Un nouveau module pour simuler des attaques de brouillage sur Ns-3

Emilie Bout, Valeria Loscrì

Un nouveau module pour simuler des attaques de brouillage sur Ns-3

Emilie Bout[1] et Valeria Loscri[1]

[1] *Inria Lille Nord-Europe, 40 Avenue Halley, 59650 Villeneuve-d'Ascq, France*

Dû à leur nature, les réseaux sans fils sont vulnérables à de nombreuses attaques et garantir leur sécurité constitue encore un défi. De nos jours, ces réseaux sont utilisés de plus en plus pour assurer la communication entre divers éléments dans des infrastructures critiques comme les hôpitaux. De ce fait, assurer un transfert de données en continu, ainsi que leur exactitude est devenu un point critique en sécurité. Les attaques de brouillage, visant à interférer intentionnellement avec le signal émis par les noeuds légitimes du réseau, peuvent provoquer une perte d'informations jusqu'à la mise hors-service d'un appareil. Cependant, les impacts de ce type d'attaque diffèrent suivant son environnement et ses paramètres d'exécution. Ainsi les répliquer afin de les étudier et de développer des systèmes de détection devient vite chronophage et fastidieux dans la vie réelle. Dans ce papier, nous avons développé un nouveau module complètement gratuit et ouvert à la recherche sur le simulateur "Network Simulator-3" (Ns-3) permettant de simuler les attaques de brouillage. Grâce à ce module, nous montrons qu'il est désormais possible d'implémenter des attaques de brouillage plus élaborées tel que celles basées sur des algorithmes d'apprentissage automatique.

Mots-clefs : Attaque de Brouillage, Science Ouverte, Simulateur, Ns-3

1 Introduction

Jamming attacks attempt to intentionally occupy a channel to prevent a transmission between two nodes and lead to denials of services. This is why it is urgent to evaluate and understand them to effectively counter this type of attack. However, precisely analysing their behaviour in real environments can be tedious and time-consuming. Indeed, this type of attack based on the vulnerabilities of the physical layer produces different behaviors according to a multitude of parameters (e.g. the number of different networks nearby, the distance from its victim). This is why careful studies of jamming attacks require expensive tools like an anechoic chamber. Additionally, the increasing use of machine learning (ML) algorithms in the coming years will lead to a significant change in the threat landscape [BLG21]. Jamming attacks using this technology will become increasingly reactive, more resilient and less identifiable by existing detection methods. In most cases, ML algorithms are data driven, and collecting it in real life takes time. Many works have already attempted to simulate jamming attacks in order to evaluate them and improve security systems. In [BPP13], the authors evaluated the impact of jamming attacks on the ZigBee protocol with the discrete event Ns-2. However, although this simulator is completely free, it is no longer maintained and the module to simulate attacks has never been provided. In [ZWGV20], the MATLAB simulator is employed to create new smart jamming attack based on Deep Reinforcement Learning. However, this simulator is not free and open access, therefore the reproducibility of the results may be impacted. In this context, a free and powerful network simulator such as Network Simulator 3 (Ns-3) could be extremely useful to increase the reproducibility of the results and the evaluation of essential parameters. In this article, we first detail the different strategies of jamming attack and the implementation of this new module. Then, we provide an example of using this new module to create a new smart mitigation method against jamming attacks. This paper is accompanied by the release of the complete source code . [†]

†. https://github.com/JammingWiFiNs3/JammingWifiModule

2 Overview of jamming attacks

Jamming attacks aim to cause a denial of service by degrading the channel's quality and preventing the exchange of packets between legitimate network nodes. A jammer has two main strategic options when executing an attack. The first is to occupy a channel during a specific period. In this situation, all the traffic is interrupted and legitimate nodes of the networks postpone their transmissions. The second is to cause a collision with a packet on transit in order to corrupt it. Two main categories permit to categories jamming attack according to their level of complexity. The first includes all the basic strategies such as constant and reactive jamming attack. The goal of the constant jamming attack is, as their name suggests, to constantly jam a channel. In order to reduce the probability of being detected and the power consumption of the attacker, the reactive attack was created. Its purpose is to jam a channel only when a communication is present on it. In addition, new detection and mitigation methods based on more elaborate methods such as Machine Learning methods have emerged in recent years. To respond to this evolution, the attacker has adapted his strategies and a new category of attack called "Smart Jamming attack" has been developed. This category contains all jamming attacks developed with more elaborate processes such as Machine Learning or Game Theory approaches. These attacks try to solve certain problems such as countering security systems or being effective without having prior knowledge.

