Even though the access point uses a smarter frequency hopping system, the PDR drops from 88.7% to 60.5% on average against the same attack under the same time.

Finally,in this chapter, we also proved that expensive hardware is not an essential requirement for the creation of smart attack. Indeed, the wireless card to generate these two types of attacks is accessible to everyone on the web around thirty euros. Moreover, these two types of attacks do not need to know its victim, consequently its implementation remains relatively easy. The attacker does not need to spend a lot of time analyzing or processing data for example. We therefore want to highlight the dangerousness of these attacks because they become accessible to all.

When attacks become defensive methods

6

In recent years, unmanned aerial systems (UAS) have continued to increase in number, technical complexity and capabilities. Their characteristics such as their mobility have made it possible to respond to many problems in several civil and military sectors. Moreover, with the rapid rise of recreational drones in commercial areas, security and privacy issues have emerged. With this type of technology, it becomes easy for an individual to illegally acquire private data or images. In this case, we speak of passive drone attack, because the drone does not influence the behaviour of its environment. Privacy attacks can lead to severe damage such as intrusion into a private area. One of the methods to counter drones is to jam the communication between them and their controller. The main objective is to stop the drone's complete communication to stop the video transmission and the commands.

In this chapter, we present how a jamming attack can be employed as defence method to counter a passive drone attack. However, as demonstrated by the US Mitigation Airport [95] and the Russian military with the Stupor weapon [96], the main disadvantage of this kind of attack is that it also jams other communications present on the same frequency. After a brief description of the diverse types of commercial drones, we first performed an analysis of their physical and data link layers in Section 6.1.2. Then we present in Section 6.2, a new type of attack, named ICARO, to jam a specific WiFi drone without disturbing other communications. Finally, in section 6.4, after explaining the main advantages of the ICARO attack, we demonstrate how FOLPETTI can be employed to counter drones that use a proprietary protocol such as DJI drones.

6.1 Illegal drones: if jamming attacks become a defense method

6.1.1 Different type of drones

The considerable progress made in the field of aeronautics combined with those in the field of telecommunications have given rise to a new type of object that is both flying and communicating, commonly called drones. The term drone refers to unmanned aerial vehicles, that is to say, a robot with the ability to fly without anyone on board. The idea is not new, indeed like most of the technological advances of the last century, the concept and the first prototypes were developed during the First World War for military purposes.

However, it was not until the arrival of several firms such as the French company PARROT, the Chinese company DJI and the American company GO-PRO to market and make drones accessible to everyone in the world [97, 98].

Table 6.1: Comparison of different drones

Type			
Name:	Google's Loon	Zipline	ANAFI
First Introduced:	2013	2016	2019
Company:	Google	Zipline	Parrot
Flight Range:	-	80km	2-5 km
Power Source:	Solar and Batteries	Batteries	Batteries
Application:	Communication	Delivery	Data Collecting

However, depending on their use case and manufacturer, the configuration of a drone can vary like all IoT devices. As a result, in the literature, several classifications of them have taken place according to these different parameters [99]. In this section, we briefly describe the different characteristics of drones depending on their use cases. Table 6.1 provides several examples of drones according to their domains, which are as follows:

- **Network Applications :** Drones have the ability to move easily and fly over most obstacles that a human could not overcome. Therefore, it would be of significant help to ensure connectivity in certain hostile locations. We could consider access points on board drones which would therefore have the ability to be mobile and move as needed. Google's LOON project launched in 2013 is an example of this usage [100]. In the same way, thanks to their mobility, a drone could make it possible to ensure a quality of service in the networks when the environment changes or deteriorates such as in natural disasters. Dynamic routings according to several parameters of the environment could be set up thanks to this technology. To best meet these needs, drones must therefore be able to provide continuous long-range communication. However, as we have seen previously, the transmissions are energy-intensive. Hence, the greater the communication distance, the more energy the drone

will consume. In parallel, the weight represents also an important factor that can vary the energy consumption of a drone. As in this use case, the drone does not need to carry many elements, the latter can be designed to be as light as possible to save as much energy. Consequently, drones designed to ensure connectivity will by definition be lightweight but will have significant range transmission.

- **Delivery:** Drones can also be used for delivery purposes. The shortest route from one point to another is often as the crow flies in real life because it avoids natural detours such as mountains or human obstacles. In this sense, a drone could facilitate and accelerate access to delivery in remote areas. The ZipLine company is harnessing this idea by providing drone distribution of various medical products to hostile locations in Rwanda, Ghana and Nigeria [101]. The Amazon company is also starting to implement its style of service to ensure even faster deliveries [102]. It can also be useful in agriculture to deploy fertilizer over large areas and missiles in military terrain. These drones need to be large and sturdy enough to support the weight of their products. Compared to the drone providing connectivity, manufacturers can play neither on the weight of this product nor on its shape. As a result, we will have larger, heavier, faster drones using either battery or directly fuels such as petroleum.
- **Data Collecting:** Finally, a drone can be used for data collection. In this category, two sub-missions can be cited: 3D mapping and video surveillance. 3D mapping consists of equipping a drone with multiple sensors such as a camera but also distance sensing elements to be able to create or model 3D mapping environments that are difficult to access. Naturally, this application is very practical in the military field (mapping of the opposing camp) but it can also be utilized in the civilian area. Indeed, it can be employed to precisely determine the area of land during real-estate transactions or to model archaeologically or tourist sites [103]. A drone also plays a significant role in video surveillance [104]. It can provide monitoring as in agriculture (remote monitoring of herds in the mountains) or humans during sporting events for example. Drones carrying out this type of mission will therefore be equipped with cameras and sensors and do not necessarily require a high communication range compared to the other two use cases. They are therefore smaller and will have a lower energy cost. Additionally, the user needs to have video feedback of what the drone is flying over, so the flight mode is more often manual than automatic.

6.1.2 How to block an illicit drone ?

In this study we focused on the last use case cited just above, that is to say drones designed for data collection. Indeed, these types of drones are more accessible to the public and can be easily hijacked to create attacks such as passive attacks. By the term "passive attack", we refer to the act of recording and collecting private images to infer personal information. This type of attack violates privacy and can lead to real consequences such as intrusion into a private place as demonstrated in [105]. Moreover, as the war in Ukraine in recent months has shown, in the military context, increasingly drones are also used to collect data on the enemy [106]. It is therefore urgent to find a new solution against this type of use of drones in civilian and military fields. A solution found in the literature, to protect civilians against illegal surveillance drones is the method of protective filters which consists of including noise to the video like blurring, pixelation and masking [107, 108]. However, as explained in [109], this method can be easily circumvented by adding another drone that films the scene from a second angle. By adding the two images, it is then possible to reconstruct data. Another solution to prevent the surveillance of illicit drones is to cut off communication with their controller with jamming attacks. In the case of an illicit surveillance drone, the goal is to jam the communication between the controller and the UAV in flight to suspend the video transmission and flight control. In recent years, several jamming attacks in this context have been developed. We report their main advantages and disadvantages in Table 6.2.

The works in [110] prove it is possible to cut off a drone's communication with a commercial jammer when the distance between the target and the attacker is short. However, the commercial attacker jams a wide band of frequencies and disrupts the complete communication around it. A similar idea was developed in [111], where the authors implement their own jamming attack based on Data Link Layer, with a *Raspberry-Py*. Consequently, the jammer is accessible and reproducible by all. However, they create a constant jamming attack and even if the communication between the drone and its controller is interrupted, other communications are impacted. In [112], the authors develop a jammer capable of jamming only an enemy and keeping other wireless communications connected with a secret key mechanism. The principal idea is to permanently jam the illicit drone so that the jamming signals are unpredictable interference for unauthorized devices, but can be picked up by authorized machines equipped with the secret keys. The authors show that the bit error rate of the enemy network is strongly impacted by this type of attack, unlike the authorized network which remains stable. However, if the private key cannot be exchanged in advance, this solution is inapplicable. To minimize interference with neighbouring networks, the authors in

Table 6.2: Survey of jamming attack against illicit drones

Ref	Year	Evaluation	Advantages	Disadvantages
[110]	2016	Experimental	Cheaper tools	Impact neighboring network
[111]	2019	Experimental	Cheaper tools	Impact neighboring network
[112]	2013	Experimental	Does not impact neighboring networks	► Require secret key ► USRP
[113]	2018	Simulation	Counter jamming attack	Theoretical Analysis
[114]	2018	Experimental	Does not impact neighboring networks	► Prior knowledge ► FPGA

[114], try to get as close as possible to the characteristics of the signal emitted by the drone. They assume knowing the communication protocol used as well as the frequency hopping and modulation to generate identical signals. They prove that having knowledge of these characteristics in advance makes it possible to considerably reduce the interference on neighboring networks. In [113, 115], the authors highlight the fact that it is urgent to design cheap jammers which can jam only the targeted drone and not the surrounding networks. Tedeschi et *al.* show that for a jammer to succeed in cutting off the communication of a drone, the transmission power must also adapt to the drone [113]. The authors of these two articles conclude by proposing the potential use of a machine learning algorithm to design an intelligent jammer capable of adapting its strategy according to drone communication.

Figure 6.1: Russian Stupor anti-drone guns

Based on these different works, we wanted to set up a cheap jammer capable of adapting according to the drone's communication strategy. These attacks are currently generated at the physical layer, which is why Software-defined radio (SDR) boards or Arbitrary Waveform Generator(AWG) generators are required. As shown in Fig 6.1, this type of equipment is bulky and consumes a lot of energy. Consequently, generating jamming attacks on the Data Link layer with light and intelligent processes could make it possible to lighten this type of weapon and increase its effectiveness. Moreover, as suggested by previous works, we used a machine learning approach to design a new jammer capable of cutting off the

communication in a dynamic and adaptable way.

