INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Exploração de Covert Channels de Rede sobre comunicações IEEE 802.15.4

JOÃO TOMÁS BAPTISTA RODRIGUES
julho de 2022

POLITÉCNICO
DO PORTO

Exploring Network Covert Channels over IEEE 802.15.4 communications

João Tomás Baptista Rodrigues

A dissertation submitted in partial fulfillment of the requirements for the degree of Master of Science, Specialisation Area of Software Engineering

Supervisor: Dr. Luís Lino Ferreira
Co-Supervisor: Dr. Ricardo Severino

Porto, July 4, 2022

Dedicatory

To my friends and family, for the motivation and patience they gave me and that allowed me to carry on this work.

Abstract

The advancements in information and communication technology in the past decades have been converging into a new communication paradigm in which everything is expected to be interconnected with the heightened pervasiveness and ubiquity of the Internet of Things (IoT) paradigm. As these technologies mature, they are increasingly finding its way into more sensitive domains, such as Medical and Industrial IoT, in which safety and cyber-security are paramount.

While the number of deployed IoT devices continues to increase annually, up to tens of billions of connected devices, IoT devices continue to present severe cyber-security vulnerabilities, which are worsened by challenges such as scalability, heterogeneity, and their often scarce computing capacity.

Network covert channels are increasingly being used to support malware with stealthy behaviours, aiming at exfiltrating data or to orchestrate nodes of a botnet in a cloaked fashion. Nevertheless, the attention to this problem regarding underlying and pervasive IoT protocols such as the IEEE 802.15.4 has been scarce.

Therefore, in this Thesis, we aim at analysing the performance and feasibility of such covert-channel implementations upon the IEEE 802.15.4 protocol to support the development of new mechanisms and add-ons that can effectively contribute to improve the current state-of-art of IoT systems which rely on such, or similar underlying communication technologies.

Keywords: IoT, Covert-Channel, IEEE 802.15.4, MAC, Timing, OMNeT++

Resumo

Os avanços nas tecnologias de informação e comunicação nas últimas décadas têm convergido num novo paradigma de comunicação, onde se espera que todos os intervenientes estejam interconectados pela ubiquidade do paradigma da *Internet of Things* (Internet das Coisas). Com a maturação destas tecnologias, elas têm-se vindo a infiltrar em domínios cada vez mais sensíveis, como nas aplicações médicas e industriais, onde a confiabilidade da informação e cyber-segurança são um fator crítico.

Num contexto onde o número de dispositivos IoT continua a aumentar anualmente, já na ordem das dezenas de biliões de dispositivos interconectados, estes continuam, contudo, a apresentar severas vulnerabilidades no campo da cyber-segurança, sendo que os desafios como a escalabilidade, heterogeneidade e, na maioria das vezes, a sua baixa capacidade de processamento, tornam ainda mais complexa a sua resolução de forma permanente.

Os *covert channels* de rede são cada vez mais um meio de suporte a *malwares* que apresentam comportamentos furtivos, almejando a extração de informação sensível ou a orquestração de nós de uma *botnet* de uma forma camuflada. Contudo, a atenção dada a este problema em protocolos de rede IoT abrangentes como o IEEE 802.15.4, tem sido escassa.

Portanto, nesta tese, pretende-se elaborar uma análise da *performance* e da viabilidade da implementação de *covert channels* em modelos de rede onde figura o protocolo IEEE 802.15.4 de forma a suportar o desenvolvimento de novos mecanismos e complementos que podem efetivamente contribuir para melhorar a ciber-segurança de sistemas IoT que dependem do suporte destas tecnologias de comunicação.

Acknowledgement

Firstly, I would like to start by thanking Instituto Superior de Engenharia do Porto (ISEP) and all the professors that passed their knowledge to me during the last five years. For Dr. Ricardo Severino a major thanks for all the guidance provided in the years where he lead me into several investigation projects, including the works performed here, works that rendered me a giant source of knowledge and allow me to develop my professional career.

For PORTIC, a thanks to all my colleagues that showed support initiative and kept assuring for the well being of my project as well as my own. Their support while I entered a new project in a new environment was fundamental for the final success result.

To all my friends in ISEP, for distracting and helping me when I need it. I hope I did the same when it was needed. I need to thank particularly Catarina Fernandes, Henrique Ribeiro, Rita Fernandes and Rui Machado for the countless hours working together and helping me achieve my full potential. Without their support and encouragement, it would have been impossible to finish this project.

For my friends outside of the study circle, thanks for the support provided when it was most needed, the moral strength given and for just being there when I needed someone to talk to.

Lastly, to my whole family, especially my parents and my sister. It is always a little difficult to consolidate work, studies, and social life, and I know that without their support and life lessons it would be impossible to finish this stage of my life. For that and everything leading to this project, you have my eternal gratitude.

Contents

List of Figures

List of Tables

List of Algorithms

List of Symbols

α	Delay time	s

List of Acronyms

ACT	Allocation Counter Table.
AES	Advanced Encryption Standard.
AHP	Analytic Hierarchy Process.
AMCA	Asynchronous Multi-Channel Adaptation.
ASN	Absolute Slot Number.
BI	Beacon Interval.
BLINK	Radio Frequency Identification Blink.
BO	Beacon Order.
CAN	Controller Area Network.
CAP	Contention Access Period.
CFP	Contention Free Period.
CI	Consistency Index.
CPS	Cyber Physical Systems.
CPU	Central Process Unit.
CR	Consistency Ratio.
CSL	Coordinated Sampled Listening.
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance.
CSV	Comma Separated Values.
D-DoS	Distributed Denial of Service.
DSME	Deterministic Synchronous Multi-channel Extension.
DSSS	Direct Spread Spectrum Sequence.
ECU	Engine Control Unit.
ETFA	Emerging Technologies and Factory Automation.
FFD	Fully Function Devices.
FTP	File Transfer Protocol.
GACK	Group Acknowledgement.
GTS	Guaranteed Time Slot.
IAT	Inter Arrival Time.
IE	Information Elements.
IEEE	Institute of Electrical and Electronics Engineers.

IEEE-SA	Institute of Electrical and Electronics Engineers Standards Association.
IIoT	Industrial Internet of Things.
IoT	Internet of Things.
KB	KiloByte.
KBPS	KiloBytes per Second.
LBNP	L-Bits to N-Packets.
LIFS	Large Inter Frame Spacing.
LLDN	Low Latency Deterministic Network.
LQI	Link Quality Indication.
LTE	Long Term Evolution.
MAC	Medium Access Control.
MCPS	MAC Common Part Sublayer.
MD	Multi-superframe Duration.
MITM	Man In The Middle.
MLME	MAC Sublayer Management Entity.
MO	Multi-superframe Order.
ns-3	Network Simulator 3.
OMNeT++	Objective Modular Network Testbed in C++.
OSI	Open System Interconnection.
PL	Packet Length.
PORTIC	Porto Research, Technology and Innovation Center.
QoS	Quality of Service.
RFD	Reduced Function Devices.
RI	Random Index.
RIT	Receiver Initiated Transmissions.
SAP	Service Access Point.
SD	Superframe Duration.
SIFS	Small Inter Frame Spacing.
SME	Small and Medium-sized Enterprises.
SO	Superframe Order.
STOIC	Secure Trustworthy Omnipresent Cyber-defenses.
TACAN	Transmitter Authentication in CAN.
TGR	Traffic Generation Rate.
TLS	Transport Layer Security.
TR	Time Replay.
TSCH	Time Slotted Channel Hopping.

Wi-Fi	Wireless Fidelity.
WPAN	Wireless Personal Area Network.
WSN	Wireless Sensor Network.

List of Formulas

Chapter 1

Introduction

1.1 Overview

The advancements in information and communication technology in the past decades have been converging into a new communication paradigm in which everything is expected to be interconnected with the heightened pervasiveness and ubiquity of the Internet of Things (IoT) paradigm. The IoT, a collection of semi-autonomous, Internet connected devices comprised of computing, networking, sensing, and actuation capabilities, interconnected with the physical world [U.S. Department of Defense 2016], encompasses a myriad of several different technologies like Wireless Sensor Networks, Radio-Frequency Identification, and Machine-to-Machine communications, and now, much more than a buzzword, is becoming a pervasive reality, enabling applications in multiple domains such as medical care [Al-Turjman, Nawaz, and Ulusar 2020], agriculture [Sinha, Shrivastava, and Kumar 2019], supply chains [Muñuzuri et al. 2020], transportation [Wang 2020] and smart cities [K. Liu, Bi, and D. Liu 2020].

As these technologies mature, they are increasingly finding its way into the industrial domain, to support what is now dubbed as the Industry 4.0, converging IoT, Cyber Physical Systems (CPS) and Cloud technologies into the factory floor. Industry 4.0 relates to the exponentially technological development that constantly outputs new ways of connecting devices and systems, that allows for new data insights, customisable products, and technological autonomy. Since its introduction it has been used to describe a massive digital transformation of manufacturing, where interconnected processes and equipment allow a mass-customisation of products and a faster response to the market [Masood and Sonntag 2020]. Researchers agree that digitisation, digital transformation, and Industry 4.0 initiatives are vital for companies' future competitiveness and their success in living up to customer expectations. This is especially important for SMEs as these companies make up for 99% of the companies in Europe.

While the number of deployed IoT devices continues to increase annually and is estimated to reach 75 billion by 2025 [Statista 2022] (figure 1.1), the amount of interconnected devices per network will exponentially increase as well.

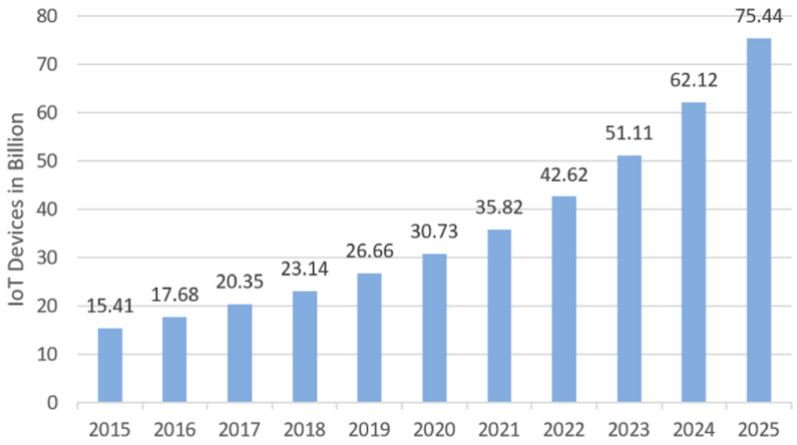


Figure 1.1: IoT Devices Statistics by 2025 [Statista 2022]

IoT verticals present a severe set of challenges, enhanced by the IoT scale, heterogeneity, and its fast adoption. In addition, this trend, despite opening exciting opportunities, also unlocks a variety of new security threats [J. Lin et al. 2017, Frustaci et al. 2018, Lu et al. 2019], with heightened security and privacy risks, particularly in industrial and medical scenarios [Z. Liu et al. 2020].

Unfortunately, most of these Internet connected IoT devices do not have the same experience induced resilience to intrusion, hacking and sabotage attacks that other computing devices have acquired [Caviglione, Merlo, and Migliardi 2018]. On the contrary, they show a significant level of vulnerability. With 70% of IoT devices found to have serious security vulnerabilities, such as unencrypted network services, weak password requirements (figure 1.2), and since 90% of devices collecting personal information [Rawlinson 2014], there is a critical need for improving IoT security approaches. The need is further exacerbated as bad actors exploit this weakness to conduct attacks against IoT infrastructure [Larson 2017, Stanislav and Beardsley 2015, Simon 2016], even taking down large swaths of the Internet by leveraging D-DoS attacks such as the Mirai botnet [Krebs 2017, Kolias et al. 2017], which relied upon illegitimate usage of 400 000 IoT devices.

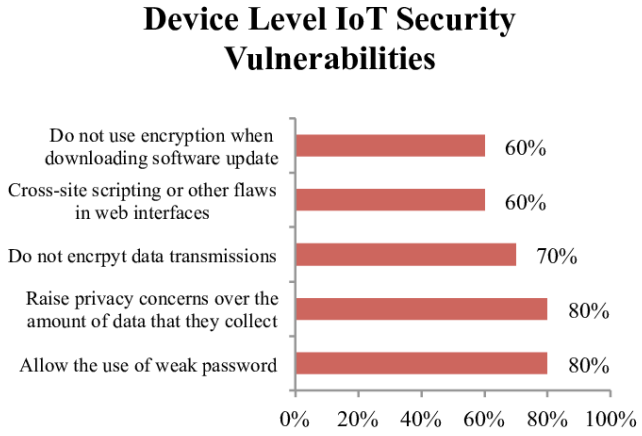


Figure 1.2: IoT devices vulnerabilities [Gamundani, Phillips, and Muyingi 2018]

IoT threats stem both from the intrinsic enhanced vulnerability of a system that is newly connected to other systems (e.g., legacy systems connected to Internet gateways), and from the incredible number of intrusion targets introduced by the IoT paradigm (e.g., home connected appliances, wearables, connected vehicles, etc.). Superimposed to this scenario, we have to take into account that each compromised system may be used for at least three different malicious goals, for instance, regarding sensors and appliances deployed in Industry 4.0, devices may be used for exfiltrating sensitive data (e.g., to steal secrets and perform industrial espionage), to physically endanger a plant (e.g., by saturating the processing power of nodes controlling machineries) or to mount an attack to more critical parts of the same plant or even to a completely different plant (e.g., enrolling the compromised node into a remotely controlled botnet) [Caviglione, Merlo, and Migliardi 2018].