3 Implementation of the module

The Ns-3 simulator is a discrete event network simulator for network systems, primarily for research and educational purposes. It is free and open access therefore, it is an ideal tool to extend the reproducibility of research results. To remain light and fast, its architecture is based on a modular system. With this idea, we create a new module to simulate jamming attacks. We based our implementation on an old jamming module that already existed but was no longer compatible with the latest version of Ns-3 [Org22]. We restructured this module and added various features, like new metrics to evaluate jamming attacks. One of the most important points of improvement is the fact that it is now possible to create attacks or defense systems, by embedding machine learning algorithms. This new module is composed of four main components: i) Physical Layer Driver Class, ii) Wireless Utility Class, iii) Jammer Class, iv) Jamming Mitigation Class as illustrated in Fig 1. The first element aims to connect the jamming module with the WiFi-Phy class of Ns-3. The physical layer of a jammer contains particular characteristics (for example the control of its own supply frequency or the creation of a specific type of packets). Therefore, this layer makes it possible to implement all these different functionalities to directly avoid changes in the core of the simulator. The second element provides and calculates different metrics for other components such as the physical layer driver class or the jammer class. The third part constitutes the heart of the module and represents a jammer node. The module code has been designed to make it as extensible as possible. Consequently in Jammer Class, thank an heritage system, it is feasible to create your own jamming strategy by creating a new subClass without modifying the whole code. The last element, Jamming Mitigation Class, has the same logic of the Jammer Class. It represent the implementation of detection and mitigation methods. The jamming module provides a set of essential functions (called APIs) to exploit it.

This module is as extensible as possible and classes are interdependent. In this version, we have implemented several sub-types of jammers such as basic jamming and smart jamming attacks. In the basic part attack, three types of jamming attacks are implemented: constant, random and reactive jamming attacks for the moment. Considering smart jamming attacks, two types of attacks are created in this module. Along the same lines, we have developed several basic and more elaborate strategies for one type of mitigation method: channel hopping.

4 Evaluation and Results

We evaluated this new module with several works, already validated with other simulators like MATLAB or with a real test-bed. A simple network is considered for the rest of the validation system part to avoid side-effect. The network is composed of four nodes: one transmitter, one receiver, one access point and one attacker inter-connected with the 802.11 protocol. One of our first works to validate our module was to

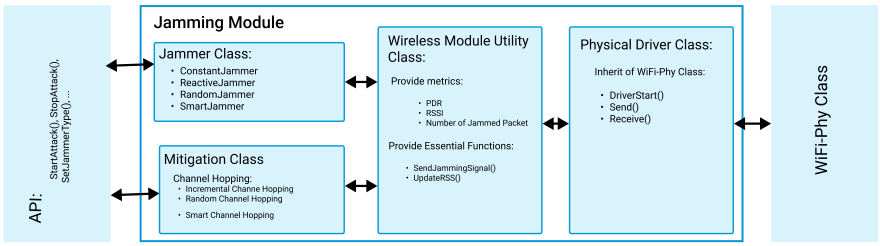
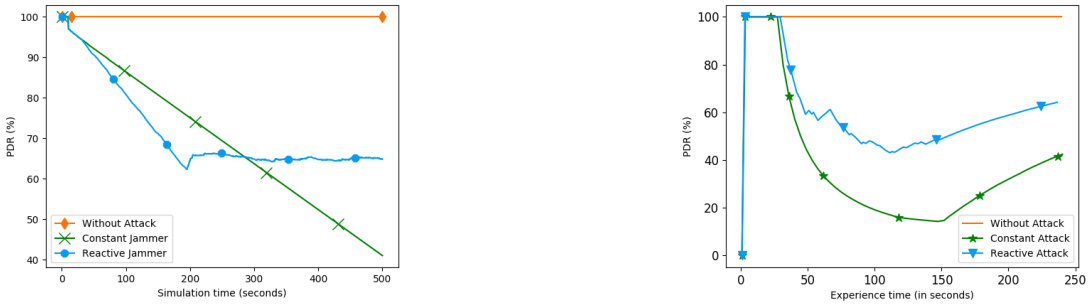


FIGURE 1: NS-3 Jamming Module Architecture



(A) Results obtained in simulated network with Ns-3

(B) Results obtained in a real environment

FIGURE 2: PDR for different types of jamming attacks

compare the impact of basic jamming attacks implemented on a real test bed. We implemented on the test-bed the same network configuration that the simulator. The attacker is equipped with an *Alfa AWUS036h* device and connected to a Raspberry-Pi. We choose this type of instrument to directly modify the MAC layer parameters.