6.1.3 Study of the physical and Data Link layers of different types of drone

It is in this sense that we wanted to develop an intelligent jamming attack that could cut off the communication of a specific drone avoiding disturbing external transmissions. However, as we have seen, there are several types of drones with diverse characteristics depending on the method of manufacture used. After several unsuccessful searches for documents explaining the differences in communication protocol between different drone models, we decided to conduct our own analysis. Therefore, we performed physical and Data Link layer behavioural analysis of several types of commercial drones: three PARROT drones and two DJI drones. The models utilized during this analysis are an ANAFI drone (PARROT), a DISCO drone (PARROT), a BEBOP 2 drone (PARROT), a PHANTOM3 (DJI) and a Mavic AIR 2S (DJI). The primary difference between the Parrot drones is that the ANAFI drone uses another type of controller (controller 3) compared to the DISCO and BEBOP drones (controller 2).

As described in the various white papers on the official Parrot website, their Drone utilizes the 802.11 (WIFI) communication protocol [116]. The Chinese company, DJI, uses a proprietary communication protocol or sometimes also named "elaborated WIFI". Hence, using a *Raspberry-Pi*, an 802.11 network card including monitor mode and the Wireshark tool, we first sniffed the packets from the DISCO and ANAFI drones.

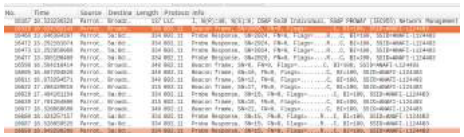
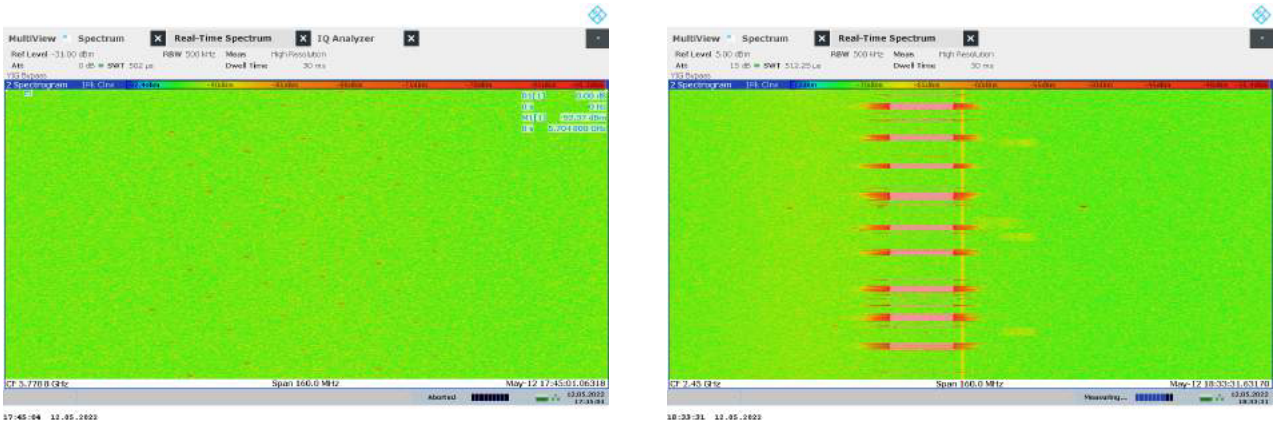


Figure 6.2: Example of ANAFI sniffing

Without significant discovery, we were able to confirm it was possible to sniff from the data link layer the packets of this type of drone as shown in Fig 6.2. The packets being visible and not encrypted, we were capable to validate that it was indeed a WiFi communication and that no security was managed for the DISCO drone. The same conclusion was equally inferred for the BEBOP drone. However, a difference with the ANAFI drone is visible although the WiFi protocol is also used. Indeed, for the ANAFI drone, the content of the data packets cannot be read without knowing the security key. To prevent passive eavesdropping and replay attacks, PARROT has included the WiFi Protected Access 2 (WPA2) encryption protocol directly on controller 3. For DJI drones, no communication could be detected from the Data Link layer.

As we could not analyze the behavior of DJI drones on the Data Link layer, we decided to make measurements directly on the physical layer. Our objective was to observe if DJI drones communicated on the same and unique channel or if frequency hopping methods were used. In an anechoic chamber, we placed a drone and its controller with a real-time



(a) **(b)**

Figure 6.3: Examples of Real Time Spectrum for different types of drones: a)DJI-PHANTOM3 drone b) PARROT-ANAFI drone

spectrum analyzer. After several measurements, we have concluded that DJI drones use frequency hopping methods on the 2.4Ghz and 5.8Ghz bands. However, as shown in Fig 6.3a, for the DJI PHANTOM the same channel hopping pattern is always utilized. The Parrot drone always uses the same frequency to communicate with its controller. Indeed, when establishing communication, the controller scans the entire band available and selects the least disturbed frequency (channel). As mentioned in [116], the WiFi 802.11a/b/g/n amendment is used, therefore, PARROT drones are able to communicate on the 2.4Ghz or 5.8Ghz bands. All the conclusions we were able to draw from these different measurements are summarized in the Table 6.3.

Table 6.3: Comparison of different drones

Type					
Name:	DISCO	BEBOP2	ANAFI	PHANTOM3	MAVIC AIR S2
Company:	Parrot	Parrot	Parrot	DJI	DJI
Controller Type:	Controller 2	Controller 2	Controller 3	Phantom 3 professional controller	DJI RC-N1
Communication Protocol:	802.11	802.11	802.11	Proprietary protocol	Proprietary protocol
Operating frequency:	2.4GHz-5.8GHz	2.4GHz-5.8GHz	2.4GHz-5.8GHz	5.725 GHz-5.825GHz	2,400GHz-2,4835GHz and 5,725GHz-5,850GHz
Frequency hopping:	no	no	no	yes	yes

6.2 Counter a WIFI drone like a Parrot ANAFI drone?

In this section, we implement a new type of attack strongly inspired by the FOLPETTI and HARPAGON frameworks to target a particular drone. The goal of this attack is to cut off communication between a drone and its controller without disrupting transmissions around it. In this section, we explain this new attack, named ICARO, which can also be seen as a framework. With real experiments, we demonstrate the consequences of this attack in two scenarios described below: indoor and outdoor environments.

6.2.1 Attack model of ICARO

As we already said in the previous Chapter 5, the reactive jamming attack makes it possible to target a specific victim. This is because the attacker alternates between listening and transmitting modes and only jams when the victim's packet is broadcasting. However, for the reactive attack to be effective, the moment of the attack must occur before the end of the transmission. This is because the transmission must

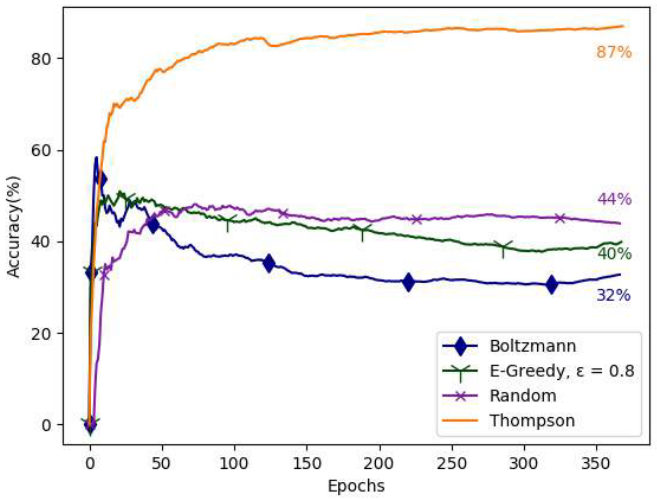


Figure 6.4: Accuracy of the various policies associated with the ICARO attack

be longer than the attacker's reaction time combined with the attack time to cause packet corruption. This transmission time depends principally on the size of the target packets and the location of the attacker. Thus, as expressed in the formula 4.1 and experimentally in the previous chapter, the jamming of short packets is generally inefficient. The idea of ICARO was born from these observations. The concept is to develop a more efficient reactive attack, i.e. to prevent the attacker from jamming a packet which will have no effect. The main objective is to increase the robustness of the attack while decreasing its energy consumption.

In the 802.11 protocols, several types of packets with different lengths are present. For example, the standard acknowledgement packets (ACK) are short. Approximately the size of this type of packet is more or less 8 bytes. On the contrary, management frames such as probe request or probe response, are larger because they contain more information like the source MAC address [117]. Based on the same approach that the FOLPETTI framework, we adapted the attacker's model. In the case of the ICARO framework, the arms of the Multi-Armed Bandit correspond to a type of packet. Therefore, based on experimentation and achieved rewards, after some time the attacker is able to block only those packets that impact the victim's communication. As already mentioned in Chapter 3, to update a multi-armed bandit algorithm, several policies have been designed in the literature [29]. Our first evaluation was to choose the optimal policy method to solve the exploration-exploitation dilemma in ICARO. Therefore, we evaluated the accuracy of the ICARO attack obtained after an attack period of one minute with various policies. Fig 6.4 reports attack accuracy for diverse types of approaches.

Three types of algorithms were considered in this study: Boltzmann, e-greedy, and Thompson Sampling. All of these

different algorithms are explained in more detail in [118]. Briefly, with the greedy method, at each round the algorithm selects the arm with the highest empirical mean with probability $1-\epsilon$, and chooses a random arm with probability ϵ. The Boltzmann algorithm is based on Luce's axiom of choice and picks each arm with a proportional probability to its average reward. The Thompson Sampling method has already been explained in detail in chapter 3. We also added an elementary exploration method, a random strategy. At each instant, the attacker randomly determines a type of packet and updates its strategy according to its feedback. As we can see in the Fig 6.4, the Thompson Sampling method is more effective. Indeed, after 360 epochs this method converges at 87% compared to the other methods which do not reach a converge above 50%. This can be explained by the fact that the Thompson Sampling algorithm is designed to be more exploratory at first.