The popular solution to these concerns has been the introduction of cryptographic techniques to support data encryption and secure authentication. However, traditional cryptography methods for network security pose important and significant challenges. On the one hand, IoT "things" are usually equipped with limited resources in terms of energy consumption, memory capacity and computational power [Cirani, Ferrari, and Veltri 2013]. Such limitations hinder the direct implantation of conventional Internet security techniques, like AES or TLS, into many IoT solutions [Riahi Sfar et al. 2018, Kim, Choi, and Hong 2017], and their absence may lead to various security and privacy attacks like eavesdropping, network side-channel attacks, and tracking. On the other hand, encryption cannot solve all security problems in IoT systems, as for instance, if an IoT node establishes covert communication with another device without being detected by an adversary, encryption is insufficient to prevent eavesdropping [Hu, C. Lin, and X. Li 2016]. Moreover, even if a message is encrypted, the metadata, such as the pattern of network traffic, can reveal sensitive information. However, if the adversary cannot detect the transmission, it has no opportunity to launch an "eavesdropping and decoding" attack even if possessing unlimited resources or capabilities to mount quantum attacks [Z. Liu et al. 2020].

In such a troublesome scenario, particularly when physical access to the IoT infrastructure is possible, a very important element in any attack is thus represented by the capability of the attacker to exploit a covert channel and information hiding techniques. In fact, no matter if the goal is exfiltration of critical information or if it is to induce unintended behavior of a node, there is the need for communicating with the compromised node without disclosing the fact that it has been compromised. Indeed, network covert channels are increasingly being used to support malware with stealthy behaviours (stegomalware), for instance to exfiltrate data or to orchestrate nodes of a botnet in a cloaked fashion [Caviglione 2021]. However, the detection of such attacks is difficult as it is unknown in advance where the secret information has been hidden, and on the other hand, network covert channels usually feature low data-rates which difficults the detection. Also, neutralization or mitigation is not straightforward, as it is hard not to disrupt legitimate flows or degrade the quality of service, particularly at the perception layer of an IIoT application. Consequently, countermeasures are tightly coupled to specific channel architectures, leading to poorly generalized and often scarcely scalable approaches.

For all these reasons, in this Thesis, we argue that there is an extreme need for:

1. Analysing the performance of covert channel attacks.
2. Devising novel techniques that can leverage the usage of such techniques in innovative ways, for instance, to devise new information security mechanisms.

Although there are several IoT enabling communication architectures that can help in achieving an energy efficient industrial communications, the communication requirements of these time critical processes demand improved Quality of Service (QoS) in terms of reliability, timeliness and robustness. These stringent requirements on the communication protocols have been increasingly supported by IEEE 802.15.4 [IEEE 2020], to address the overgrowing demands for low-power, low-range, and robust wireless communication.

The Institute of Electrical and Electronics Engineers Standards Association (IEEE-SA) published the IEEE 802.15.4e amendment during the fall of 2012 [IEEE 2012], aiming at enhancing and extending the functionalities of the IEEE 802.15.4-2011 protocol. The enhancements consist of several MAC behaviors, which besides providing deterministic communication, are also designed to support multi-channel frequency hopping mechanisms, such as in the case of the Deterministic and Synchronous Multichannel Extension (DSME) and Time Slotted Channel Hopping (TSCH). There are also other MAC behaviors like the Low Latency Deterministic Network (LLDN), which uses Time Division Multiple Access (TDMA) to provide timing guarantees. DSME and TSCH were incorporated into the revised version IEEE 802.15.4-2020 [IEEE 2020].

Unfortunately, there has been no serious analysis of the covert channel vulnerabilities of this communication protocol. On the other hand, the few analysis of such subject in previous versions of the protocol [Martins and Guyennet 2010, Nain and Rajalakshmi 2017, Tuptuk and Hailes 2015] only focused upon stenography or storage-based covert channels, not addressing a fundamental threat of network timing channels. Moreover, there is no simulation model available that encompasses such implementations, which is fundamental for network designers and security experts to evaluate risks, and to further support the development of detection and mitigation strategies.

Therefore, in this Thesis, we aim at analysing the performance and feasibility of such covert channel implementations upon the IEEE 802.15.4 protocol to support the development of new mechanisms and add-ons that can effectively contribute to improve the current state-of-art of IoT systems which rely on such, or similar underlying communication technologies. In order to achieve this, this Thesis starts by providing the cyber-security and IoT communities with a simulation model, featuring several covert channel implementations, that can be easily deployed and analysed. This implementation is then used as a baseline to evaluate the performance of different covert-channel techniques in respect to different networking settings. This knowledge will be fundamental to devise new mechanisms that leverage these hidden protocol features to implement innovative information security ideas.

1.2 Research Context

This Thesis was carried out in alignment with the STOIC (Secure Trustworthy Omnipresent Cyber-defenses), Research framework and supported by the CybersSecIP project, at the

Porto Research, Technology and Innovation Center (PORTIC) of the Polytechnic Institute of Porto (P.PORTO). The STOIC framework aims at developing innovative mechanisms and technologies for achieving improved cyber-security in distributed wireless infrastructures. This research is tightly connected with the work being pursued in the CybersSecIP project, coordinated by P.PORTO, which aims at further strengthening the scientific competences and innovation potential of the North region of the country, to tackle the cyber-security challenge, through investment in a small set of enabling technologies and knowledge, in a coherent program organized in two research lines: one related to the design and protection of secure digital systems, and a second centered on data security and privacy.

1.3 Research Objectives

The Research Objectives aimed for this Thesis are as follows:

1. Understand the fundamentals of IoT communications, major IoT cyber-security threats and their impact in such networks, particularly the importance of covert channels.
2. Realize and present an argumentation about the severity of the problem addressed in the proposal in regards to IoT cyber-security.
3. Survey covert channel techniques and overview current state of the art. Compare different approaches.
4. Develop relevant metrics for the performance evaluation and comparison of different covert channel techniques.
5. Overview the IEEE 802.15.4-2020 standard and investigate new covert channel opportunities to exploit in the protocol.
6. Setup a simulation platform for the implementation and evaluation of the covert channel techniques.
7. Analyse and evaluate the performance of the selected covert channel techniques in the IEEE 802.15.4.
8. Write and publish a research paper featuring part of the results.

1.4 Research Contributions

The contributions of this Thesis are two-fold:

1. Implementation of several covert channel techniques on a IEEE 802.15.4 simulation model, and its public release for the cyber-security and IoT communities.
2. Performance analysis of different covert channel techniques for the IEEE 802.15.4 and evaluation of the impact of its network settings upon the covert channels.
    - Article "Exploring Timing Covert Channel Performance over the IEEE 802.15.4" [Severino, Rodrigues, and Ferreira 2022] submitted and accepted at the 27th International Conference on Emerging Technologies and Factory Automation (ETFA).

1.5 Thesis Structure

This thesis is composed of an introduction (chapter 1), where a theoretical introduction to the subject is presented, accompanied by the problem in hands, and how this Thesis addresses the mentioned security concerns. This is followed by an overview of the IEEE 802.15.4 communication protocol in Chapter 2, including a segment of its history and an explanation of the newly added functionalities. Next, it provides a research background, where covert channels as a concept are introduced, along with their history and relevant research works (chapter 3), particularly those which focus the usage of covert channels in the IEEE 802.15.4 protocol. Next, this Thesis introduces the network covert channel simulation model used in this Thesis (chapter 4). This chapter presents a value analysis, which includes a thorough analysis of several simulation frameworks. The final architecture of the solution is then explained together with possible design alternatives. Finally, we describe in this chapter the implementation used to carryout the performance analysis of several timing covert channels techniques in the IEEE 802.15.4. This performance evaluation is detailed in chapter 5). In here, a brief description of the simulation setup appears, along with the obtained results and its analysis, featuring several observations and conclusions. Finally, in chapter 6, several general conclusions are drawn from the obtained results, and a few ideas are proposed for future work.

Chapter 2

Overview of the IEEE 802.15.4 Protocol

2.1 General Description

Wireless sensor networks are a fundamental technology to support low-cost, unattended monitoring of a wide range of environments [Baronti et al. 2007], including IIoT. One of the main protocols established as a standard for the physical and MAC sub-layers is the IEEE 802.15.4, due to its flexibiity in adapting to different quality-of-service requirements.

2.1.1 IEEE 802.15.4-2003

Firstly created in 2003, IEEE 802.15.4 was designed for a low-data rate, low power, low-complexity, low cost and short range radio frequency protocol of wireless personal area networks (WPAN) [Alkama and Bouallouche-Medjkoune 2021]. This WPAN will feature a number of devices that can be either Fully Function Devices (FFD) or Reduced Function Devices (RFD) [Kurunathan 2021]. The FFD type devices implement the full protocol set and act as a network coordinator. A great example of a device of the FFD type is the network's PAN coordinator. This device has the responsibility of controlling the additional devices of the network, as well as time synchronizing them all between each other to guarantee a smooth packet transaction [Bensky 2019, Farahani 2008]. The network's end devices are usually RFDs. They are intended for application that are extremely simple, such as a sensor gathering information and are not capable of routing packages.

As said previously, the IEEE 802.15.4 protocol is responsible for specifying two OSI layers: the physical layer and the MAC sublayer (figure 2.1).

Physical Layer

Regarding the first one, the protocol transmits in waves of three possible frequencies: 2.4 GHz (with 16 channels); 915 MHz (with 10 Channels) and 868 MHz (with only one channel) [Kurunathan 2021]. The mainly used one, 2.4GHz, is able to transmit information at a data rate of 250 Kbps.

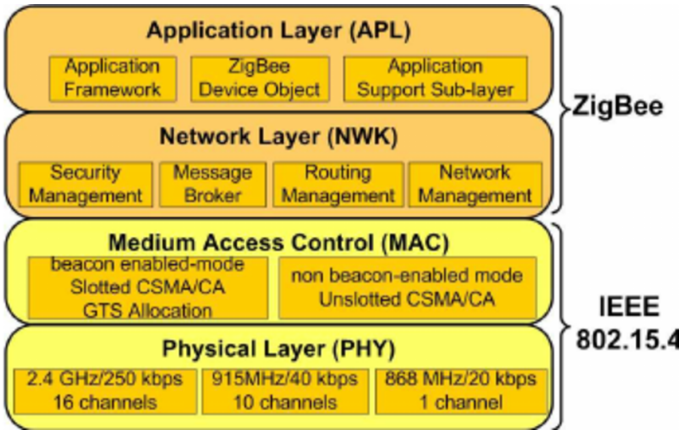


Figure 2.1: IEEE 802.15.4 Layers [Cunha et al. 2007]

MAC Sublayer

The MAC sublayer, where most of our contributions are focused, is composed by two major components: a beacon and a superframe. The beacon is a periodic signal emitted by the PAN coordinator of the network that enables the synchronization of the nodes and is the only broadcasted message in the whole network. A superframe consists of a Contention Access Period (CAP) and of a Contention Free Period (CFP). During the CAP, a CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) contention algorithm is implemented, allowing for the nodes in the network (apart from the PAN coordinator) to compete in order to allocate slots in the CFP [Valero, Bourgeois, and Beyah 2010]. In this second one, there is a division of the superframe into Guaranteed Time Slots (GTSs). In each of this time slots, a node can allocate it to send a message to another one, and when its time arrives, that superframe slot will be responsible for the transmission of the packet (or set of packets) from the node allocating to its destination. An entire superframe is composed of 16 time slots, from which 7 are reserved for the CFP and the rest are allocated to the CSMA/CA integration in the CAP.

2.1.2 IEEE 802.15.4e (enhanced)

In 2012, an amendment [IEEE 2012] was proposed to the legacy IEEE 802.15.4 standard, to satisfy the requirements of emerging IoT applications. The goal was to define a low-power multi-hop MAC protocol, capable of addressing the emerging needs of embedded (industrial) applications [De Guglielmo, Brienza, and Anastasi 2016].

In the MAC layer (where all the upgrades performed in this amendment were realized), five new MAC behaviours were implemented in order to improve its flexibility in accommodating different kinds of application requirements [De Guglielmo, Brienza, and Anastasi 2016]. They are:

- Deterministic Synchronous Multi-channel Extension (DSME): aimed to support applications with requirements in terms of timeliness and reliability
- Time Slotted Channel Hopping (TSCH): that targets industrial automation and process control and supports multi-hop and multi-channel communications

- Radio Frequency Identification Blink (BLINK): intended for item/people identification, location and tracking and its packets are generally sent by "transmit only" devices through the Aloha protocol
- Asynchronous Multi-Channel Adaptation (AMCA): ideal for large deployments, such as smart utility networks
- Low Latency Deterministic Network (LLDN): intended for factory automation, where a critical requirement is the low-latency of network transmissions

Both the DSME and the TSCH behaviours will be explained further along, since they will be further studied in this Thesis.

The amendment in discussion, besides the new MAC behaviours, also brought in some enhancements:

Multi-channel access

In the legacy IEEE 802.15.4, in each time slot, a node was able to communicate by sending a packet in the slot in itself. But, in this slot, a single allocation could be performed, that being, several packets could be passed, but only from node A to B. In the enhanced proposition a multi-channel access was introduced, allowing, in a single time slot, several node allocations in multiple channels. The nodes are given the capability to access the channel either through channel hopping mechanisms or channel adaptation mechanisms. With channel hopping, the sequence is statically predetermined in advance. Alternatively, with channel adaptation, the PAN Coordinator has the ability to allocate different channels for data transmission [Kurunathan 2021].