We evaluated the effectiveness of our system with several metrics but here for space reason, we focus on the Packet delivery ratio (PDR). PDR corresponds to ratio between the total of packets correctly received over the total of packets sent. The PDR behavior for the constant qnd reactive jamming attacks is shown in Fig 2. In Fig 2a the results are obtained with our module. We observe that the constant attack constantly degrades the PDR and at the end of the simulation the PDR is equal to 40%. The reactive jamming attack has a strong effect at the start of the attack. However after 200 seconds of the simulation, the PDR varies between 60 and 70%. The same results are obtained in a real environment as demonstrated in Fig 2b. In the case of the real environment, the attack begins after 50 seconds of the experimentation and finished after 150 seconds. For the constant attack the PDR significantly drops after the beginning of the attack. For the reactive attack the same behaviour is observed as with the simulator. Indeed, after few seconds of the simulation the PDR significantly drops. However, after some time, the latter increases again to stabilize around 60-70%. Even if the results obtained with the simulator take longer to be seen than in real environment, the results obtained with our module are close to reality.

Our second analysis concerns the reproducibility of a defense method. Indeed, we decided to implement the same algorithm in [TCLM16] to prove the adaptability of this module to develop a smart mitigation method. In this paper, authors implement a new strategy of channel hopping strategy based on Multi Armed bandit algorithm. Channel hopping consists of dynamically changing the communication channel of legitimate nodes to mitigate the impact of jamming attack. As for jamming attacks, several strategies exist in the literature, but more and more works try to develop more elaborate methods based on machine

learning approaches. Indeed, in this paper, the transmitters use a multi-armed bandit algorithm to converge as quickly as possible on the optimal channel (the least noisy channel). The authors model the channel selection dilemma with a Thompson sampling based approach and compare it with a e-greedy technique. They first analyze their model theoretically in the MATLAB simulator and then on a real test-bed. Their measurements show that Thompson sampling algorithm reaches 99% of the oracle throughput after 26 seconds approx. 57% faster than the next best algorithm (e-greedy) which achieves the same throughput in 60 seconds.

In order to compare our module, we implemented these two smart channel hopping methods. The noise is generated by a constant jamming node which has the ability to randomly select a target channel.

Fig 3 represents the throughput for different algorithms of channel hopping according to the time. The closer the throughput is to 1.0, the less effect the attack has on the network. The first observation that we can make is that the same behaviour of the network is observed during the real experimentation or the simulation. Indeed, in these two case, we see that the Multi Armed Bandit algorithm combined with the Thompson sampling algorithm policy converges faster to the optimal solution than the e-greedy method. Indeed, the Thompson Sampling policy reaches 87% after 22.1 seconds of the simulation. The same rate of the throughput is achieved at 47.3 seconds for the e-greedy policy. Consequently the Thompson Sampling policy is 53% faster than the e-greedy solution. In the reference work, the authors show that their approach is 57% faster than the algorithm based on e-greedy. The results obtained with the simulator are close to a realistic case.

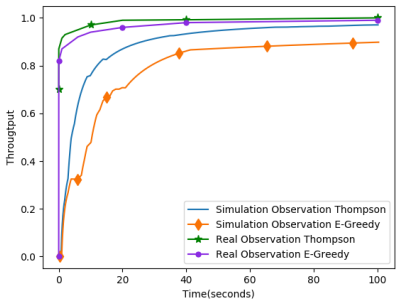


FIGURE 3: Throughput comparison on 10 accessible channels ($\mu = 0.80$)

5 Conclusion

In this paper, we introduce a jamming module for the ns-3 simulator. The module has been designed to be extensible in a very flexible way. We prove its scalability by developing an intelligent channel hopping method based on existing work. We have proven that this module can be interesting for evaluating jamming attacks in networks and gives a first overview of their impact. It can save members of the scientific community time and money. In addition, it can be useful to give a first idea on certain delicate parameters such as energy consumption. In future work, we will extend this module with additional functionalities like another mitigation method: the attenuation transmit power .

References

[BLG21] Emilie Bout, Valeria Loscri, and Antoine Gallais. How machine learning changes the nature of cyberattacks on iot networks: A survey. *IEEE Communications Surveys Tutorials*, pages 1–1, 2021.

[BPP13] Sachi D. Babar, Neeli R. Prasad, and Ramjee Prasad. Jamming attack: Behavioral modelling and analysis. In *Wireless VITAE 2013*, pages 1–5, 2013.

[Org22] NS-3 Organization. Old jamming module, 2022.

[TCLM16] Viktor Toldov, Laurent Clavier, Valeria Loscrí, and Nathalie Mitton. 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)*, pages 1–6, 2016.

[ZWGV20] Chen Zhong, Feng Wang, M. Cenk Gursoy, and Senem Velipasalar. Adversarial jamming attacks on deep reinforcement learning based dynamic multichannel access. pages 1–6, 05 2020.