Therefore, we adopted the Thompson Sampling algorithm, and the complete ICARO algorithm is described in Algorithm 2.

Algorithm 2 ICARO Algorithm

Require: $type$: tab of type of different packet, t_j : number of successful jamming , $timeFirst$ = time of listening during first step, $firstStep$: Boolean to execute the first step, $timeJamming$: Time of Jamming, $timeListen$: Time of Listening

1: **if** $firstStep = true$ **then** ▷ First Step
2: $type$ = FirstListening(time)
3: $t_j = c = 0$ ▷ Second Step
4: **while** True **do**
5: **for** `j=0,j>length(type),j++` **do**
6: sample $r_j \sim$ beta $(\alpha_j + t_j, \beta_j + t_j)$
7: $packetType =$ argmax $\{\bar{r}_j\}$
8: c ++
9: ReactiveJAM($m, timeListen, timeJamming$)
10: $newPacket$ = Listen($timeListen$)
11: **if** $newPacket ==$ retry or $newPacket == null$ **then**
12: t_m+ = 1
13: **else**
14: t_m+ = 0
15: UpdatePolicy(m, t_m)

The first step of the ICARO attack consists in listening to the communication for a certain time in order to recognize the diverse types of packets that are transmitted by the victim. This step is optional, the attacker can decide to launch the attack directly from part 2, i.e at line 4. Indeed, as we have already mentioned, the goal of ICARO is to stop a drone's communications. Therefore, we assume that a detection system has already identified the presence of said drone as well as its type. Thus, thanks to this knowledge of type and brand, we are able to deduce the communication protocol

used as well as all types of packets that it can use. For example, if an ANAFI drone has been identified, ICARO can deduce the 802.11 protocol is used and twelve types of various packet management, six types of packet control and eight types of data frames could be employed by the drone. However, the first step reduces the convergence time of the attack by limiting the number of possible choices for the multi-armed bandit algorithm. The second step represents the main algorithm of the ICARO attack and consists of the policy update. The success of the attack is determined from several metrics like the number of retransmissions and the idle time of the victim. Indeed, most of the time, if a packet is corrupted it will be retransmitted by the sender. This information is visible directly with the sequence number of the frame. Additionally, the jammer also gains a positive reward if no transmissions are sniffed for a period after the jamming phase. The advantages of these metrics are they can be obtained easily and even when the communication is encrypted with the WiFi Protected Access 2 (WPA2) protocol.

6.3 Indoor Environment Evaluation

We first evaluated ICARO in a real indoor environment to demonstrate its potential effect on a targeted drone and other nearby communications. The main goal of this scenario is to prove that ICARO attack is able to cut off the communication of its victim, without disrupts the other concurrent communications, even though they occur at the same frequency of the target drone. In the following section we describe in detail the environment and then the results obtained.

Description of the network model

The environment is composed of two legitimate nodes connected via an Access Point, two drones, one attacker and one sniffer, as shown in Fig 6.5. The main parameters are succinctly summarized in Table 6.4

The composition of the network is as follows:

- **The targeted drone:** The targeted drone is the ANAFI model of the PARROT brand. As mentioned in the previous Section 6.1.3, the communication protocol used by the drone with its controller is the 802.11 b/g/n amendment and no channel hopping method is present.
- **The neighboring network:** As we want to show the attack does not influence other nearby networks using the identical communication protocol, we have included three classic laptops in the same room. The three classic laptops correspond to the testbed developed in previous Chapter 5 during the FOLPETTI and HARPAGON evaluations. Finally, we have included in our network model

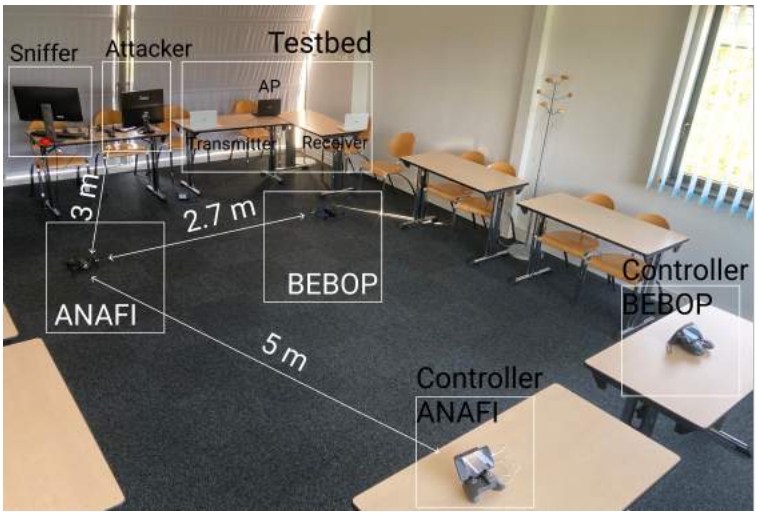


Figure 6.5: Representation of the network assessed for the Indoor Environment

another PARROT drone: the BEBOP drone. The type of communication protocol employed by the BEBOP and its controller is also the 802.11 b/g/n protocol. To evaluate the effectiveness of ICARO on the environment, we added a sniffer, a *Raspberry-Pi* coupled with *Alfa AWUS036h* device to record all communication on the channel.

- **The attacker:** The ICARO attack was implemented in the same *Raspberry-Pi* paired with the *Alfa AWUS036h* device already described in Chapter 4. In this study, reactive jamming has been implemented to scramble all packets from a specific source. Random jammer randomly selects its attack and idle time between 1 and 4 seconds.

As the distance between network elements plays a significant role in the transmission time of a packet, this parameter remains fixed throughout the experiments. As shown in Figure 6.5, the access point is placed 2 meter away from the attacker. The transmitter and the receiver are one meter away from it. The ANAFI device is placed 3 meters from the attacker and the distance between the controller and the drone is 5 meters. The similar distance for the controller and the BEBOP drone is also respected. The attacker is also 3 meters from the BEBOP drone. During the experiments the drone fly at 1 meter from the ground. The testbed laptops were configured to be on the same communication channel as the drones.

6.3.1 Assessment of the ICARO attack

To estimate the effectiveness of ICARO attack, we evaluated it against several other strategies of jamming attack which are a) constant jamming attack, b) random jamming attack, c) reactive jamming attack. Several metrics "capturing" different

Table 6.4: Indoor experiment settings

Jammer	
Parameter Name	**Setting Used**
Jamming pulse duration	0.001 Second
Packet Size	50 bytes
Power Consumption for T_x mode	0.67 W
Power Consumption for R_x mode	0.34 W
Power Consumption for $Idle$ mode	0.30 W
Power Consumption for $Sleep$ mode	0.01 W
Testbed	
Parameter Name	**Setting Used**
Packets Size	Between 50 and 1400 bytes.
Number of retransmission allowed	1
Waiting time for acknowledgment	1 Second

aspects of the network model have been considered a) the Packet Delivery Ratio (PDR), b) the percentage of disconnection time, c) the number of retransmissions generated and, d) the energy expenditure of the attacker. The results proposed below correspond to an average of five experiments taking into account a confidence interval of 95%. The attack begins after 30 seconds of experimentation and lasts 1 minute.

The first notable observation, visible in Fig 6.6a is that for the ANAFI drone, the PDR drops to 0% with the constant and ICARO attacks. Indeed, after 5 seconds of constant jamming, the communication between the controller and the drone is interrupted. The same observation is also observable with the ICARO attack. The drone is no longer capable to communicate with its controller after 9 seconds of ICARO attack. Note that the PDR of the drone without attack is not equal to 100% because the experiments were carried out in a closed environment which included many users on the transmission channel and the neighboring channels. We have denoted more than 5 other access points on the same channel and 4 on neighboring channels, hence we also have a presence of natural interference. The second observation we noticed in Fig 6.6a is that for reactive and random attacks, communication is uninterrupted. The reactive jamming has an impact but is less important than the constant or the

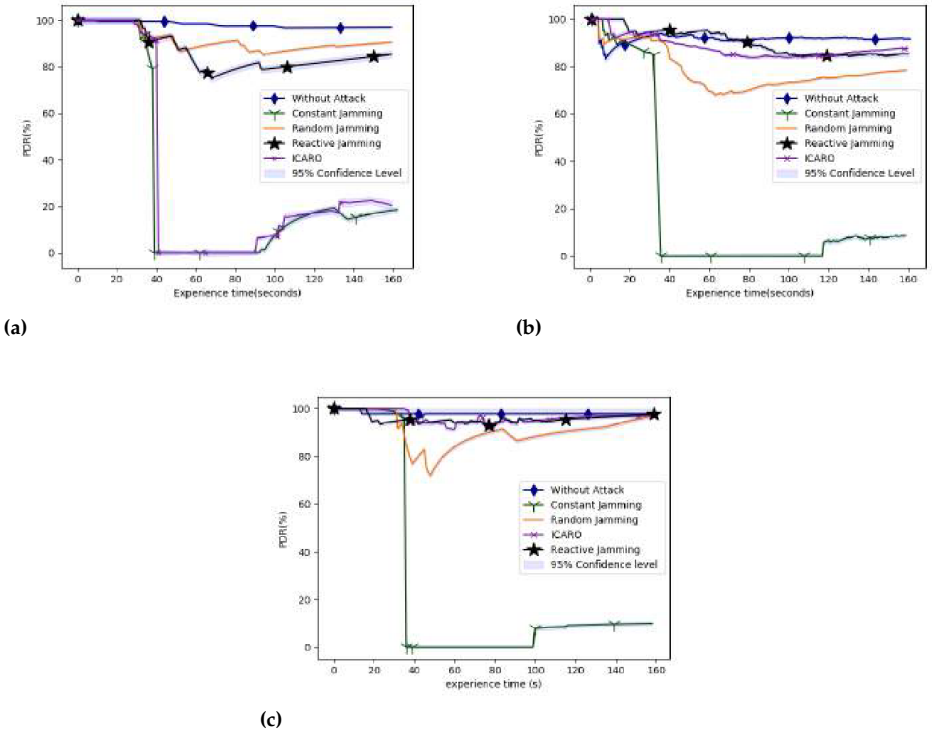


Figure 6.6: PDR values for the the different component of the network a) ANAFI Drone, b)Testbed network and c) BEBOP drone

ICARO attacks. With the random attack, the PDR reaches a minimum at 80%, and 74% with the reactive attack. Therefore, the reactive attack is more effective than the random attack but fails to cut off the drone's communication.