Information Elements (IE)

Although in the existent IEEE 802.15.4, Information Elements is already a defined and present concept, the enhanced module presents itself with some additional features, such as some unique IE to support the new MAC behaviours, i.e. the DSME behaviour, in which the IE contains information regarding the superframe specification, such as the number of superframes in a multi-superframe, number of channels, time synchronization specification, Group Acknowledgement and channel hopping specifications [Kurunathan 2021].

Low latency and low energy

The newly enhanced protocol has presented itself with some features regarding low latency communications, more suitable for industrial control applications and low energy consumption, providing a trade-off between latency and energy efficiency. The IEEE 802.15.4e allows devices to operate at a very low duty cycle and also provides deterministic latency, which is a main requirement for time-critical applications. This new amendment introduces two low energy mechanisms based on the latency requirements of the applications. On one side, we have the Coordinated Sampled Listening (CSL) that is mostly applied in contexts where a very low latency is imperative. On the other side, there is the Receiver Initiated Transmissions (RIT) mechanisms, that are used for latency-tolerant applications. [Kurunathan 2021].

Multi-purpose frames

All the new MAC behaviours introduced in the amendment of the IEEE 802.15.4 protocol are composed by different frame formats, all designed to better accommodate the specific needs of each behaviour. For example, the DSME variant frame formats are designed to support applications where determinism and scalability are fundamental. This frame variant referred thus supports guaranteed time-slots with multi-channel capability. It also contains the ability to send several acknowledgements in a group (Group Acknowledgements (GACK)) in order to reduce the overall delay for several GTS based transmissions [Kurunathan 2021].

Enhanced Beacons

As the name indicates, the Enhanced Beacons are an extended version of the standard beacons present in the legacy IEEE 802.15.4 protocol. These new beacons contain application-specific content to the specified MAC behaviours, in this case, the DSME and the TSCH. They contain information regarding the status of the behaviour and if the low-energy mode is activated and about the respective channel hopping sequence [Kurunathan 2021].

MAC performance metrics

The new enhanced protocol proposition introduces an evaluation feature to provide information on the link performance to help the network layer in its routing decisions, performing more efficiently. The information transmitted includes the number of transmitted packets that required retries before acknowledgement, number of frames that did not release an acknowledgement, number of packets properly acknowledged and number of received frames that were discarded due to being considered insecure [Kurunathan 2021].

Fast Association

In the legacy IEEE 802.15.4, a device association to a network must be delayed so that the device waits until the end of the MAC response wait time to request the association data from the PAN Coordinator. In the new amendment, the Fast Association mechanism is introduced. With this, the delay is removed from the connection, being that, if the resources are available, the PAN Coordinator instantly allocates a short address to the device, followed by the emission of an association response, containing the new device short address and the status of the association [Kurunathan 2021].

Group Acknowledgement

Particular to the new DSME and LLDN MAC behaviours, this new feature of the enhanced protocol allows for the grouping of several successful transmissions acknowledgements into a single Group Acknowledgement (GACK) either within the adjacent beacon interval or as a separate frame. The PAN Coordinator can also, for a single time-slot, assign a dedicated GACK that will deal with that time-slot exclusively. With this, the number of single acknowledgements travelling the network will significantly diminish and, therefore, it will greatly improve the efficiency of the network [Kurunathan 2021].

2.1.3 IEEE 802.15.4-2020

Since 2003, this protocol has suffered many amendments and has had several versions, with the newer one being released in 2020 [IEEE 2020]. This new release of the IEEE 802.15.4 encompasses all the propositions in the enhanced amendment. According to [Choudhury et al. 2021], one of the principal upgrades was, with the DSME and TSCH MAC behaviours, the ability to overcome the limit of 7 guaranteed time slots in the contention free period per superframe.

2.2 Deterministic Synchronous Multi-channel Extension (DSME)

This new MAC behaviour was introduced by the IEEE 802.15.4e enhancement, and later made official in the IEEE 802.15.4-2020 revision. This new network approach is able to schedule time and frequency slots to one or multiple communication partners and, therefore, reducing or even avoiding packet collisions [Kauer, Köstler, and Turau 2018].

Superframe Structure

As said previously, in the legacy IEEE 802.15.4 protocol, the superframes are each composed by 16 time slots: 1 for the beacon containing network and time information, 8 for the Contention Access Period (CAP) and 7 for the Contention Free Period (CFP). In the CAP section of the superframes, nodes exchange control messages via CSMA/CA using a single channel. These messages, when addressed to the PAN Coordinator, can be allocation requests for time slots in the CFP section of the superframe. In this second period (CFP), each node will transmit packets to the node it expressed desire to in the message of time slot allocation sent previously [Meyer, Mantilla-González, and Turau 2021]. In order to accommodate the needs of the newly implemented DSME, an additional structure was developed, a multi-superframe. This new concept is defined as a set of superframes, where the number a superframes present in a multi-superframe varies according to the defined Superframe Order (SO) (figure 2.2). In fact, the possible configurations regarding the size of the information channel are:

- Superframe Order (SO): A number that is used to compute the amount of superframes are present in a given network's multi-superframe, where:

$$0 \leq SO \leq MO$$

- Multi-superframe Order (MO): This order helps in calculating the amount of multi-superframes that can be sent in a single beacon interval, where:

$$SO \leq MO \leq BO$$

- Beacon Order (BO): The base of all metrics, used to calculate the size in symbols of the beacon interval, where:

$$MO \leq BO \leq 14$$

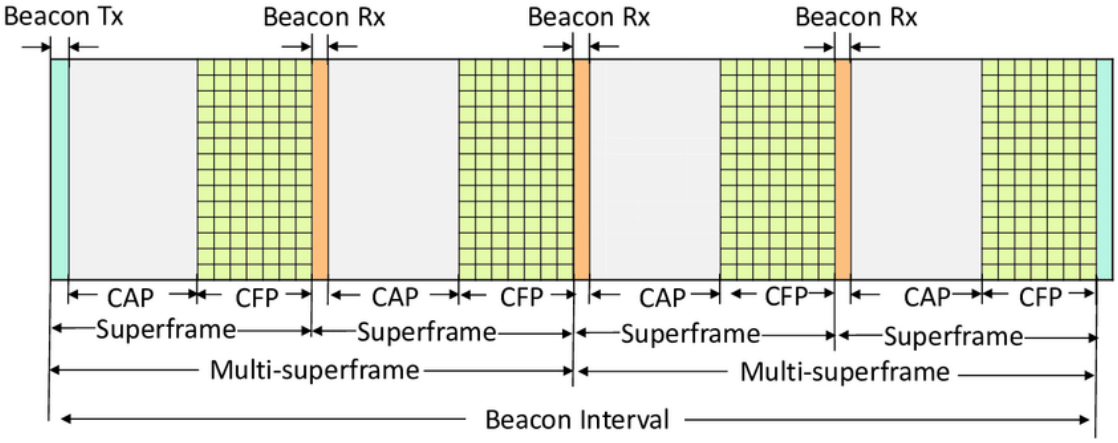


Figure 2.2: IEEE 802.15.4 DSME Multi-superframe structure [Battaglia et al. 2020]

With these values, the IEEE 802.15.4-2020 protocol [IEEE 2020] allows for the calculation of several metrics relevant to the well flowing of the network, such as:

- Beacon Interval: Duration that will be an indicator for when a new beacon needs to be released (2.1)

$$BeaconInterval = aBaseSuperframeDuration * 2^{macBeaconOrder} \quad (2.1)$$

2.1: Beacon Interval Formula

- Multi-superframe Duration: Duration, in symbols, of the multi-superframe that will contain the several superframes to transmit information (2.2)

$$MultiSuperframeDuration = aBaseSuperframeDuration * 2^{macMultisuperframeOrder} \quad (2.2)$$

2.2: Multi-superframe Duration Formula

- Superframe Duration: Duration, in symbols, of the superframe, value that will be crucial in determining the size of each time slot (2.3)

$$SuperframeDuration = aBaseSuperframeDuration * 2^{macSuperframeOrder} \quad (2.3)$$

2.3: Superframe Duration Formula

- Superframes in a Multi-superframe: Number of superframes that, in a given configuration, can fit inside a multi-superframe (2.4)

$$SuperframesInMultiSuperframe = 2^{macBeaconOrder - macSuperframeOrder} \quad (2.4)$$

2.4: Superframes Number in Multi-superframe Formula

- Multi-superframes in Beacon Interval: Number of multi-superframes that, in a given configuration, can be transmitted inside a beacon interval's time (2.5)

$$MultiSuperframesInBeaconInterval = 2^{macBeaconOrder - macMultisuperframeOrder} \quad (2.5)$$

2.5: Multi-superframes Number in Beacon Interval Formula

Multi-channel Transmissions

These new superframe contexts also contain the newly added feature of transmissions over multiple channels, allowing for a single time slot to contain up to 16 concurrent communications between nodes, with the particularity that, in a given time slot, a node that is already enrolling in a channel cannot be in another channel, either being as a receiver or as a transmitter.

CAP Reduction

To further maximize the available guaranteed bandwidth for time critical appplications, CAP Reduction feature was introduced for DSME. This proposal enables the protocol to remove the CAP on all but the first superframe of a multi-superframe, guaranteeing that this initial CAP space is enough for the nodes to exchange allocation requests for the entire multi-superframe (i.e. a node could allocate a time slot in the second superframe of the multi-superframe).

To seize this opportunity to further increase the performance of the network, the IEEE 802.15.4 DSME protocol provides the CAP reduction flag. If this flag is enabled, then only the first superframe of the multi-superframe will have the CAP space, while all the other superframes will have the beacon slot occupied and all other 15 slots will be dedicated to guaranteed time slots in the CFP portion of the superframe (figure 2.3).

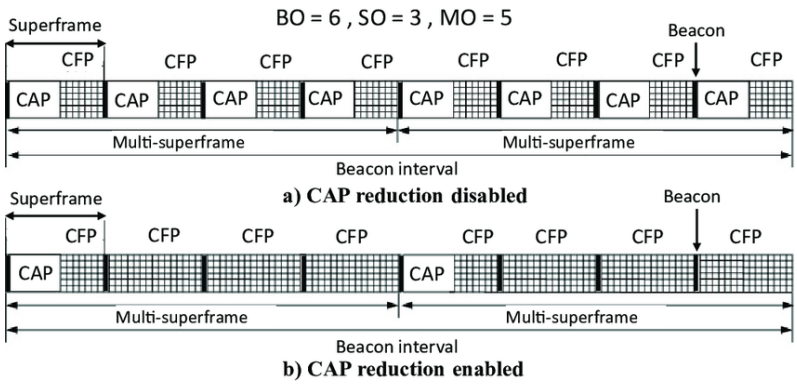


Figure 2.3: With and without CAP Reduction comparison [Sahoo, Pattanaik, and Wu 2019]

Beacon Scheduling

As [Meyer, Malessa, et al. 2021] states in his conclusions of group acknowledgements' worthiness in the context of the IEEE 802.15.4 DSME protocol that the hidden-node situation is a very pertinent issue in the trade-off between throughput and acknowledgement delay in a DSME network. This hidden-node issue occurs when two beacons, both being neighbours of a third node but without being able to communicate with each other, try to allocate the same slot to communicate to this third node. The outcome of these situations is the overlapping of the transmission beacon slot, that being, since the beacons of both transmissions collide, the receiving node cannot hear either beacon transmitted, resulting in an information loss [Hwang and Nam 2014].

The IEEE 802.15.4 DSME, to address this issue, was incorporated with a mechanism that solves this contention problem by prolonging the received beacon allocation commands via sending a *DSME-Beacon allocation notification*. This command frame is sent to all nodes via broadcast by the PAN Coordinator in order to inform all nodes present in the network (even the ones without direct contact within themselves) of an allocation of a node in a superframe, in order to avoid the hidden-node problem from happening. All the nodes collect the allocation information and add it to their own Allocation Counter Table (ACT) [Hwang and Nam 2014].

2.3 Time Slotted Channel Hopping (TSCH)

The newly updated IEEE 802.15.4-2020 [IEEE 2020] brought the new MAC behaviours to allow itself to be more scalable and adapt to specific situations better. One of these new behaviours was the Time Slotted Channel Hopping (TSCH), a particularization of the protocol that is mainly intended for the support of process automation application with a particular focus on equipment and process monitoring [De Guglielmo, Brienza, and Anastasi 2016]. This MAC behaviour was developed in order to improve the reliability of the network, all while maximizing energy savings. This can be done with a static schedule that is periodically repeated [Ben Yaala, Theoleyre, and Bouallegue 2016]. It makes use of pseudorandom channel hopping to combat external interference and frequency-selective multipath fading [Elsts, Fafoutis, et al. 2017].

Slotframe

In this MAC behaviour, the concept of superframe is replaced by a slotframe. Each of these slotframes consists of a number of cells, described by time slot and channel offset. The entirety of the network schedule is a collection of one or more periodically repeating slotframes [Fafoutis et al. 2018]. In each of the time slots contained in the slotframes, packets transmission can occur via contention algorithms (CSMA/CA) or non-contention, like the one available in the CFP portion of a superframe in the IEEE 802.15.4 DSME MAC behaviour [Kurunathan 2021] (figure 2.4). Each basic communication resource has as an identifier a pair formed by the definition of the time slot and the channel it will be transmitting on. A very important metric for the composition of the physical channel from the logical one is the Absolute Slot Number (ASN), that dictates the total number of time slots elapsed since the network deployment [Vogli et al. 2018].

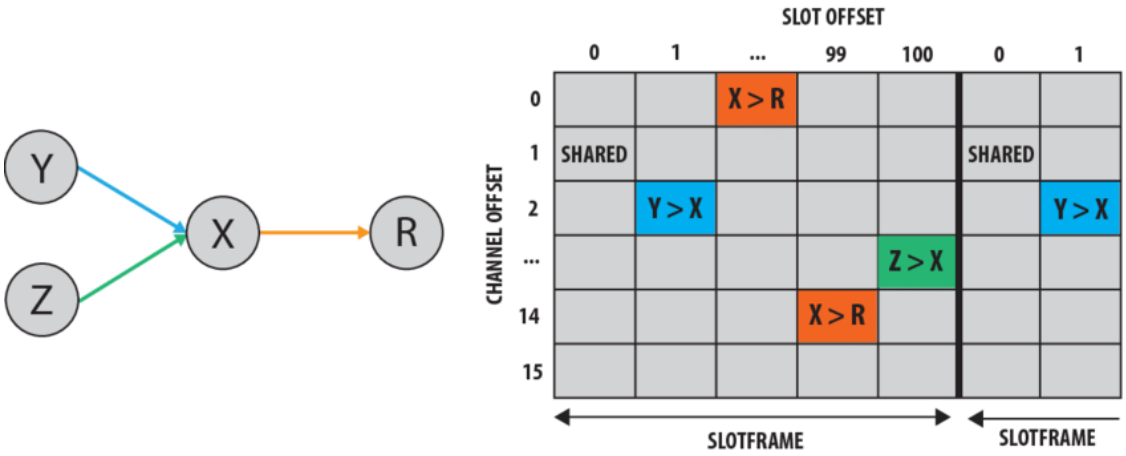


Figure 2.4: IEEE 802.15.4 TSCH Slotframe structure [Daneels et al. 2017]

Similar to the extension of a superframe into a multi-superframe in the DSME, an augmentation of the transmission structure also occurs here, where the concept of a multi-slotframe is introduced. These are established in parallel to one another, where they are sending information simultaneously in the same time slots, but differing on the channel they are using [Meng et al. 2018]. The maximum number of usable channels for transmitting information between nodes is 16 and, therefore, a maximum number of 16 slotframes per multi-slotframe is also established. In networks of this kind, some channels can be left out of usage to improve energy efficiency or in case the channel quality is deteriorated [Kurunathan 2021].

Time Synchronization

The IEEE 802.15.4 TSCH required for its operation, that nodes' were tightly time-synchronized. Synchronization between network nodes happens whenever a device exchanges a packet with a time source neighbor. From here, two different synchronizations can happen: an acknowledgement-based one, that relies on the time information received via the acknowledgment frame sent from the packet receiver, or a frame-based synchronization, that gets the correct time from the arrival time of a frame from the time source neighbor. In these networks, a PAN Coordinator works as a time source neighbor, keeping track of the devices of the network and as a time keeper, meaning that, to keep synchronization, the network devices must communicate with the PAN Coordinator to keep themselves synced with the entirety of the network [Elsts, Duquennoy, et al. 2016].

Because of that requirement, [Stanislowski et al. 2014] developed an adaptive synchronization, meaning that each network node would learn its drift compared to the time source and adaptively compensates for it. But, even with this new algorithm to solve the major synchronization demand, there were still synchronization delay errors of up to tens of seconds. [Elsts, Duquennoy, et al. 2016] developed a method that synchronizes the nodes to a micro-second accuracy and they were successful in their implementation.

Chapter 3

Research Background

3.1 Overview of Covert Channels

3.1.1 Principles and History of Covert Channels

These days, most information passed between two nodes all the way across the world is subjected to being intercepted and end up in the wrong hands. Several cryptography methods are being created and perfected, however, their vulnerabilities are also being discovered on a daily basis, resulting in information leaks and possible losses that could mean the complete shutdown of an entity.

A solution to this problem, is to hide the very existence of communications all together. [Lampson 1973] developed the first concept and implementation of the covert channel, i.e. those not intended for information transfer at all, such as the service program's effect on the system load.

Initially, the primordial network covert channels were mainly storage based, with the principal target being the header of the packets. [Rowland 1997] proposed the utilization of the IPv4 and TCP header to embed hidden information. In its conclusions, he inferred that the method used could turn a machine into a very stealthy transmission device that would appear to be working normally. The first evidence of an exploit of the timing features of network packets was developed by [Wolf 1989]. In his works, he studied several LAN protocols and inferred their vulnerability to covert channel techniques. Although an evident bandwidth issue would appear, a simple exploit was found in delaying the acknowledgment sent in response to a single message by a single time frame. In conclusion, he referred that covert channels alone are not an extraordinary threat unless they can be exploited in a trojan-horse style.

A metaphor widely used to explain cryptography, the Alice and Bob example, can also be applied here [Simmons 1984] (figure 3.1). This metaphor states that two prisoners (Alice and Bob), while in different cells, have a single communication method that is the transmission of messages one to another, but with every package being thoroughly analysed by a warden. Since they intent on escaping the prison, they must communicate through was the warden cannot fathom what is truly being said, that being, establishing a "subliminal channel" between them in full view of the warden, even though the messages themselves contain no secret (to the warden) information. This warden can present itself in a passive or an active mode [Wendzel et al. 2015]. In the first, he will simply analyse the network, detecting any anomalies and further evidences of a covert channel and proceeds in the

extraction of the embedded messages. The second, on the other hand, introduces himself as an active member of the network, trying to modify the sent messages or inserting its own fraudulent messages [Pfitzmann 1996].

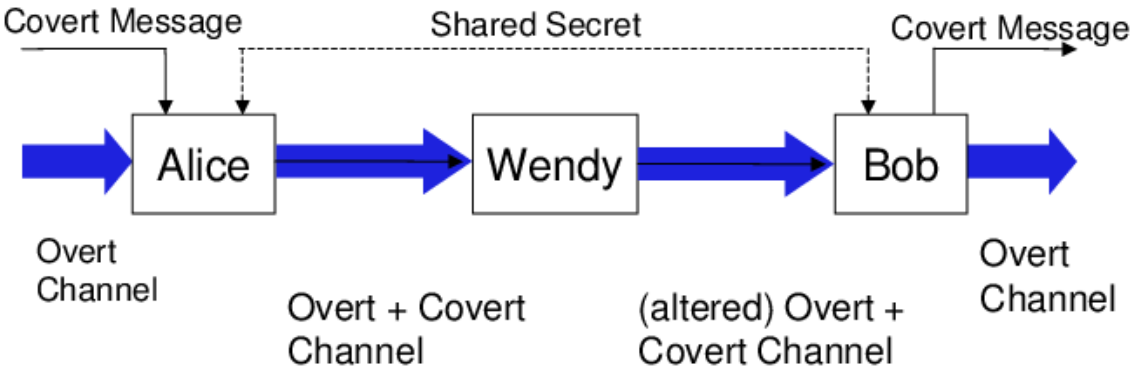


Figure 3.1: Alice and Bob metaphor [Zander and Armitage 2008]

Since this date, the evolution regarding covert channels has been immense, with the implementations varying from network to network, protocol to protocol and device to device. As [Hou and Zheng 2020] referred in their research, covert channels have, since then, been implemented in all kind of network protocols available and with a high range of usability, such as the LTE for mobile communication and simple Wi-Fi (IEEE 802.11) for close range networks.

3.1.2 Authentication

The covert channel applications can present themselves in multiple fashions and with several connotations, such as malevolent usages like sensitive information exfiltration. [Shah, Molina, and Blaze 2006] developed an exploit that combined keystroke loggers and covert channels to retrieve the information assembled by those tools. The usage of a covert channel was primal in order to successfully retrieve the information gathered by the malign software in a covert way, since these are notoriously hard to detect or eliminate.

But a covert channel can also have beneficial applications, such as the exchange of message integrity information and the detection and prevention of Man-In-The-Middle attacks [D. Johnson et al. 2010]. Another beneficial application of covert channel technology is related to the authentication of devices and, therefore, their transmissions. For example, MITM attacks can be used to infiltrate networks and either just listen to communications and steal information, or to manipulate nodes into thinking the attacker is the node they are trying to communicate with. [M. Johnson, Lutz, and D. Johnson 2016] utilized a MITM approach to the covert channel implementation, allowing two nodes to communicate between themselves over existing traffic completely concealed. Another reality is the ability of a MITM attacker to impersonate a node in the network, manipulating other nodes into believing that he is in fact a member of the network and communication with him freely [Cramer and Damgård 1997]. To tackle this last issue, an authentication method can be effective, because by granting that each node is properly insured to be himself, the transmissions being sent and received are guaranteed to be reaching and being emitted from the correct individual.

The possibility of developing an authentication methodology using covert channels has already been studied and analysed, with the help of several techniques to authenticate a network node. [H. Xie and Zhao 2015] designed and implemented a lightweight authentication method in the FTP protocol. This method consists of each node possessing its owns and each other's authentication keys and verifying them when required. These are passed between them by use of an on-off covert channel, as shown in figure 3.2.

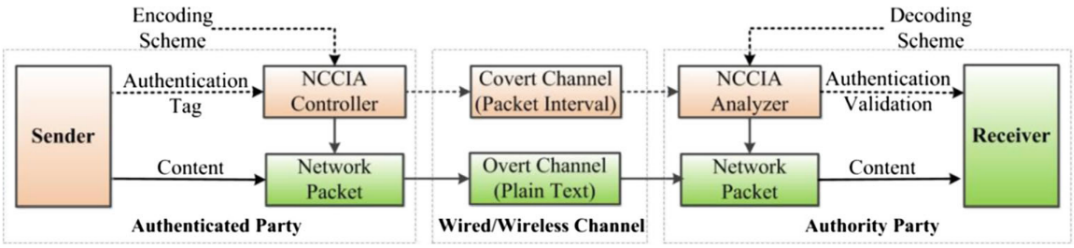


Figure 3.2: Framework of the Covert Channel Module [H. Xie and Zhao 2015]

In the CAN network, [Ying et al. 2019] developed a system to authenticate nodes through the insecure CAN bus. TACAN provides secure authentication of ECUs by relying on covert channels without modifying the CAN protocol. Here, the authentication performed is more complex than the lightweight model presented before, utilizing keys and hashing algorithms to generate authentication messages that will then be compared between sender and receiver to verify each other's identities.

3.1.3 Storage Covert Channel Techniques

Information can be passed on from a sender to a receiver without anyone involved in the network (a warden) realising that some data went covertly from one side to the other. Analysing the network's protocol standard, several reserved fields can be found and some techniques can be used to maximize that same storage covert channel, ensuring that the maximum amount of information can be passed covertly (figure 3.3).

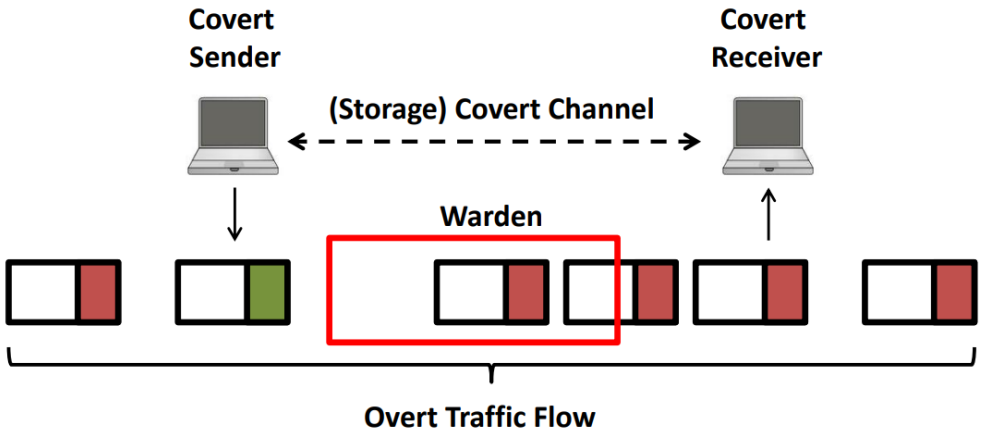


Figure 3.3: Storage Covert Channel [Caviglione 2021]

[Wendzel et al. 2015] classified covert channels of this type into several different patterns, being that these were obtained from a selection of 109 covert channel techniques:

Size Modulation Pattern

The hidden message is embedded into the size of a header element or of a PDU, adding paddings or offsets to modify the overall size of a given element, with different sizes (or size differences) being assigned to readable information.

Sequence Pattern

The covert channel alters the sequence of elements in the header or the PDU to encode covert data. This can be achieved with an alteration to the position of one (or more) elements in the header or by the number of these elements contained in the header or the PDU.

Add Redundancy Pattern

In techniques of this type, additional header elements are created (or expanded) in order to gain an extra amount of empty space suitable to hide information to transmit covertly.

PDU Corruption/Loss Pattern

To transmit covert information, the system generates corrupted PDUs that are embedded with covert information, or drops network packets to signal the hidden information. An additional technique is the drop of a certain number of network packets in order to generate a fixed packet loss rate, that has a covert meaning to the receiver.

Random Value Pattern

The covert channel, after the system generates a random value, checks the packet for a header that contains that obtained value and replaces that number with the information to be covertly transmitted.

Value Modulation Pattern

Of a given element in the header of a packet, the covert channel will modulate the value it contains, changing it to a value from 0 to the maximum it can contain. One technique contained in this pattern is the case-modification, that simply changing the case (upper or lower) of a letter in a packet header element could imply hidden meaning by be overlooked by the network warden.