These results are also confirmed by the disconnected time percentage values summarized in the table 6.5. Under constant attack, the ANAFI drone is disconnected 92% of the time. During an ICARO-type attack, the drone is shut down 84% of the time, i.e. 8% less than the constant. However, this value remains considerable because with a 60 seconds attack the drone is disconnected for 50.4 seconds. With reactive and random attacks, the device stays connected to its controller, hence the disconnection time is 0%.

As our objective is to observe if the ICARO attack also impacts the communications of other neighboring networks, we then measured the PDR of the testbed under the influence of these different attacks. Fig 6.6b reports the PDR values for the testbed network. Unsurprisingly, constant and random attacks impact testbed network performance. By definition, these attacks occupy the communication channel and block all transmissions present on it. As a result, all components using the latter are affected. The reactive and ICARO attacks select the packets to be jammed to compromise them. Therefore, as confirmed by the PDR of the testbed, it does not affect communications in the surrounding area. Indeed, the PDR of the transmitter of the testbed during these two types

of attacks is equivalent to 85%, that is to say, 5% less than when there are no attacks. This slight decrease corresponds to the expected behavior when adding a new user to the same communication channel.

The same observation is perceptible in Table 6.5. Indeed, as observed with the PDR the constant attack also has important consequences on the tesbed. As a result, over a minute of attacks, we observed that the different devices of the networks are disconnected 94% of the time. The other attacks have little consequence, the communication remains stable and none of the elements of the testbed are disconnected.

Finally, we evaluated the PDF metric of the neighboring drone: the BEBOP. Fig 6.6c and Table 6.5 confirm that the ICARO attack does not influence the communication behavior of a nearby non-target drone. The disconnection time obtained with the constant attack for the BEBOP drone, i.e. 91%, also confirms that communication is degraded under its influence.

Table 6.5: Percentage of disconnection time

	Constant	**Random**	**Reactive**	**ICARO**
Testbed	94%	-	-	-
BEBOP	91%	-	-	-
ANAFI	92%	-	-	84.25%

We also evaluated the number of retransmissions generated by the different type of attacks in Table 6.6. The constant attack behavior is also confirmed with this metric. Indeed, before the various elements of neighboring networks are completely disconnected, we see that this type of interference has an impact on their efficiency. Before the testbed was completely disconnected, 15 retransmissions occurred, i.e 9 more than during a transmission without attack. 314 retransmissions are also produced with this type of assault on the ANAFI drone and 15 for the BEBOP drone. The number of retransmissions with the ICARO attack confirms the results obtained with the PDR. Indeed, for the testbed and the BEBOP drone, the results acquired under the ICARO attack are close to the values obtained during normal behavior.

We have just shown that the ICARO attack makes it possible to disconnect a target drone from its controller with the least possible impact on surrounding communications. However, since the second objective of the ICARO attack is to reduce the energy consumption of the attacker, we also evaluated the energy consumption for the different attack types in Table 6.7.

The constant jamming attack has only one state, which is the transmission state. So if we combine the formula 3.13 with

Table 6.6: Number of retransmissions generated by different types of jamming attacks

	Normal	**Constant**	**Random**	**Reactive**	**ICARO**
Testbed	6	15	73	48	11
BEBOP	14	32	54	17	15
ANAFI	62.5	119.5	161	314	71

Table 6.4, the total energy expended for one minute of attack is 40.2J. As seen in the table 6.7, the random jamming attack is the least energy intensive with an energy consumption for one minute of activity of 26.86J. This type of attack was designed to save energy as much as possible by including an inactivity phase, in this case the idle operating mode. Thus, although the jammer transmits on average for an attack minute of 19.59 seconds, the rest of the time it is in the idle state which consumes less energy. However, we can see that the energy consumption of the ICARO attack is close to that of the reactive jamming attack with a value of energy equal to 28.65J. Indeed, with this strategies the jammer alternates between two modes of operating which are listening and transmitting. Although the receive state is a little more energy intensive than the idle mode, with the multi armed bandit algorithm the attacker manages to quickly converge on the optimal strategy. Consequently, the attacker does not waste its energy when it knows that its jamming will have no effect and remains in listening mode, state that consumes less energy than transition mode.

Table 6.7: Energy Consumption for different type of jamming attacks

	Constant	**Random**	**Reactive**	**ICARO**
Energy expended in T_x mode	40.2 J	13.13 J	26.8 J	13.93 J
Energy expended in T_x mode	0	0	7.4 J	14.72 J
Energy expended in $Idle$ mode	0	13.73 J	0	0
Total Energy Consumption	40.2J	26.86 J	34.2 J	28.65J

In this section, we have demonstrated that the ICARO attack can jam packets from a specific drone without impacting other communications. Moreover, the strategy employed allows

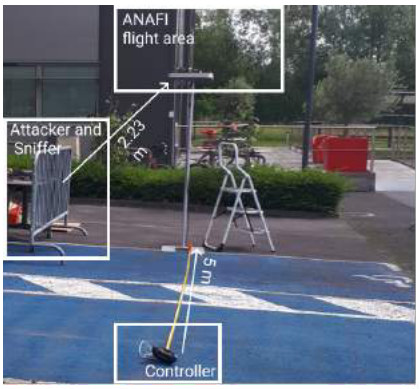


Figure 6.7: Setup of the evaluation in outdoor environment

jamming only the packets which will have consequences for the victim. Because of this, our attacker learns the optimal jamming moment and is therefore more efficient and less energy-consuming.

6.3.2 Second study in outdoor environment

The second assessment was conducted in an outdoor environment for several reasons. The first was to judge this attack in an environment with less natural interference. The second reason is to evaluate it according to the distance parameter. Therefore, we also wanted to evaluate the ICARO attack when the drone possesses the possibility of using this tool to observe how it behaves during jamming with a GPS signal. As for the previous section, we initially present the network model, then the diverse results obtained with the ICARO attack in an outdoor environment.

6.3.3 Network model

The target drone is the same type used in the previous section, the ANAFI drone of the brand PARROT. As shown in Fig 6.7, the distance between the controller and the drone is fixed at 5 meters during all the different measurements. Moreover the drone flies at a stable altitude of 2 meters. To evaluate the consequence of the distance parameter between the attacker and the drone, we made several measurements at different distances which are: 2.23m, 5.38m, 10.19m, 15.13m and 20.09m. The alfa device used for the attacker has a maximum transmission range of 50m-100m according to the manufacturers in ideal conditions [57]. We have added the same sniffer already present in the previous section to the network configuration to record all the different communications from the external environment.

Table 6.8: Results obtained for outdoor evaluation

Distance	Number of deconnections	Number of connection	Percentage of disconnection time (%)	Number of Sniffed Packets	Number of retransmissions
2.23	1	0	64.1	2529.5	154.2
5.38	1.5	0.3	55	2334.5	124.5
10.19	1. 4	0.4	45.34		
15.13	1.4	0.4	30.53	1870	93.8
20.09	0.2	-	16.66	1624	89

6.3.4 Assessment of the ICARO attack

We evaluate the performance of ICARO attack according to several metric: a) the number of deconnections, b) the number of connection, c) the percentage of disconnection time d) the number of retransmissions and e) the number of sniffed packets. For the number of connections, we have counted here only the number of successful connections. The term successful connection number refers to the fact that the 4 exchanges of messages necessary for authentication with the Extensible Authentication Protocol over LAN (eapol) protocol have been carried out. Indeed, the ANAFI drone uses the WiFi Protected Access 2 (WPA2) protocol to secure its communication. By consequently, the exchange of information (data packets) cannot take place until the key sharing is complete. The number of sniffed packets is the number of packets, all types combined, which attacker managed to sniff and read during 30 seconds of listening. The various results are reported in Table 6.8 according to the different positions of the attacker. Values reported are an average of five experiments of 30 second of ICARO attack.

The first observation we can make is that the number of sniffed packets decreases with distance. Indeed at 2.23 meters from the drone, the attacker sniffed an average of 2529.5 packets, against 1624 when it is at 20.9 meters, i.e. 905.5 packets less. Therefore, the attack being based on the reactivity of the attacker and the different packets sniffed, it would be rational to note that the attack is therefore less effective when the attacker is far from his victim. The results obtained for the number of retransmissions generated by the attack confirm this hypothesis. Indeed, the number of retransmissions when the attacker is positioned at 2.25 meters is equal to 154.2 and 89 when the attacker is at 20.09 meters. However, although the attacker is less effective when the distance is higher, the ICARO attack achieves its main objective which is to disconnect the drone from its controller. Indeed, the percentage of

disconnection times obtained at 20.09 meters is 16.66%. In these experiments, the official maximum transmission range of the attacker's antenna is about 50 m reported by the manufacturer. However, these values were calculated in an ideal environment. Although we were outside natural obstacles were present such as trees. However, the experiments carried out prove that the attack manages to be effective even at long distances. It is possible to add a long-range type antenna to the ICARO attacker, like those used by the ANAFI drone.