Reserved/Unused Pattern

A packet header is composed by many elements, with some of them being (in a given network/protocol) unused or reserved. This means that the regular channel will overlook any information being present in those fields. This opens an opportunity to embed covert data into those very same fields.

3.1.4 Timing Covert Channel Techniques

Covert timing channels include the secret messages into the timing behavior at the sender and then extract the covert messages at the receiver. Normally, the delays in network packets are used to deliver covert messages [Tian et al. 2020] (figure 3.4).

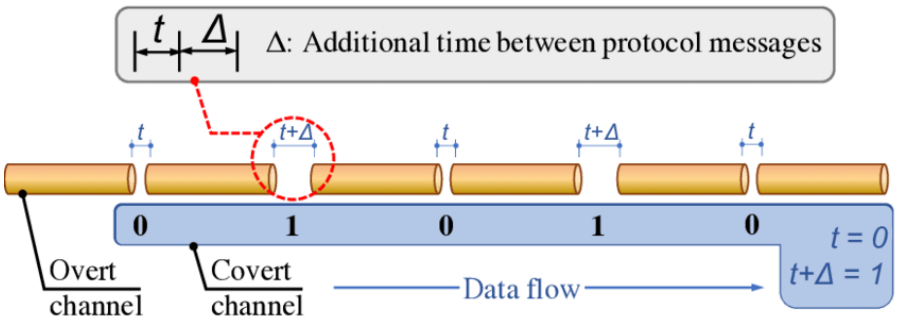


Figure 3.4: Timing Covert Channel Scheme [Cotroneo, De Simone, and Natella 2021]

On-Off

A very simple covert channel was developed by [Lu et al. 2019] and it's called the "On-Off", where a timing interval Tc pre-negotiated between sender and receiver is the key to the information shared. If a packet suffers a sending delay of Tc, then the receiver will interpret this jitter as a 1 bit. If the packet is sent without waiting Tc, then it will mean bit 0.

L-bits to N-packets

Another technique is one the most seen and used ones in scientific works done over the years, the "L-bits to N-packets" [Lu et al. 2019]. This one allows the user to send L bits hidden in N packets, being that L <= N <= Channel length.

Jitterbug

The "Jitterbug" [Lu et al. 2019] method is based on the addition of legitimate latency to the packets being transmitted in the channel. This latency added should be calculated given the already existent network normal latency to manipulate the final delay to the desired one. In the receiver side, if the latency of the channel is dividable by a pre-determined W value, then it means the bit 0. If, on another hand, is dividable by W/2, then it means the bit 1.

Time Replay

[Lu et al. 2019] developed a method to send information covertly through timing intervals, the "Time Replay" technique. This method, similarly to other techniques, revolves around the latency and the packet interarrival times. But, to add an extra layer of undetectability, the values of possible time intervals were divided in two lists, so when the delay was a value contained in the S0 list, the covert bit was 0 and when the delay was present in the S1 list, the bit transmitted was 1.

Inter Arrival Time

The Inter Arrival Time (IAT) between packets could also be used to transmit some covert information to a receiver node in the network. [Ying et al. 2019], through a covert timing channel based on the IAT between packets, successfully authenticated the node transmitting the information and, therefore, made the information transmitted through the typical channel trustworthy. In this method, the authentication message is sent in the exact middle of the authentication frame, this being possible with the help from silence bits, with these being sent in equal number in the beginning and end of the authentication frame, encapsulating the relevant message in the middle.

Packet Length

The possible variability of the length of the packets being transmitted in a network is also able to be translated into covert information. With the weight of the packet header and of the current payload already obtained, an additional piece of information is added to the payload (without meaning) to alter the full packet length [Han et al. 2020]. Obtaining the packet, the receiver will get the packet length and, according to that, if the length is an even value, the covert bit is 0. If the length is an odd number, the bit is 1 [Elsadig and Fadlalla 2018].

L-Bits to N-packets adaptive

Based on the "L-Bits to N-Packets" technique, [Tahmasbi, Moghim, and Mahdavi 2016] developed a method that, before any covert information being transmitted, an analysis of the network is performed in order to get the delay values of the channel and, after them being analysed, they are split into two lists (much like in the Time Replay technique). These delays are then applied to the packets and, according to the list they are placed in, they transmit different information. The analysis performed beforehand allows for an extra layer of undetectability for all the used delay values will be congruent with the regular delays of the network.

Packet Sequence Number

The header on the packets contains a field called the "sequence number" which states the order of sending of a series of sequential packets. If in the receiver node this order is wrong (1,2,3,4 changes to 1,2,4,3), this could have a special meaning to the packet receiver [Zhang et al. 2019]. [Tan et al. 2018] used this method to purposefully alter this sequence in the emitting process to guarantee the order will not be the correct one in the other node and, therefore, create a covertly encrypted sequence of data.

Bit-Rate

Bit-rate is a metric that can be used to to evaluate a channel's performance. It translates the amount of bits a channel can transfer in a given space of time. If a channel has a higher bit-rate than another, that channel is more efficient than the latter. This bit-rate can depend on a various number of conditions, some of which are hardware related, so variations of this field can be considered as normal. [Tan et al. 2018] took advantage of this fact and implemented a covert channel that, altering the bit-rate of the channel to a value more or

less than a pre-accorded one, it would mean a different thing to the receiver node of the network.

Packet Loss

A network can suffer from several factors that can provoke a series of consequences for the transmission quality. One of this consequences is packet loss, that is when a packet (or a series of packets) is lost in the network, by being sent but not being captured by the receiver node. To create a covert channel from the number (or percentage) of lost packets it is necessary to modify some packets so that they are supposed to be "lost" in the network [Tan et al. 2018]. With that and considering the "sequence number" described above, the receiver node with be on the lookout for inconsistencies on the sequence and will interpret the change in there.

Inter-Arrival Time Probabilistic

One final method is based on probabilistic methods [Kiyavash et al. 2013]. An array of bits is encoded using a formula based on a matrix and its values. An additional shaping of this encrypted value is then performed to assign it to an inter arrival time and impose it on a delay between two packets. The receiver node will then de-shape this delay using probabilistic methods, resulting in the encrypted value of the information. With the help of the previous matrix, a decryption will occur, revealing the original information meant to be transmitted.

3.1.5 Covert Channels in 802.15.4 Protocol

The idea of adapting covert channels to the 802.15.4 protocol has been tried before. At the physical layer, [Nain and Rajalakshmi 2017] developed a covert channel using the Direct Spread Spectrum Sequence (DSSS) variant of stenography, where they also propose a secret acknowledgement and error detection method to ensure reliable communication in the covert channel. In the end, they concluded that, using their method, they could transfer confidential information reliably at a significant rate without considerably affecting the performance of primary data reception.

[Tuptuk and Hailes 2015] proposed to develop a covert channel based on link quality. The metric used in the 802.15.4 protocol is known as the Link Quality Indication (LQI), and is typically used in low-power radio devices to give a measure of signal strength. In this paper, they develop two covert channel based attacks: modulation of transmission power and modulation of sensor data. After the analysis of the data retrieved from the attacks execution, they were considered undetectable and thus successful methods for covertly extracting information of an exposed system.

The disadvantage of such approaches is that access and control of the node's physical layer is mandatory, something that is often not possible since these features are deeply embedded at the radio transceiver's firmware. In this line, covert techniques that can function at the MAC sub-layer are much more attractive, particularly since these are usually implemented by a software stack, to which an application can much easily gain access to.

In the 802.15.4 protocol, several types of frames exist and can be altered to embed some secret information to pass along. A normal data frame, for instance, contains a number of loopholes, such as the Frame Control field that contains 16 bits, each with their own significance. From those 16, 5 of them (7-9th and 12-13th) are reserved, that meaning they do not carry information and are disregarded by the machine, and information can be passed covertly in those fields [Martins and Guyennet 2010].

Another example is the Sequence Number field. This contains the numbering of each packet to identify it in the acknowledgement message. If the packet is signaled as not part of a sequence, then this field is not initialized on the transmission and is only assigned on reception, meaning that its value is not checked upon reception, being able to carry 8 bits of hidden information [Martins and Guyennet 2010].

Lastly, the Address Info field. This contains information on the destination of the packet and on the sender, including addresses and PAN coordinator references. In the source address specifically, if the packet is specified has having a nonexistent source address, this field is not checked and not used, but with its space still there vacant. This space can go from 16 bits (short) to 64 (extended), opening the door on a major covert data transmission between source and destination nodes [Martins and Guyennet 2010].

For other types of frames in the protocol, some of the same fields apply. For instance, for a beacon frame, the Sequence Number field can still be uninitialized and used for covert purposes. On an acknowledgement frame, not only this Sequence Number field applies, but also the Frame Control one, also specified above [Martins and Guyennet 2010]. Storage covert channels, however, are significantly easier to detect and mitigate than timing covert channels.

As shown, regarding the IEEE 802.15.4 -2020, a *de facto* standard for the underlying WSN infrastrucutre of many IoT and Industry 4.0 systems, little attention has been given to the deployment of covert channels in the protocol. Available literature approaches the protocol in its previous versions, only focusing storage-based covert channels, not addressing a fundamental threat of network timing channels. Moreover, there is no simulation model available that encompasses such covert channel implementations, which is fundamental for network designers and security experts to evaluate risks, and to further support the development of detection and mitigation strategies.

Chapter 4

Network Covert Channel Simulation Model

This Chapter overviews the development of the Network Timing Covert Channel simulation model. It presents a brief value analysis, which includes a detailed comparison and selection of simulation frameworks using an analytic evaluation method. Furthermore, is describes the model implementation design, along with its design alternatives. To conclude the chapter, an experimental validation is performed and documented.

4.1 Overview

Some of the objectives of this Thesis is to thoroughly analyze the performance of known timing covert channel techniques applied to the IEEE 802.15.4 protocol, using metrics such as covert channel concealment, transmission efficiency and capacity, while exploring new opportunities for innovative covert channel usage regarding authentication and authorization techniques in a low computing power environment. To achieve this, it is fundamental to rely on simulation tools that can support the analysis of the performance of such communication channels. This chapter addresses this and, in addition to the model in itself, the simulation setup and configurations, along with an utilization manual will be fully made available in order to boost future works in regard of this very same topic. This simulation model will feature a variety of covert channel implementations in both the DSME and the TSCH MAC behaviours of the protocol. The referred implementations will be the target of an elaborated performance analysis and comparison, in order to extract the most out of the covert channel and its usages in transmitting covert information.

4.2 Value Analysis

4.2.1 Opportunity Identification

The lack of available implementations of network covert channels upon the IEEE 802.15.4, hinders the analysis of such attacks' impact and the development of new detection and mitigation mechanisms. We believe it is worthwhile and of great importance to the IoT and cyber-security community to undertake such endeavor. Furthermore, it is a fundamental step to support the research work on these topics that is to be carried out in current research frameworks highlighted in Section 1.2.

4.2.2 Opportunity Analysis

To implement a set of network covert channels, it is fundamental to rely on already existent and updated IEEE 802.15.4 protocol simulation models, avoiding the extensive job of implementing many of the complex protocol communication features. Thus, the existence of already developed and available models, which usually are connected to a particular simulation framework, is a main criteria in the selection of the simulation framework as implementation candidate. Since the protocol has been around since 2003 (with multiple updates, the most recent one tracing back to 2020), several simulation models were develop using a variety of tools. In what follows we present an overview of relevant simulation frameworks along with the available models.

OMNeT++

Objective Modular Network Testbed in C++ (*OMNeT++*) is a discrete event simulator based on *C++* for modeling simulated networks. Everything involving communications, wired or wireless, is passive of modeling in this open-source software. The simulator can be used for traffic modeling of telecommunication networks, to protocol modeling, to modeling networks and hardware components (including multiprocessors and other distributed hardware systems) [Varga 2003]. With the aid of the *INET* framework, an open-source *OMNeT++* model suite for wired, wireless and mobile networks, the simulation could fulfill the necessity of communication between the several components of the network and, therefore, simulate the packets and the information being covertly passed between them (figure 4.1).

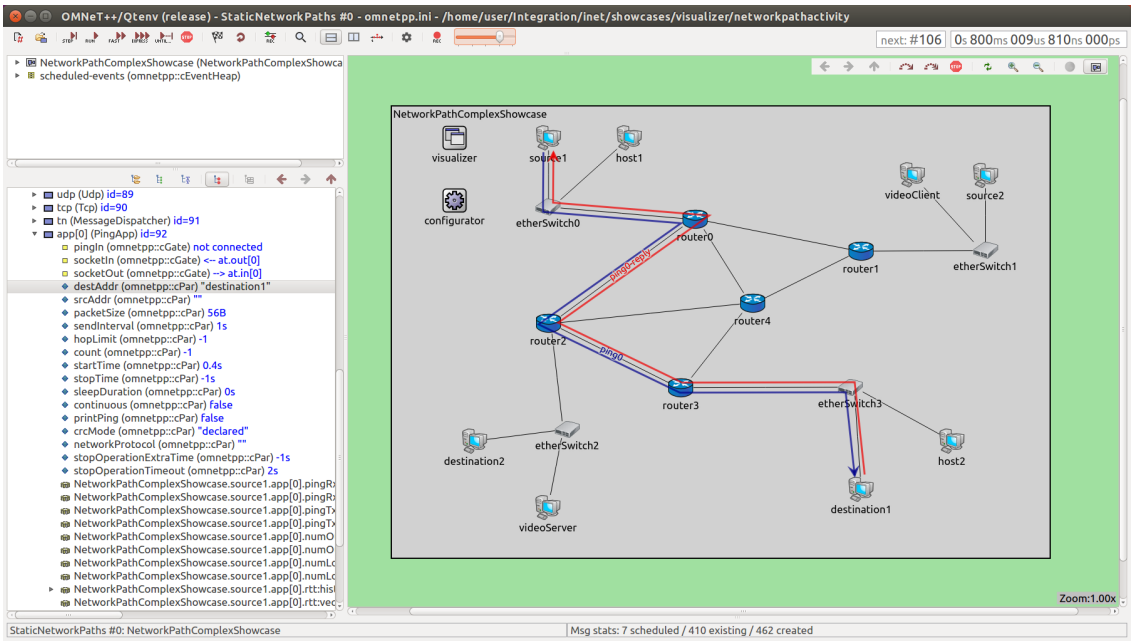


Figure 4.1: OMNeT++ INET Simulation [OMNeT++ 2022]

But the use of OMNeT++ as a simulation framework was only half the picture, since there had to be a model implemented that reproduced the desired protocol communication features, like the Guaranteed Time Slots support of the DSME protocol. After some research, the only option available was openDSME [Köstler et al. 2016]. This was an open-source

implementation of the latest version of the 802.15.4 DSME protocol developed by a team of researchers in the *Institute of Telematics* at *Hamburg University of Technology*.

Since DSME is only one of the variants of the 802.15.4 protocol, TSCH had also to be available in the form of a simulation model in the framework to be chosen. Again from the *Hamburg University of Technology*, but this time from the *Institute of Communication Networks*, a team of researchers developed and made available in an open-source license a fully functional TSCH model that integrates both OMNeT++ and the INET framework to create interactions and send packets between simulation nodes [Krueger and Yevhenii 2022]. This model was created to impact the development of wireless avionics intra-communications.

NS-3

ns-3 is a discrete-event network simulator for Internet systems, targeted primarily for research and educational use. *ns-3* is free, open-source software, licensed under the *GNU GPLv2* license, and maintained by a worldwide community [nsnam 2022]. It was developed as an upgrade from the already highly used ns-2. It succeeds in improving the realism of the models in order to make them closer in implementation to the actual software implementations they represent (figure 4.2).

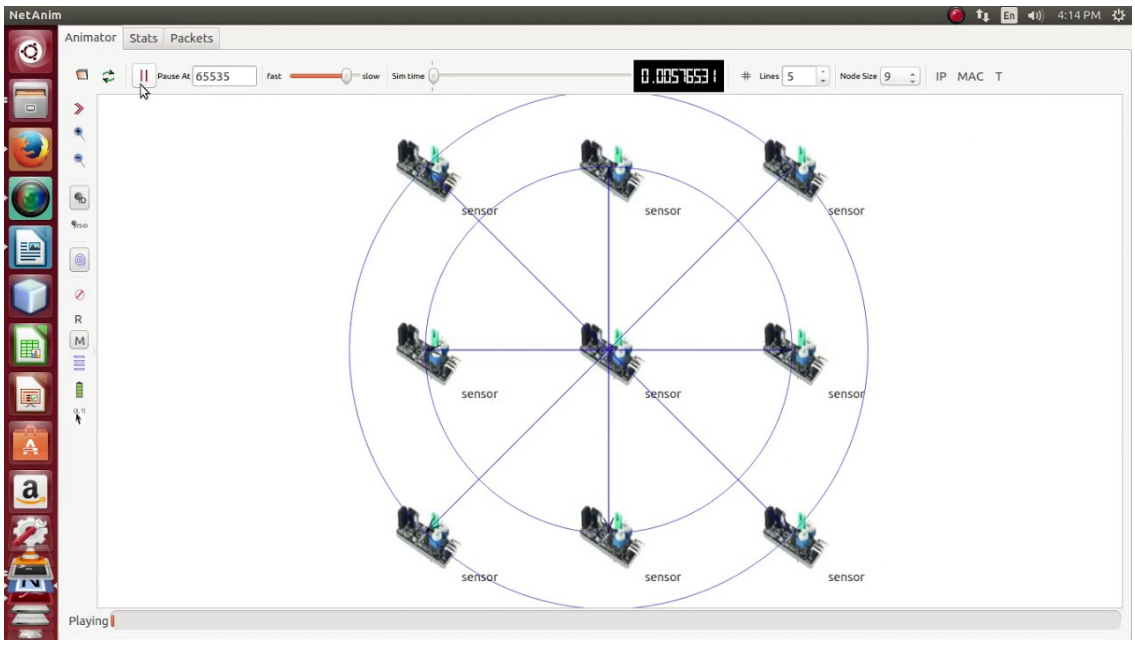


Figure 4.2: ns-3 Simulation [NS3 simulations 2017]

[Hwang and Nam 2014] successfully implemented the DSME variant of the enhanced protocol IEEE 802.15.4e in a beacon collision environment in order to schedule the beacons and, therefore, avoid collisions and loss of information. This model was also tested and, based on those experiments, improvements to the beacon scheduling model present in DSME are proposed and result in an enhanced DSME.

The TSCH operational mode of the enhanced protocol 802.15.4e was implemented in the ns-3 simulation framework by Kourzanov [kourzanov 2022]. The model was, after that, fully

disponibilized in a public GitHub repository for the further usage by the scientific community to extend the works developed.

CooJa

CooJa Simulator is a flexible Java-based network simulator specifically designed for Wireless Sensor Networks. This simulation framework is exclusive of the Contiki operating system, this focusing primarily on low power IoT devices. CooJa is flexible in that many parts of the simulator can be easily replaced or extended with additional functionality. Example parts that can be easily replaced or extended include the simulated radio medium, simulated node hardware, and plug-ins for simulated input/output [Osterlind et al. 2006] (figure 4.3).

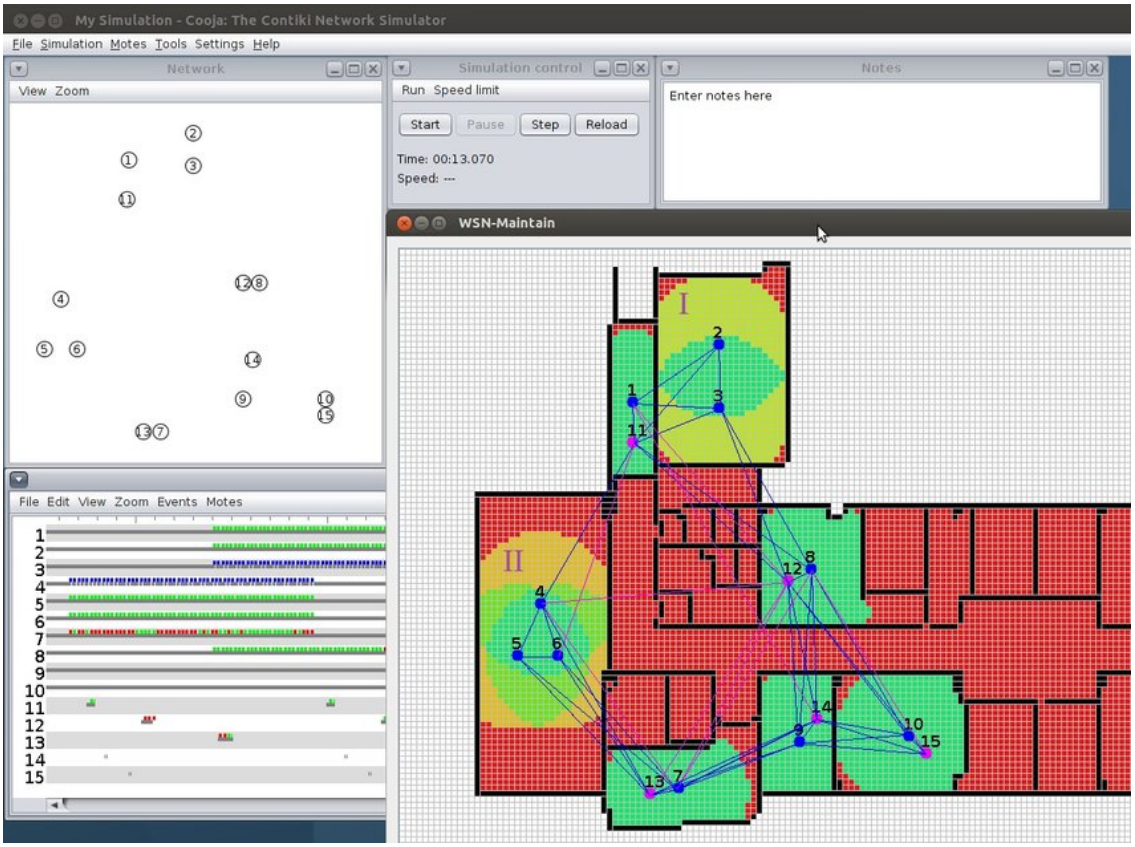


Figure 4.3: CooJa Simulation [Sitanayah, Sreenan, and Fedor 2013]

Like the OMNeT++ simulation DSME inplementation performed by investigators in the *Institute of Telematics* at *Hamburg University of Technology*, that same team also developed the referred model (openDSME) [Köstler et al. 2016] in the Cooja simulation framework.

According to the article created by [Shih, Xhafa, and Zhou 2015], an implementation of the TSCH version of 802.15.4 was achieved successfully and fully tested.

4.2.3 Analytic Hierarchy Process (AHP)

In order to compare and make an appropriate decision on which simulation framework would be the best option to carry out the works being performed in the course of this thesis, a multi-criteria decision analysis had to be performed. This could be made with a variety of analytic methods, but the chosen one was the Analytic Hierarchy Process (AHP) [Saaty 1994]. This method uses numeric techniques that aid the deciders in choosing an option from a discreet set of alternatives. It allows the prioritization of alternatives in a situation of conflicting criteria, in order to select the option that most adapts to the solution required.

Criteria

1. **Knowledge of the tool (Previous usage):** The knowledge of the tool as proven itself as a fundamental metric in choosing the framework, as the usage of a tool already studied before saves valuable time in learning of the functionalities available and the general know-how of the framework.
2. **Performance:** Since the system available to run the simulations is not a high-spec build, the performance analysis is very weighted on. Although the simulations will not be very large in either time or number of nodes, the availability is a factor that would be largely taken into account.
3. **Flexibility:** The ability for the solution to adapt to possible or future changes in its requirements, since the work being done is highly experimental and requires a framework that can adapt to changes either in the protocol or the covert channel in itself, this metric is again a very important one to take into account.
4. **Complexity:** The amount of interaction between modules in a system. Although the simulations that will take place in the works performed in this thesis do not contain an high amount of modules, a simulation framework that do not possess a high complexity factor in terms of connections between modules is also relevant.

With these criteria in place, an Hierarchic Decision Tree in its initial phase could already be drafted to help with the understanding of the evaluation that will be taken place (figure 4.4).

	Knowledge	Performance	Flexibility	Complexity
Knowledge	1	3	5	7
Performance	1/3	1	2	4
Flexibility	1/5	1/2	1	2
Complexity	1/7	1/4	1/2	1

Table 4.1: Criteria Weights Matrix

AHP process of selection

Firstly, each metric is assigned to a weight from 1 to 10 (1 being the lowest priority and 9 being the highest) in relation to one another. In the view of the works that will be performed in the effort of this thesis, the priorities were decided as shown in table 4.1. From this, and after the sum performed in table 4.2, a normalization is required to achieve the matrix in its final form with each criteria having its relative priority (table 4.3).

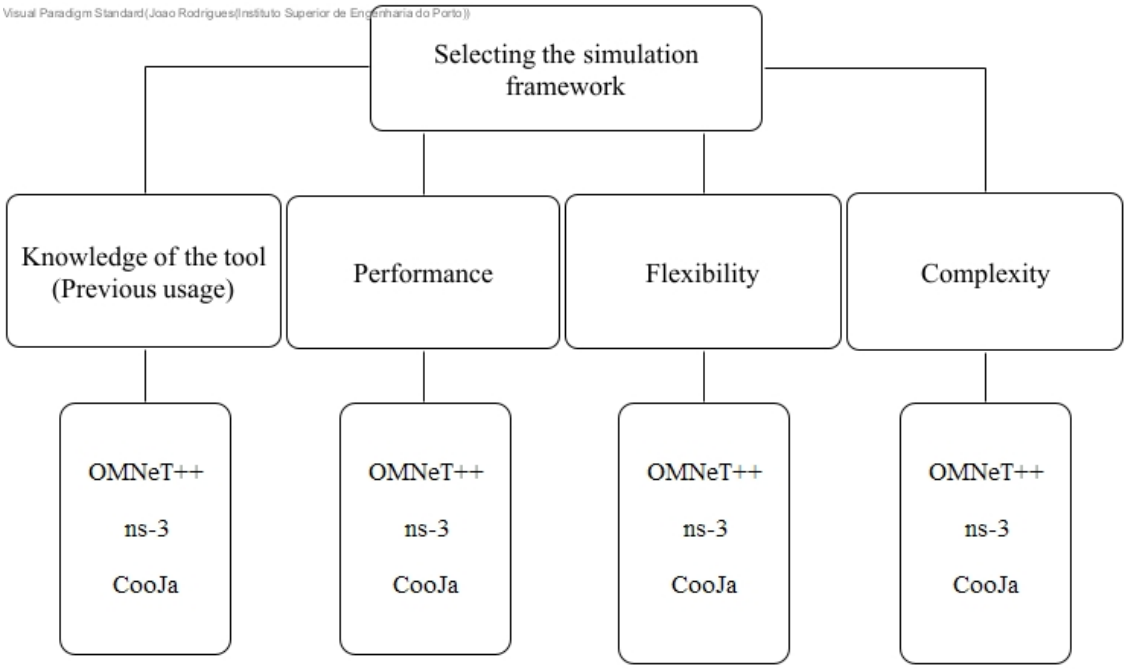


Figure 4.4: Hierarchic Decision Tree

	Knowledge	Performance	Flexibility	Complexity
Knowledge	1	3	5	7
Performance	1/3	1	2	4
Flexibility	1/5	1/2	1	2
Complexity	1/7	1/4	1/2	1
Sum	176/105	19/4	17/2	14