6.4 Discussion

6.4.1 ICARO to counter a WIFI drone

ICARO was designed to counter a WIFI drone not using the channel hopping method. Based on the results obtained in the previous Section 6.2, we observe that this attack based on ML has better performance than a basic strategy such as constant jamming method. Indeed, this new type of attack improves the strategy of a reactive attack by selecting the packets that will have the most consequences for the victim. We demonstrated that for the case of the WiFi communication between a drone and a controller this method is effective. For the indoor case, the direct consequences observed are that at the start of the attack the video will first freeze and/or be retransmitted with very high latency. Then, after 9 seconds of attack, when the controller is completely disconnected from the device, the drone will signal that if no communication is reestablished, it will land at its position within 3 minutes. Consequently, if the ICARO attack is maintained during 3 minutes, the attacker will no longer be able to recover its drone. Concerning the outdoor case, the same behavior was observed. The only difference is that when the communication is lost with the controller, the drone returns to the takeoff point. This functionality is possible because the drone has the possibility of obtaining the GPS coordinates.

We can consider ICARO attack as a framework. It main advantage that it does not require training based on dataset, the update of its policy is done in real time. The only prerequisites to perform the ICARO attack is knowing what communication protocol our victim is using. Therefore, this approach can be applicable for all types of communication protocols and for different use cases. ICARO can be also seen as a framework because an attacker can perform several type of known attack with it. In this thesis, we have used it to improve the efficiency of reactive jamming while reducing its power consumption, but it can also be used to enhance replay or grey hole attacks. In the case of a gray hole attack, it may be that the deletion of a certain type of packet has as many consequences as the deletion of all the packets. Consequently, with the same process as ICARO, the attacker will be able

to know which type of packet to delete generates the most retransmission and thus saves energy.

Compared to other works presented in Section 6.1.2, the jamming ICARO attack is elaborated from the Data Link layer. Although the response time is longer, this approach has real advantages. The creation of jamming attack require several expensive tools such as USRP components. This type of equipment is bulky and consumes a lot of energy. Consequently, generating jamming attacks on the MAC layer with light and intelligent processes could make it possible to lighten the jamming weapon of the soldier like STUPOR and make it more effective.

6.4.2 FOLPETTI to counter a Proprietary protocol

In section 6.1.3, we demonstrated that DJI drones use a proprietary protocol. The only knowledge we have is that these types of drones use frequency hopping on the 5.8Gz frequency band. In this case as DJI drone utilizes frequency hopping, ICARO cannot be used. However targeting this type of drone is possible with the FOLPETI framework. Indeed, in Chapter 5, we prove that the FOLPETTI framework coupled with a jamming attack has an impact on the victim even if the channel hopping scheme is randomly defined. The main problem in this situation is that, since it is impossible to know the communication protocol used, it is impossible to sniff the packets on the data link layer.

To resolve this problem, FOLPETTI can be directly implemented on the physical layer at the first time. Indeed, the FOLPETTI pair with a jamming attack uses the received signal strength indicator as a reward. Although this metric in Chapter 5 is obtained from the data link layer, it can be calculated by the network card from the physical layer. Consequently, it is easily possible to directly implement FOLPETTI framework in a physical layer with a signal generator.

As DJI drones operate at specific frequencies to avoid interference, the FOLPETTI attack will also be targeted. After a few seconds FOLPETTI converges and can deduce the pattern used by the targeted drone and will only jam on the latter's transmission frequency. Therefore, this type of attack is unlikely to cut communication with other nearby devices.

6.5 Conclusion

In this chapter, we presented ICARO, a novel smart attack designed to counter passive drone attacks. This new type of attack is based on Multi-Armed algorithm and permit to considerably improve the effectiveness of reactive attack

against illicit drone. We evaluated ICARO in two environments, indoor and outdoor. For the first environment, we demonstrated that ICARO attack achieves to cut-off the communication of the drone at 84.25% during all the time of attack against 92% for the basic constant jamming attack. Although these results obtained are a little lower than those obtained for a classic attack, the main advantage of ICARO is that it does not disrupt communications around it. Indeed, unlike constant attacks which disconnects the surrounding communication for 94% of the attack time, ICARO records a result of 0%. We also proved that the energy consumption of the attacker with ICARO is reduced compared to the constant attack. Then, we discuss the possibilities for an attacker of available jamming attacks to jam different types of drones without interfering with surrounding communications.

Conclusion 7

In this chapter, we present the results and perspectives of this thesis. In section 7.0.1, we recall the objectives. We then present the results of the work that has been carried out in relation to these objectives. Finally, in section 7.0.2, we set out a set of envisaged perspectives in relation to what has already been achieved.

7.0.1 Contributions

The initial goal of this thesis was to create new intelligent attacks leading to denial-of-services in IoT networks. These last years, machine learning algorithms have solved many problems in many areas. However, this technological advance can be considered a double-edged sword, as malicious people can also benefit from it. Attacks then become more robust, autonomous and harder to detect. In this thesis we show that it is possible to take advantage of the machine learning algorithm to create new intelligent attacks and we demonstrate new vulnerabilities. We also show that smart attacks can be diverted from their primary objective to create more robust defense methods.

The contributions of this thesis can be summarized into several main axes:

1. We have created a new framework, called HARPAGON, capable of modeling the interaction between an attacker and his victim. Based on the theory of Markov chains, HARPAGON, makes it possible to create several types of passive and active attacks on several IoT communication protocols. With real experiments, we show that HARPAGON increases the performance of eavesdropping and jamming attacks while minimizing the power consumption of the attacker. Consequently, we demonstrate that the duty-cycle mechanism can be considered as a threat because we can predict the future slot of transmission with advanced algorithms.
2. We have designed FOLPETTI, another type of framework able to understand and predict the behavior of victims and more particularly their channel hopping strategy. Based on the Multi-Armed Bandit algorithm, FOLPETTI can be coupled to several types of attacks and can predict the future transmission channel of its victim. Through simulations and experimentation on a real testbed, we were able to evaluate different frequency hopping methods, from the simplest strategies to a more advanced strategy also based on MAB.

3. With ICARO attack, we illustrate how a smart attack can be employed in defense method against illicit drones. Indeed, machine learning algorithms can improve the effectiveness of jamming attacks against drones by being more targeted. With real experiences in different environments, we prove that ICARO can target a particular drone while avoiding jamming communications around it.
4. Another contribution is the creation of a jamming module on the NS-3 simulator. We have set up a system that is not only as extensible but also includes new methods of mitigation and jammer strategies. It is now possible to create smarter attacks based on more advanced algorithms such as reinforcement learning in NS-3 simulators. We prove its scalability by developing an intelligent channel hopping method and a jamming attack based on existing works.

The objective of the thesis was to create intelligent attacks to highlight new vulnerabilities in IoT networks and find new detection methods. However, although we have shown that the duty cycle and channel hopping mechanisms can pose threats, we have taken the creative part out of showing that smart attacks can be used as defense method. Indeed, the employment of HARPAGON considerably reduces the energy expenditure of an attacker. This factor is not negligible, especially in the IoT networks. In the case of defense, this model can be useful to obtain datasets for detection methods based on numerous data with little energy consumption. FOLPETTI can be used to improve the monitoring performance of a communication based on several channels. Indeed, nowadays to monitor Bluetooth or WiFi communication using channel hopping, we need as many monitoring devices as channels. In this way, FOLPETTI could considerably reduce this number of devices and increase data monitoring performance. Finally, ICARO is a smart jamming attack designed for defense system. The main goal is to jam a target drone without disrupting the other communication in WiFi communication. ICARO was designed to only jam a specific channel. However, for DJI drones that used channel hopping, it is possible to use the FOLPETTI attack to track channel hopping and jam only on a specific frequency.

7.0.2 Perspectives

We present in the rest of this section perspectives related to the different axes of the thesis.

At the moment, HARPAGON is based on Markov Chain Theory. This process is based on several ML algorithms, consequently, we plan to implement this Markov Chain theory in a reinforcement learning algorithm. One of the objectives will be to make the process more autonomous by

choosing the characteristics of the framework such as the maximum energy cost. Moreover, HARPAGON attack has been evaluated in simple networks, it might be interesting to assess the impact of this smart attack over a large network.

If we combine the logic of the FOLPETTI and ICARO attacks, it would be possible to create a framework that could counter several types of drones. Indeed, it could be very interesting to have a framework able to adapt according to the type of drone that one wishes to attack. For example, if a DJI drone is identified with a proprietary protocol, FOLPETTI can be chosen to jam the drone. However, if it is, a PARROT drone, the ICARO attack can be considered. For the moment it is two types of framework are used to scramble the communication with a drone and its controller, but another possible extension of this type of framework could also be to take control after a disconnection on the drone.

In the same logic, it could be interesting to combine these three frameworks to design an effective jammer against drones. Indeed, HARPAGON can be easily adaptable to the strategy of the drone communication. For example, we have seen that after a disconnection, several types of drones in an indoor environment remain in hover mode for 3 minutes and then land. During these 3 minutes, the jammer must remain in jamming mode to avoid any reconnection but after that it is not obliged to continue its attack. Therefore, we can modify the HARPAGON interaction scheme and combine it with FOLPETTI or ICARO to reduce the energy consumptionof the attacker.

Finally, like any software and simulation system, new functionalities can be permanently integrated in the new NS-3 module. In future works, we will extend this module with additional functionality such another mitigation method. Several more evolved algorithms for detection can be also added later like a ML-based detection algorithm.

Bibliography

[1] Cisco. *Cisco edge to enterprise iot analytics for electric utilities solution overview*. URL: https://www.cisco.com/c/en/us/solutions/collateral/%20data-center-virtualization/big-data/solution-overview-c22-740248 (visited on 06/10/2022) (cited on page 2).

[2] Keyur K Patel, Sunil M Patel, et al. 'Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges'. In: *International journal of engineering science and computing* 6.5 (2016) (cited on page 2).

[3] Vinod Kumar. 'Smart environment for smart cities'. In: *Smart Environment for Smart Cities*. Springer, 2020, pp. 1–53 (cited on page 2).

[4] Partha Pratim Ray. 'Internet of things for smart agriculture: Technologies, practices and future direction'. In: *Journal of Ambient Intelligence and Smart Environments* 9.4 (2017), pp. 395–420 (cited on page 2).

[5] Raksha Ghosh et al. 'Intelligent transportation systems: A survey'. In: *2017 International Conference on Circuits, Controls, and Communications (CCUBE)*. Dec. 2017, pp. 160–165. DOI: 10.1109/CCUBE.2017.8394167 (cited on page 2).

[6] Li Da Xu, Wu He, and Shancang Li. 'Internet of things in industries: A survey'. In: *IEEE Transactions on industrial informatics* 10.4 (2014), pp. 2233–2243 (cited on page 2).

[7] Mrinai M Dhanvijay and Shailaja C Patil. 'Internet of Things: A survey of enabling technologies in healthcare and its applications'. In: *Computer Networks* 153 (2019), pp. 113–131 (cited on page 2).

[8] Serhat Burmaoglu, Ozcan Saritas, and Haydar Yalcin. 'Defense 4.0: Internet of things in military'. In: *Emerging Technologies for Economic Development*. Springer, 2019, pp. 303–320 (cited on page 3).

[9] Ban Al-Omar et al. 'Role of information and communication technologies in the smart grid'. In: *Journal of Emerging Trends in Computing and Information Sciences* 3.5 (2012), pp. 707–716 (cited on page 3).

[10] David E Whitehead et al. 'Ukraine cyber-induced power outage: Analysis and practical mitigation strategies'. In: *2017 70th Annual Conference for Protective Relay Engineers (CPRE)*. IEEE. 2017, pp. 1–8 (cited on page 3).

[11] Bryce Alexander, Sohaib Haseeb, and Adrian Baranchuk. 'Are implanted electronic devices hackable?' In: *Trends in cardiovascular medicine* 29.8 (2019), pp. 476–480 (cited on page 3).

[12] Mahdi Dibaei et al. 'Attacks and defences on intelligent connected vehicles: A survey'. In: *Digital Communications and Networks* 6.4 (2020), pp. 399–421 (cited on page 3).

[13] Dan Dragomir et al. 'A Survey on Secure Communication Protocols for IoT Systems'. In: *2016 International Workshop on Secure Internet of Things (SIoT)*. 2016, pp. 47–62. DOI: 10.1109/SIoT.2016.012 (cited on page 3).

[14] Akshay Kumar Goel et al. 'Attacks, Countermeasures and Security Paradigms in IoT'. In: *2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT)*. Vol. 1. 2019, pp. 875–880. DOI: 10.1109/ICICICT46008.2019.8993338 (cited on page 4).

[15] John Seymour and Philip Tully. 'Weaponizing data science for social engineering: Automated E2E spear phishing on Twitter'. In: *Black Hat USA* 37 (2016), pp. 1–39 (cited on page 4).

[16] 'Deeplocker–concealing targeted attacks with ai locksmithing'. In: () (cited on page 4).

[17] Elias Bou-Harb and Nataliia Neshenko. *Cyber threat intelligence for the internet of things*. Springer, 2020 (cited on page 9).

[18] Saurabh Singh et al. 'Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions'. In: *Journal of Ambient Intelligence and Humanized Computing* (2017), pp. 1–18 (cited on page 10).

[19] Swarup Bhunia et al. 'Protection Against Hardware Trojan Attacks: Towards a Comprehensive Solution'. In: *IEEE Design & Test* 30.3 (2013), pp. 6–17. DOI: 10.1109/MDT.2012.2196252 (cited on page 10).

[20] Meriem Bettayeb, Qassim Nasir, and Manar Abu Talib. 'Firmware Update Attacks and Security for IoT Devices: Survey'. In: *Proceedings of the ArabWIC 6th Annual International Conference Research Track*. ArabWIC 2019. Rabat, Morocco: Association for Computing Machinery, 2019. DOI: 10.1145/3333165.3333169 (cited on page 10).

[21] Ioannis Andrea, Chrysostomos Chrysostomou, and George Hadjichristofi. 'Internet of Things: Security vulnerabilities and challenges'. In: *2015 IEEE symposium on computers and communication (ISCC)*. IEEE. 2015, pp. 180–187 (cited on page 10).

[22] Fatimah Alkhudhayr et al. 'Information security: A review of information security issues and Techniques'. In: *2019 2nd International Conference on Computer Applications & Information Security (ICCAIS)*. IEEE. 2019, pp. 1–6 (cited on page 10).

[23] A. D. Wood and J. A. Stankovic. 'Denial of service in sensor networks'. In: *Computer* 35.10 (Oct. 2002), pp. 54–62. DOI: 10.1109/MC.2002.1039518 (cited on page 11).

[24] Sokratis Kartakis et al. 'Demystifying Low-Power Wide-Area Communications for City IoT Applications'. In: New York, NY, USA: Association for Computing Machinery, 2016. DOI: 10.1145/2980159.2980162 (cited on page 11).

[25] R. Smith. *Assault on California Power Station Raises Alarm on Potential for Terrorism*. URL: https://www.wsj.com/articles/SB10001424052702304851104579359141941621778 (cited on page 11).

[26] Nicolas López et al. 'On the performance of 6LoWPAN using TSCH/Orchestra mode against a jamming attack'. In: *2019 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*. 2019, pp. 1–5. DOI: 10.1109/CHILECON47746.2019.8988035 (cited on page 11).

[27] Cong Pu et al. 'EYES: Mitigating forwarding misbehavior in energy harvesting motivated networks'. In: *Computer Communications* 124 (2018), pp. 17–30. DOI: https://doi.org/10.1016/j.comcom.2018.04.007 (cited on page 12).

[28] Kittipong Chomboon et al. 'An Empirical Study of Distance Metrics for k-Nearest Neighbor Algorithm'. In: Jan. 2015, pp. 280–285. DOI: 10.12792/iciae2015.051 (cited on page 14).

[29] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018 (cited on pages 16, 105).

[30] Miles Brundage et al. 'The malicious use of artificial intelligence: Forecasting, prevention, and mitigation'. In: *arXiv preprint arXiv:1802.07228* (2018) (cited on page 18).

[31] *Machine learning platform: OpenAI-Gym*. https://gym.openai.com/. Last accessed 20 June 2022 (cited on page 18).

[32] *Machine learning platform: Tensorflow*. https://www.tensorflow.org/?hl=fr. Last accessed 20 June 2022 (cited on page 18).

[33] Intel. *FPGA vs. GPU for Deep Learning*. https://www.intel.fr/content/www/fr/fr/artificial-intelligence/programmable/fpga-gpu.html. Last accessed 20 June 2022. 2020 (cited on page 18).

[34] Paul Syverson. 'A taxonomy of replay attacks [cryptographic protocols]'. In: *Proceedings The Computer Security Foundations Workshop VII*. IEEE. 1994, pp. 187–191 (cited on page 21).

[35] Wenyuan Xu et al. 'The feasibility of launching and detecting jamming attacks in wireless networks'. In: *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*. 2005, pp. 46–57 (cited on pages 23, 53, 68).

[36] Aristides Mpitziopoulos et al. 'A survey on jamming attacks and countermeasures in WSNs'. In: *IEEE Communications Surveys & Tutorials* 11.4 (2009), pp. 42–56 (cited on page 24).

[37] Selen Gecgel, Caner Goztepe, and Gunes Karabulut Kurt. 'Jammer detection based on artificial neural networks: A measurement study'. In: *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*. 2019, pp. 43–48 (cited on page 24).

[38] Mohamed A Aref, Sudharman K Jayaweera, and Stephen Machuzak. 'Multi-agent reinforcement learning based cognitive anti-jamming'. In: *2017 IEEE wireless communications and networking conference (WCNC)*. IEEE. 2017, pp. 1–6 (cited on page 24).

[39] Sunakshi Jaitly, Harshit Malhotra, and Bharat Bhushan. 'Security vulnerabilities and countermeasures against jamming attacks in Wireless Sensor Networks: A survey'. In: *2017 International Conference on Computer, Communications and Electronics (Comptelix)*. IEEE. 2017, pp. 559–564 (cited on page 24).

[40] Raymond Pickholtz, Donald Schilling, and Laurence Milstein. 'Theory of spread-spectrum communications-a tutorial'. In: *IEEE transactions on Communications* 30.5 (1982), pp. 855–884 (cited on page 24).

[41] Mohammed M Olama et al. 'Hybrid DS/FFH spread-spectrum: a robust, secure transmission technique for communication in harsh environments'. In: *2011-MILCOM 2011 Military Communications Conference*. IEEE. 2011, pp. 2136–2141 (cited on page 24).

[42] Yusnita Rahayu et al. 'Ultra wideband technology and its applications'. In: *2008 5th IFIP International Conference on Wireless and Optical Communications Networks (WOCN'08)*. IEEE. 2008, pp. 1–5 (cited on page 24).

[43] Gyungmin Kim et al. 'Frame-Selective Wireless Attack Using Deep-Learning-Based Length Prediction'. In: *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 2018, pp. 1–2. DOI: 10.1109/SAHCN.2018.8397145 (cited on pages 25, 28).

[44] Saidhiraj Amuru and R. Michael Buehrer. 'Optimal Jamming using Delayed Learning'. In: *2014 IEEE Military Communications Conference*. 2014, pp. 1528–1533. DOI: 10.1109/MILCOM.2014.252 (cited on pages 25, 28).

[45] SaiDhiraj Amuru et al. 'Jamming Bandits—A Novel Learning Method for Optimal Jamming'. In: *IEEE Transactions on Wireless Communications* 15.4 (2016), pp. 2792–2808. DOI: 10.1109/TWC.2015.2510643 (cited on pages 26, 28).