Table 4.2: Criteria Weights Matrix With Sum

	Knowledge	Performance	Flexibility	Complexity	Relative Priority
Knowledge	105/176	12/19	10/17	1/2	0,5791
Performance	35/176	4/19	4/17	2/7	0,2326
Flexibility	21/176	2/19	2/17	1/7	0,1213
Complexity	15/176	1/19	1/17	1/14	0,0670

Table 4.3: Normalized Criteria Weights Matrix

$$A = \begin{bmatrix} 1 & 3 & 5 & 7 \\ 0,33 & 1 & 2 & 4 \\ 0,20 & 0,50 & 1 & 2 \\ 0,14 & 0,25 & 0,50 & 1 \end{bmatrix} \longrightarrow \begin{bmatrix} 0,60 & 0,63 & 0,59 & 0,50 \\ 0,20 & 0,21 & 0,24 & 0,29 \\ 0,12 & 0,11 & 0,12 & 0,14 \\ 0,09 & 0,05 & 0,06 & 0,07 \end{bmatrix} \longrightarrow X = \begin{bmatrix} 0,58 \\ 0,23 \\ 0,12 \\ 0,07 \end{bmatrix}$$

Using the formula described in 4.1, where A and X were obtained above, the obtaining of λ_{max} was possible and, from then the Consistency Index (CI) (4.2) and, finally, the Consistency Ratio (CR) (4.3). In this last formula, the RI index (Random Index) is a constant developed by [Saaty 1994] and investigated by [Alonso and Lamata 2006], where the values used in this analysis are the ones referred as to from "Wharton".

$$A \times X \approx \lambda_{max} \times X \tag{4.1}$$

4.1: λ_{max} Formula

$$CI = \frac{\lambda_{max} - n}{n - 1} \tag{4.2}$$

4.2: Consistency Index

$$CR = \frac{CI}{RI} \tag{4.3}$$

4.3: Consistency Ratio

$$\begin{bmatrix} 2,36 \\ 0,94 \\ 0,49 \\ 0,21 \end{bmatrix} \approx \lambda_{max} \begin{bmatrix} 0,58 \\ 0,23 \\ 0,12 \\ 0,07 \end{bmatrix}$$

$$\lambda_{max} = \frac{(2,36 \div 0,58) + (0,94 \div 0,23) + (0,49 \div 0,12) + (0,21 \div 0,07)}{4} \approx 3,81$$

$$CI = \frac{\lambda_{max} - n}{n - 1} = \frac{3,81 - 4}{4 - 1} \approx -0,06$$

$$CR = \frac{CI}{RI} = \frac{-0,06}{0,90} \approx -0,07$$

Given that the consistency ratio is $-0,07$, and that $-0.07 < 0,1$, therefore, a conclusion can be made that the properties obtained from the calculations above are considered as consistent.

Simulator's results

From there, the next step is the comparison of each simulation framework in their performance according to each metric to be evaluated. To conclude about the simulator's performance, a research had to be made to gather data about the metrics to be analysed. [Zarrad and Alsmadi 2017] compared the OMNeT++ with the ns-3 (of the relevant simulators) for integrability, reusability, testability, flexibility and complexity. In all these metrics, the OMNeT++ was considered as a better simulator by a small margin, beating closely the ns-3.

[D. Xie, J. Li, and Gao 2020] compared 5 simulators through different simulation scenarios and with different outcomes. The OMNeT++ was considered a more complete simulator with a higher economy index (the cost of establishing a simulation environment is small and the economic cost is low). On another metric, [Khan, Bilal, and Othman 2012] compared

the OMNeT++ with the ns-3 and concluded on performance metrics, such as CPU and memory usage, where the ns-3 surpassed all marks and established itself as the one with the best performance.

In [Weingartner, Lehn, and Wehrle 2009], the graphical interface of the OMNeT++ simulation framework was considered as a defining factor in surpassing ns-3, for its ease of usage and intuitiveness, while also agreeing with the previous research's conclusion of the performance greatness of the ns-3. Regarding the final metric, and considered as most important, the knowledge of the tool highly favours the OMNeT++ simulation framework, has it was used in a previous work conducted by the student performing this work.

From tables 4.4 to 4.11 are the results of each simulation framework in comparison with one another regarding the relevant metrics, results that fill the hierarchic decision tree developer earlier in this chapter with some new values that help furthermore in the understanding of this evaluation (figure 4.5).

Knowledge	OMNeT++	ns-3	CooJa
OMNeT++	1	9	9
ns-3	1/9	1	1
CooJa	1/9	1	1
Sum	11/9	11	11

Table 4.4: Knowledge Weights Matrix

Knowledge	OMNeT++	ns-3	CooJa	Relative Priority
OMNeT++	9/11	9/11	9/11	0,8181
ns-3	1/11	1/11	1/11	0,0909
CooJa	1/11	1/11	1/11	0,0909

Table 4.5: Knowledge Weights Normalized Matrix

Performance	OMNeT++	ns-3	CooJa
OMNeT++	1	1/3	3
ns-3	3	1	6
CooJa	1/3	1/6	1
Sum	13/3	3/2	10

Table 4.6: Performance Weights Matrix

Performance	OMNeT++	ns-3	CooJa	Relative Priority
OMNeT++	3/13	2/9	3/10	0,2510
ns-3	9/13	2/3	3/5	0,6530
CooJa	1/13	1/9	1/10	0,0960

Table 4.7: Performance Weights Normalized Matrix

Flexibility	OMNeT++	ns-3	CooJa
OMNeT++	1	3	6
ns-3	1/3	1	3
CooJa	1/6	1/3	1
Sum	3/2	13/3	10

Table 4.8: Flexibility Weights Matrix

Flexibility	OMNeT++	ns-3	CooJa	Relative Priority
OMNeT++	2/3	9/13	3/5	0,6530
ns-3	2/9	3/13	3/10	0,2510
CooJa	1/9	1/13	1/10	0,0960

Table 4.9: Flexibility Weights Normalized Matrix

Complexity	OMNeT++	ns-3	CooJa
OMNeT++	1	3	6
ns-3	1/3	1	3
CooJa	1/6	1/3	1
Sum	3/2	13/3	10

Table 4.10: Complexity Weights Matrix

Complexity	OMNeT++	ns-3	CooJa	Relative Priority
OMNeT++	2/3	9/13	3/5	0,6530
ns-3	2/9	3/13	3/10	0,2510
CooJa	1/9	1/13	1/10	0,0960

Table 4.11: Complexity Weights Normalized Matrix

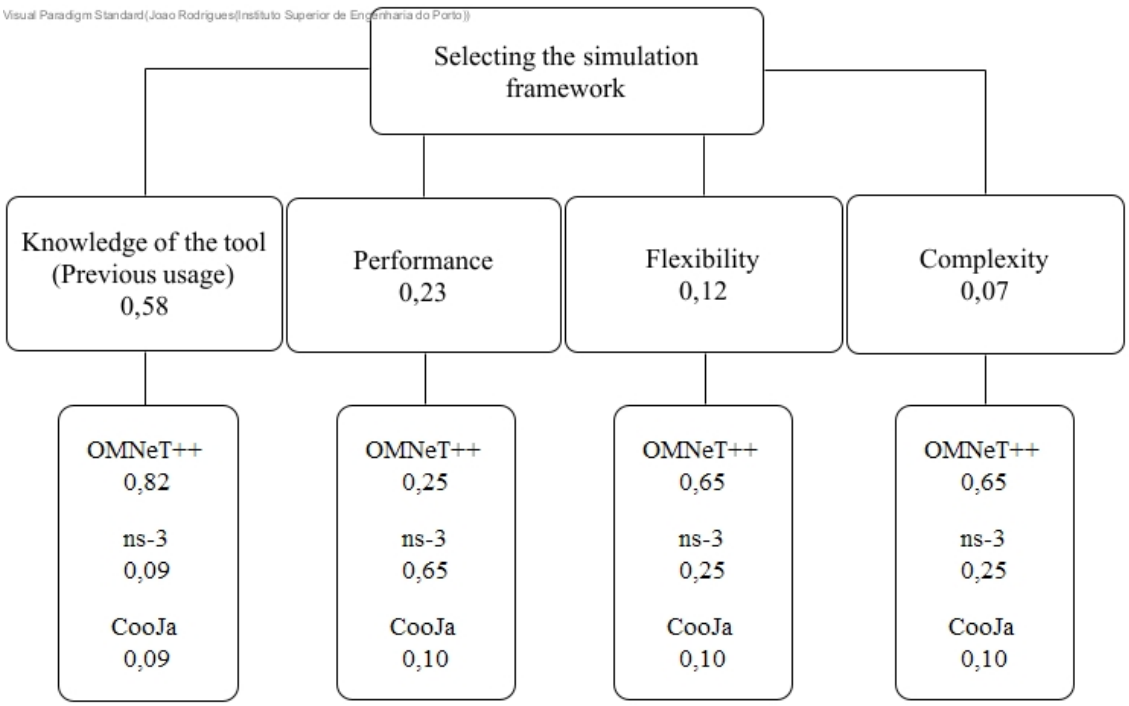


Figure 4.5: Hierarchic Decision Tree With Priorities

Conclusion

With the values calculated in the matrices above, came the time to perform the final calculation in order to obtain the final results of priority.

$$\begin{bmatrix} 0,82 & 0,25 & 0,65 & 0,65 \\ 0,09 & 0,65 & 0,25 & 0,25 \\ 0,09 & 0,10 & 0,10 & 0,10 \end{bmatrix} \times \begin{bmatrix} 0,58 \\ 0,23 \\ 0,12 \\ 0,07 \end{bmatrix} \approx \begin{bmatrix} 0,66 \\ 0,25 \\ 0,09 \end{bmatrix}$$

In conclusion, through an analytic hierarchic process decision making method in order to obtain the most suitable simulation framework to carry on the works to be performed in the latter stages of this thesis, the final conclusion is that the most sensible choice is the OMNeT++ simulation framework, achieving a final score of 66% capability. In second place, the ns-3 obtained a 25% mark and, finally, the CooJa simulator with 9%.

4.3 Architecture for DSME Timing Covert Channel Analysis

4.3.1 openDSME Overview

As presented in the previous section, in terms of simulation framework, the final decision was to rely on the OMNeT++ tool together with INET. The IEEE 802.15.4 DSME model was made available by [Köstler et al. 2016].

This model is composed by two fundamental layers integrated into the MAC link layer, the DSMELayer and the DSMEAdaptionLayer (figure 4.6). The first one is responsible for implementing the newly released DSME MAC behaviour and all its features, while the adaption layer is liable for the base IEEE 802.15.4 functions required to perform a cohesive link with the rest of the OSI layers [Köstler et al. 2016]. The higher layer communicates with the lower DSME layer through two interfaces known roughly as Service Access Points (SAP):

- MAC Common Part Sublayer (MCPS-SAP): Responsible for sending and receiving messages that are composed with a payload, with also the availability of queueing messages for transmission.
- MAC Sublayer Management Entity (MLME-SAP): In charge of the network management tasks.

The DSMEPlatform module is responsible for harmonizing the whole model, interconnecting both the DSMELayer and the DSMEAdaptionLayer. This module is also responsible for the emission of timers (important in the developments to be performed in the works of this thesis) and for the correct flow of the packets throughout the network. In the process of a packet transmission, the message contained in said packet is an instance of a DSMEMessage, with this class containing the packet and its full payload, along with some validations regarding the status of the transmission. This messages are generated by a traffic generator module, that creates them using a configured payload length value and releases them into the network, emitting a signal to inform the network nodes that a new transmission started.