[46] Shaoshuai ZhuanSun et al. 'A novel jamming strategy-greedy bandit'. In: *2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN)*. 2017, pp. 1142–1146. DOI: 10.1109/ICCSN.2017.8230289 (cited on pages 26–28).

[47] Noah Thurston, Garrett Vanhoy, and Tamal Bose. *INTELLIGENT JAMMING USING DEEP Q-LEARNING*. 2018. URL: http://hdl.handle.net/10150/631658 (cited on pages 26–28).

[48] Chen Zhong et al. 'Adversarial Jamming Attacks on Deep Reinforcement Learning Based Dynamic Multichannel Access'. In: *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. Seoul, Korea (South): IEEE Press, 2020, pp. 1–6. DOI: 10.1109/WCNC45663.2020.9120770 (cited on page 27).

[49] Shaoshuai ZhuanSun, Jun-an Yang, and Hui Liu. 'An algorithm for jamming strategy using OMP and MAB'. In: *EURASIP Journal on Wireless Communications and Networking* 2019.1 (2019), pp. 1–11 (cited on pages 27, 28).

[50] Shaoshuai Zhuansun, Jun-an Yang, and Hui Liu. 'Apprenticeship learning in cognitive jamming'. In: *Optimal Control Applications and Methods* 40.4 (2019), pp. 647–658 (cited on pages 27, 28).

[51] Tugba Erpek, Yalin E. Sagduyu, and Yi Shi. *Deep Learning for Launching and Mitigating Wireless Jamming Attacks*. 2018. DOI: 10.48550/ARXIV.1807.02567. URL: https://arxiv.org/abs/1807.02567 (cited on pages 27, 28).

[52] L. Galluccio, G. Morabito, and S. Palazzo. 'Analytical Evaluation of a Tradeoff between Energy Efficiency and Responsiveness of Neighbor Discovery in Self-Organizing Ad Hoc Networks'. In: *IEEE J.Sel. A. Commun.* 22.7 (2006), pp. 1167–1182. DOI: 10.1109/JSAC.2004.829336 (cited on page 32).

[53] V. Loscri. 'An Analytical Evaluation of a Tradeoff between Power Efficiency and Scheduling Updating Responsiveness in a TDMA Paradigm'. In: QSHINE '07. Vancouver, Canada: Association for Computing Machinery, 2007. DOI: 10.1145/1577222.1577269 (cited on page 32).

[54] Ruben M. Sandoval et al. 'Optimal Policy Derivation for Transmission Duty-Cycle Constrained LPWAN'. In: *IEEE Internet of Things Journal* 5.4 (2018), pp. 3114–3125. DOI: 10.1109/JIOT.2018.2833289 (cited on page 33).

[55] L.M. Feeney and M. Nilsson. 'Investigating the energy consumption of a wireless network interface in an ad hoc networking environment'. In: *Proceedings IEEE INFOCOM 2001. Conference on Computer Communications. Twentieth Annual Joint Conference of the IEEE Computer and Communications Society (Cat. No.01CH37213)*. Vol. 3. 2001, 1548–1557 vol.3. DOI: 10.1109/INFCOM.2001.916651 (cited on page 34).

[56] wolfram. *Home page Mathematica tool*. Last accessed 30 may 2022. 2022. URL: https://www.wolfram.com/mathematica/ (cited on page 40).

[57] Atheros Communications. *Single-Chip 2x2 MIMO MAC/BB/Radio with PCI Express Interface for 802.11n 2.4 and 5 GHz WLANs*. URL: https://datasheetspdf.com/datasheet/AR9280.html (cited on pages 40, 113).

[58] Yi Wang et al. 'IEEE 802.11p performance evaluation and protocol enhancement'. In: *2008 IEEE International Conference on Vehicular Electronics and Safety*. 2008, pp. 317–322. DOI: 10.1109/ICVES.2008.4640898 (cited on page 43).

[59] L. Ophir, Y. Bitran, and I. Sherman. 'Wi-Fi (IEEE 802.11) and Bluetooth coexistence: issues and solutions'. In: *2004 IEEE 15th International Symposium on Personal, Indoor and Mobile Radio Communications (IEEE Cat. No.04TH8754)*. Vol. 2. 2004, 847–852 Vol.2. DOI: 10.1109/PIMRC.2004.1373819 (cited on page 43).

[60] Arash Ahmadfard and Ali Jamshidi. 'A channel hopping based defense method against primary user emulation attack in cognitive radio networks'. In: *Computer Communications* 148 (2019), pp. 1–8. DOI: https://doi.org/10.1016/j.comcom.2019.09.003 (cited on page 43).

[61] Antoine Gallais et al. 'Denial-of-Sleep Attacks against IoT Networks'. In: *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*. 2019, pp. 1025–1030. DOI: 10.1109/CoDIT.2019.8820402 (cited on page 43).

[62] Hiba Dakdouk et al. 'Reinforcement Learning Techniques for Optimized Channel Hopping in IEEE 802.15.4-TSCH Networks'. In: *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. New York, NY, USA: Association for Computing Machinery, 2018, pp. 99–107. DOI: 10.1145/3242102.3242110 (cited on page 45).

[63] Alessio Diamanti, José Manuel Sánchez Vílchez, and Stefano Secci. 'An AI-empowered framework for cross-layer softwarized infrastructure state assessment'. In: *IEEE Transactions on Network and Service Management* (2022), pp. 1–1. DOI: 10.1109/TNSM.2022.3161872 (cited on page 45).

[64] Nathan Franklin Saraiva de Sousa and Christian Esteve Rothenberg. 'CLARA: Closed Loop-based Zero-touch Network Management Framework'. In: *2021 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. 2021, pp. 110–115. DOI: 10.1109/NFV-SDN53031.2021.9665048 (cited on page 45).

[65] Viktor Toldov et al. 'A Thompson sampling approach to channel exploration-exploitation problem in multihop cognitive radio networks'. In: *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. 2016, pp. 1–6. DOI: 10.1109/PIMRC.2016.7794785 (cited on pages 47, 63, 79, 90).

[66] Mujahid Mohsin et al. 'IoTSAT: A formal framework for security analysis of the internet of things (IoT)'. In: *2016 IEEE Conference on Communications and Network Security (CNS)*. 2016, pp. 180–188. DOI: 10.1109/CNS.2016.7860484 (cited on page 48).

[67] Korporn Panyim et al. 'On limited-range strategic/random jamming attacks in wireless ad hoc networks'. In: Nov. 2009, pp. 922–929. DOI: 10.1109/LCN.2009.5355041 (cited on page 53).

[68] Jonathan Villain et al. 'Detection of cyber-attacks on Wi-Fi networks by classification of spectral data'. In: *2020 XXXIIIrd General Assembly and Scientific Symposium of the International Union of Radio Science*. 2020, pp. 1–3. DOI: 10.23919/URSIGASS49373.2020.9232196 (cited on page 54).

[69] Jeroen Wyffels et al. 'Influence of Bluetooth Low Energy on WIFI Communications and Vice Versa'. In: *ECUMICT 2014*. Ed. by Lieven De Strycker. Cham: Springer International Publishing, 2014, pp. 205–216 (cited on page 54).

[70] MT Atlantique. *Representation of the anechoic chamber*. Last accessed 10 june 2022. URL: https://www.imt-atlantique.fr/fr/actualites/une-nouvelle-chambre-anechoique-pour-le-departement-micro-ondes (cited on page 54).

[71] MAthWork. *MATLAB for the Machine Learning*. URL: https://fr.mathworks.com/solutions/machine-learning.html (visited on 06/10/2022) (cited on page 54).

[72] Piotr Gawlowicz and Anatolij Zubow. 'Ns-3 Meets OpenAI Gym: The Playground for Machine Learning in Networking Research'. In: *Proceedings of the 22nd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. MSWIM '19. Miami Beach, FL, USA: Association for Computing Machinery, 2019, pp. 113–120. DOI: 10.1145/3345768.3355908 (cited on page 54).

[73] András Varga and Rudolf Hornig. 'An Overview of the OMNeT++ Simulation Environment'. In: Simutools '08. Marseille, France: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008 (cited on page 55).

[74] Asmae Bengag, Omar Moussaoui, and Mimoun Moussaoui. 'A new IDS for detecting jamming attacks in WBAN'. In: *2019 Third International Conference on Intelligent Computing in Data Sciences (ICDS)*. 2019, pp. 1–5. DOI: 10.1109/ICDS47004.2019.8942268 (cited on page 55).

[75] Sachi D. Babar, Neeli R. Prasad, and Ramjee Prasad. 'Jamming attack: Behavioral modelling and analysis'. In: *Wireless VITAE 2013*. 2013, pp. 1–5. DOI: 10.1109/VITAE.2013.6617054 (cited on page 55).

[76] Chen Zhong et al. 'Adversarial Jamming Attacks on Deep Reinforcement Learning Based Dynamic Multichannel Access'. In: May 2020, pp. 1–6. DOI: 10.1109/WCNC45663.2020.9120770 (cited on page 55).

[77] Mert Eygi and Gunes Karabulut Kurt. 'A Countermeasure against Smart Jamming Attacks on LTE Synchronization Signals'. In: *Journal of Communications* (Jan. 2020), pp. 626–632. DOI: 10.12720/jcm.15.8.626-632 (cited on page 55).

[78] Ivan Martinez, Philippe Tanguy, and Fabienne Nouvel. 'On the performance evaluation of LoRaWAN under Jamming'. In: *2019 12th IFIP Wireless and Mobile Networking Conference (WMNC)*. 2019, pp. 141–145. DOI: 10.23919/WMNC.2019.8881830 (cited on page 56).

[79] Alvaro Diaz and Pablo Sanchez. 'Simulation of Attacks for Security in Wireless Sensor Network'. In: *Sensors* 16.11 (2016). DOI: 10.3390/s16111932 (cited on page 56).

[80] Nsam. *Wave model, url = https://www.nsnam.org/docs/models/html/wave.html*. (Visited on 06/10/2022) (cited on page 58).

[81] Nsam. *Energy Framework model, url =https://www.nsnam.org/docs/models/html/energy.html*. (Visited on 06/10/2022) (cited on page 58).

[82] Nsam. *Old jamming attack module*. URL: https://www.nsnam.org/wiki/Wireless_jamming_model (visited on 06/10/2022) (cited on page 58).

[83] Valeria Loscri Emilie Bout. *A new jamming module for ns-3*. URL: https://github.com/JammingWiFiNs3/JammingWifiModule (visited on 06/10/2022) (cited on page 59).

[84] Valeria Loscri Emilie Bout. *Documentation for the new jamming module on ns-3*. URL: https://ns3-jamming-documentation.herokuapp.com/ (visited on 06/10/2022) (cited on page 59).

[85] Francesco Guala. 'Models, Simulations, and Experiments'. In: *Model-Based Reasoning: Science, Technology, Values*. Ed. by Lorenzo Magnani and Nancy J. Nersessian. Boston, MA: Springer US, 2002, pp. 59–74. DOI: 10.1007/978-1-4615-0605-8_4 (cited on page 77).

[86] Scapy. *Documentation of SCAPY tool*. URL: https://scapy.net/ (visited on 06/10/2022) (cited on page 78).

[87] Hostapd tool. *Documentation of Hostapd tool*. URL: https://en.wikipedia.org/wiki/Hostapd (visited on 06/10/2022) (cited on page 78).

[88] Electron Framework. *Documentation of Electron framework*. URL: https://www.electronjs.org/ (visited on 06/10/2022) (cited on page 79).

[89] Vanhoef Mathy. *Documentation of Modwifi tools*. URL: https://github.com/vanhoefm/modwifi (visited on 06/10/2022) (cited on page 79).

[90] A. Communications. *Single-chip 2x2 mimo mac/bb/radio with pci express interface for 802.11n 2.4 and 5 ghz wlans.* (Cited on page 79).

[91] Michael Spuhler et al. 'Detection of reactive jamming in DSSS-based wireless communications'. In: *IEEE Transactions on Wireless Communications* 13.3 (2014), pp. 1593–1603 (cited on page 81).

[92] Abderrahim Benslimane, Abdelouahid El yakoubi, and Mohammed Bouhorma. 'Analysis of Jamming Effects on IEEE 802.11 Wireless Networks'. In: *2011 IEEE International Conference on Communications (ICC)*. 2011, pp. 1–5. DOI: 10.1109/icc.2011.5962627 (cited on page 83).

[93] Bhawna Ahuja, Deepak Mishra, and Ranjan Bose. 'Optimal Green Hybrid Attacks in Secure IoT'. In: *IEEE Wireless Communications Letters* 9.4 (2020), pp. 457–460. DOI: 10.1109/LWC.2019.2958910 (cited on pages 84, 86).

[94] Zequ Yang, Peng Cheng, and Jiming Chen. 'LearJam: An energy-efficient learning-based jamming attack against low-duty-cycle networks'. In: *2014 IEEE 11th International Conference on Mobile Ad Hoc and Sensor Systems*. IEEE. 2014, pp. 354–362 (cited on pages 85, 86).

[95] Blue Ribbon Task Force on UAS Mitigation at Airports. *Interim Report*. URL: https://uasmitigationatairports.org/wp-content/uploads/2019/07/BRTF-Report-New-2.pdf (visited on 07/10/2022) (cited on page 97).

[96] The EurAsian time. *Russia Uses Electromagnetic Stupor Anti-Drone Weapon To Counter Ukrainian UAVs*. URL: https://eurasiantimes.com/from-fifa-world-cup-to-ukraine-russia-uses-electromagnetic-stupor-anti-drone-weapon-to-counter-ukrainian-uavs/ (visited on 07/10/2022) (cited on page 97).

[97] Parrot. *Parrot WebSite*. URL: https://www.parrot.com (visited on 06/10/2022) (cited on page 98).

[98] DJI. *DJI WebSite*. URL: https://www.dji.com/ (visited on 06/10/2022) (cited on page 98).

[99] M. Hassanalian and A. Abdelkefi. 'Classifications, applications, and design challenges of drones: A review'. In: *Progress in Aerospace Sciences* 91 (2017), pp. 99–131. DOI: https://doi.org/10.1016/j.paerosci.2017.04.003 (cited on page 98).

[100] X-Company. *Loon Expanding internet connectivity wit stratospheric balloons*. URL: https://x.company/projects/loon/ (visited on 06/10/2022) (cited on page 98).

[101] ZipLine Company. *Zipline Project*. URL: https://flyzipline.com/ (visited on 06/10/2022) (cited on page 99).

[102] Amazon Company. *Amazon prime air prepares for drone deliveries*. URL: https://www.aboutamazon.com/news/transportation/amazon-prime-air-prepares-for-drone-deliveries (visited on 06/10/2022) (cited on page 99).

[103] Sharad Kumar Gupta and Dericks P Shukla. 'Application of drone for landslide mapping, dimension estimation and its 3D reconstruction'. In: *Journal of the Indian Society of Remote Sensing* 46.6 (2018), pp. 903–914 (cited on page 99).

[104] Naqqash Dilshad et al. 'Applications and Challenges in Video Surveillance via Drone: A Brief Survey'. In: *2020 International Conference on Information and Communication Technology Convergence (ICTC)*. 2020, pp. 728–732. DOI: 10.1109/ICTC49870.2020.9289536 (cited on page 99).

[105] Jan Henrik Ziegeldorf, Oscar Garcia Morchon, and Klaus Wehrle. 'Privacy in the Internet of Things: threats and challenges'. In: *Security and Communication Networks* 7.12 (2014), pp. 2728–2742.

[106] Kim Hartmann and Keir Giles. 'UAV exploitation: A new domain for cyber power'. In: *2016 8th International Conference on Cyber Conflict (CyCon)*. 2016, pp. 205–221. DOI: 10.1109/CYCON.2016.7529436 (cited on page 100).

[107] John Villasenor. 'Observations from Above: Unmanned Aircraft Systems and Privacy'. In: *Harvard journal of law & public policy* 36 (Mar. 2013), pp. 457–517 (cited on page 100).

[108] Pavel Korshunov and Touradj Ebrahimi. 'Using face morphing to protect privacy'. In: *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*. 2013, pp. 208–213. DOI: 10.1109/AVSS.2013.6636641 (cited on page 100).

[109] Margherita Bonetto et al. 'Privacy in Mini-drone Based Video Surveillance'. In: May 2015. DOI: 10.13140/RG.2.1.4078.5445 (cited on page 100).

[110] Jan Farlik, Miroslav Kratky, and Josef Casar. 'Detectability and jamming of small UAVs by commercially available low-cost means'. In: *2016 International Conference on Communications (COMM)*. 2016, pp. 327–330. DOI: 10.1109/ICComm.2016.7528287 (cited on pages 100, 101).

[111] Ottilia Westerlund and Rameez Asif. 'Drone Hacking with Raspberry-Pi 3 and WiFi Pineapple: Security and Privacy Threats for the Internet-of-Things'. In: *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*. 2019, pp. 1–10. DOI: 10.1109/UVS.2019.8658279 (cited on pages 100, 101).

[112] Wenbo Shen et al. 'Ally Friendly Jamming: How to Jam Your Enemy and Maintain Your Own Wireless Connectivity at the Same Time'. In: *2013 IEEE Symposium on Security and Privacy*. 2013, pp. 174–188. DOI: 10.1109/SP.2013.22 (cited on pages 100, 101).

[113] Pietro Tedeschi, Gabriele Oligeri, and Roberto Di Pietro. 'Leveraging Jamming to Help Drones Complete Their Mission'. In: *IEEE Access* 8 (2020), pp. 5049–5064. DOI: 10.1109/ACCESS.2019.2963105 (cited on page 101).

[114] Karel Pärlin, Muhammad Mahtab Alam, and Yannick Le Moullec. 'Jamming of UAV remote control systems using software defined radio'. In: *2018 International Conference on Military Communications and Information Systems (ICMCIS)*. 2018, pp. 1–6. DOI: 10.1109/ICMCIS.2018.8398711 (cited on page 101).

[115] Jaakko Marin et al. 'Neural networks in the pursuit of invincible counterdrone systems'. In: *IEEE Potentials* 41.1 (2022), pp. 14–21. DOI: 10.1109/MPOT.2021.3113639 (cited on page 101).

[116] Parrot. *ANAFI WHITE PAPER*. URL: https://www.parrot.com/assets/s3fs-public/2020-07/white-paper_anafi-v1.4-en.pdf (visited on 07/10/2022) (cited on pages 102, 103).

[117] Mustafa Ergen. 'IEEE 802.11 Tutorial'. In: *University of California Berkeley* 70 (2002) (cited on page 105).

[118] Volodymyr Kuleshov and Doina Precup. 'Algorithms for multi-armed bandit problems'. In: *arXiv preprint arXiv:1402.6028* (2014) (cited on page 106).

Special Terms

A
AWG Arbitrary Waveform Generator. 101

I
IoT Internet of Things. 1
ITS Intelligent Transportation Systems. 2

M
MAB Multi Armed Bandit. 47
MDP Markov Decision Process. 47

N
NS2 Network Simulator 2. 55
NS3 Network Simulator 3. 56

O
OSI Open Systems Interconnection. 11

P
PDR Packet Delivery Ratio. 24
PER Packet Delivery Ratio. 68

R
RFID Radio Frequency Identification Protocol. 1
RSSI Received Signal Strength Indicator. 67

S
SDR Software-defined radio. 101

T
TCP Transmission Control Protocol. 77

U
UAS Unmanned Aerial Systems. 97
UDP User Datagram Protocol. 26