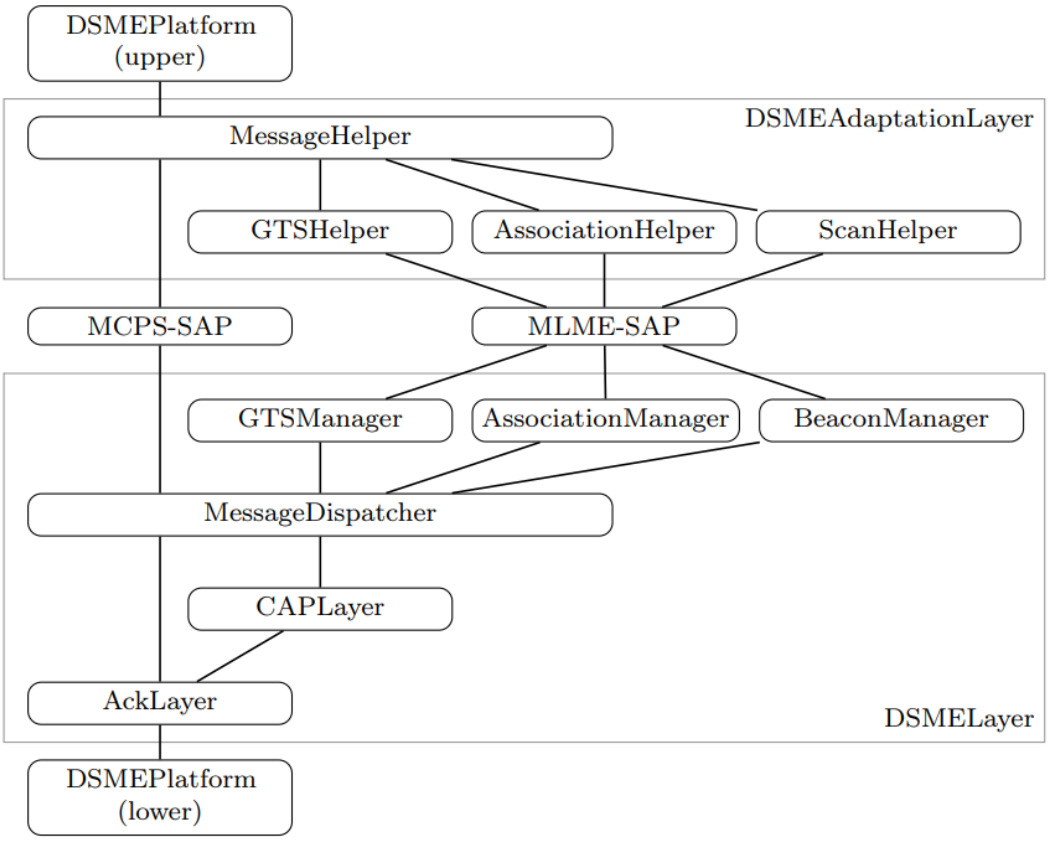


Figure 4.6: openDSME architecture [Köstler et al. 2016]

4.3.2 Covert Communications Module Description

A preexistent condition of implementing the covert channel in the CFP section of the superframe was in effect. This was imperative, since the guaranteed time slots provide an assurance of available bandwidth for a given transmission compared to the CAP section, where the contention algorithm is based on competition. Given the existent module of the IEEE 802.15.4 DSME explained before, the approach to insert code in order to add a covert channel had to be thoroughly planned. Some requirements had to be taken into account regarding several factors:

Function

The main concern of this implementation is its function, that being, a functional covert channel implementation, capable of transmitting covert information from one node to another without raising suspicion and without the encoded message being generally visible to a network warden.

Modularity

The base idea from the concept of modularity is to decouple design decisions that are likely to change, so that they can remain independent and affect the minimal surroundings possible [Sullivan et al. 2001]. Applying this concept to the implementation to be performed

is imperative, due, not only to the independency of the module from the rest of the model, but to achieve a proper responsibility segregation, since no other existent component of the network should have the task of implementing a covert channel in itself.

Performance

[Cortellessa, Di Marco, and Inverardi 2011] defines performance as a measure of how effective is a software system with respect to time constraints and allocation of resources, with some relevant metrics being response time, throughput and utilization. Since the works being discussed in this thesis have tight timeliness constraints, performance metrics relevant to time are being highly taken into account. As referred earlier in chapter 1, IoT devices are not capable of highly resource-consuming algorithms and functions, so the lightweightness of the implementation is also very important for the final goal to be achieved.

4.3.3 Design Alternatives

Alternative 1

Firstly, the idea was to implement a covert channel into a traditional communication system between two nodes. This would be achieved through the use of a delay-injecting algorithm that is capable of retarding the packets in the right amount of time, this delay that will be interpreted by the algorithm responsible for the receiving part of the covert channel. It will compare the time between the current packet and the one before that and, with that difference, it will determine the covert information sent in that transmission.

After some research and testing on the existent model, the connection to be made between this new Covert Helper module and the ones already present in the DSME model had to be performed in two different places, the MessageDispatcher module in the DSMELayer for the transmitting node and the MessageHelper in the DSMEAdaptionLayer for the receptor node (figure 4.7).

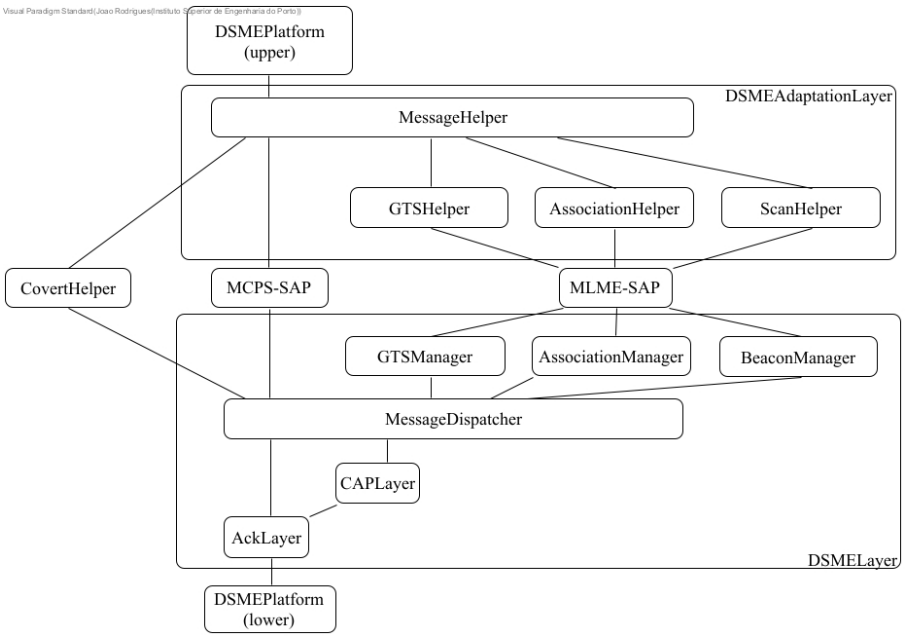


Figure 4.7: openDSME Covert Module architecture (alternative 1)

Alternative 2

But this first alternative had two major setbacks. Firstly, it would require a communication between the two fundamental layers of the model (DSMELayer and DSMEAdaptionLayer) that is only possible in existence in the Service Access Points. The addition of a module interconnecting both layers would violate the protocol's rules of the message's flow through the layers of the network. Secondly, the singular model alternative would encompass both the sender and the receiver covert channel implementations, and that presents itself as a design flaw. They could be further encapsulated, requiring that only the designated node for transmitting covert information possess the implementation for transmission and likewise for the covert receiver node, resulting in a proper distribution and allocation of responsibilities.

Therefore, this new design alternative encompass this observations and further divides the Covert Helper module into two: the Covert Helper, responsible for the transmission of the messages with covert information and, therefore, inserting time delays into the network and the Covert Helper Receiver, in charge of interpreting the information being transmitted in the subliminal channel, that being, calculating the timing interval between received packets and, with that value, decoding the information being covertly transmitted.

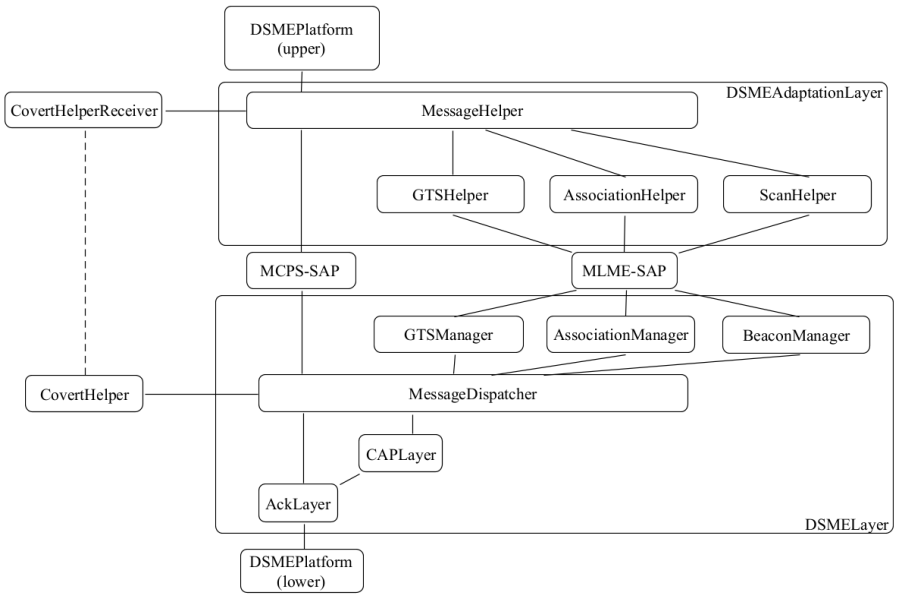


Figure 4.8: openDSME Covert Module architecture (alternative 2)

4.4 Implementation

The covert channel implementation was performed in the MAC layer, divided into two different sections, the covert channel information sender and the information receiver.

4.4.1 Covert channel transmitter

From the sender's point of view (algorithm 4.1), a covert message to be transmitted must be defined in the initialization of the node, being that this message, if shorter than the total of transmitted information, will proceed to be continuously transmitted in loop.

Being that in the simulations currently running several nodes are transmitting packets, a limitation to a single node transmitting covert information was implemented by checking if the current node transmitting is the desired one. Passing this checkpoint, the covert message will be deconstructed into bits, with 2 counters to help the node acquire the bit to be transmitted next. These 2 counters will help on the character to be transmitted of the message and, from that character, the other counter will inform of the bit to be encoded.

In the On/Off covert channel itself, it being of the timing variety, a delay had to be arranged to inform the receiver that the bit being passed was the one we wanted to. This delay was studied and tested in order to achieve the lowest possible delay in order to fully maximize the covert channel efficiency. The lowest delay possible to create a difference recognized by the receiver is the minimum time possible, 1 symbol. So, when the desired bit to be transmitted is 1, the algorithm will automatically create a delay of 2 symbols in before the transmission of the message (figures 4.9 and 4.10).

```
symbols = unsigned(diffIntervals[x]);
letterCounter++;
if(letterCounter >= 208)
    letterCounter = 0;
lastTime = millis;
covertPackets++;
covertPacketsCounter++;
bitssent++;
myfile.open ("results/writer" + std::to_string(so) + ".csv", std::ios_base::app);
myfile << (std::to_string(millis) + "," + std::to_string(covertPacketsCounter) +
myfile.close();
this->dsme.getPlatform().startDelay(NOW + unsigned(symbols));
return;
```

Figure 4.9: Delay Implementation

```
std::vector<uint8_t> diffIntervals {0,2};
std::vector<char> diffChars {'0','1'};
```

Figure 4.10: Delay Intervals

The other techniques implemented, being them the Time Replay and the L-Bits to N-Packets, involved passing additional information through the channel other than a single bit. To achieve such, the first step of the implementation was to create and assign the delay intervals to be considered by the model. For instance, a L-Bits to N-Packets implementation that encodes 2 covert bits at a time requires 4 possible intervals to be assigned to all the transmissions that can occur ("00", "01", "10" and "11"). The same logic is applied to the technique's 4 bit and 8 bit variants. After the intervals list is fully setup, the program will retrieve the bits to be sent from the full string and will, from them, get the corresponding delay from the list.

On the other hand, the Time Replay is implemented with lists of intervals to each possible transmission, where each possible combination of bits to be transmitted can be passed with 3 different delays. So, like the previous technique, but where each interval has 2 redundant possible delays to increase confidentiality. In its implementation, after the delay's matrix

filling, the module will retrieve the bits to be sent from the full phrase and get its list from the matrix. Following that, a random number between 0 and 2 will be generated utilizing a random number generator and its result will give the delay that will be used to pass the desired information to the receptor node.

Since the traffic generator only starts on the thirtieth second of the simulation to guarantee that all the simulation is flowing correctly when the experiments are performed, an additional delay (of 3 symbols) was added as a startup value. When the receiver obtains a packet with this delay, it resets all its variables related to the covert channel and starts a fresh reception. The additional delay was implemented using the OMNeT++'s self message functionality (figure 4.11). This creates a self message to the current node, but this message is created only after the inputted delay. In the treatment of the self message, the packet is redirected to the normal flow of the model, but the delay is already done at this point.

```
    delete msg;
} else if(strcmp(msg->getName(), "covert") == 0) {
    dsme->getMessageDispatcher().sendDoneGTS();
    //dsmeAdaptionLayer.getMessageHelper().sendMessageDown();
} else {
    MacProtocolBase::handleSelfMessage(msg);
}
```

Figure 4.11: Self Message Reception

In the original model, the code will be injected into the MessageDispatcher module of the DSMELayer, more particularly, in the sendDoneGTS method (figure 4.12). In here, the transmitter node will pop messages from the queue to be sent and, in each one, will check the time slot for available remaining time for transmission. This native method was also tweaked in order to account the delay, that is, if the time slot has space for a whole packet transmission plus the maximum amount of delay to be inserted.

```
void MessageDispatcher::sendDoneGTS2(enum AckLayerResponse response, IDSMEMessage* msg) {
    lastMessage = msg;
    lastResponse = response;
    this->dsme.getCovertHelper().insertDelayOnOff(msg);
}


void MessageDispatcher::sendDoneGTS() {
    enum AckLayerResponse response = lastResponse;
    IDSMEMessage* msg = lastMessage;

    LOG_DEBUG("sendDoneGTS");
```

Figure 4.12: Covert Sender method injection

4.4.2 Covert channel receiver

On the receiver side (algorithm 4.2), a verification is also performed to ensure the packets having their transmission interval scanned are originally sent from the node responsible of sending the covert information. Next, the current interval (actual time - last covert packet

reception time) is found and deconstructed using the following formulas, that will allow for the time in milliseconds to be converted into symbols. With this value, it was possible to gather the amount of delay inserted into the packet received (in LIFS constants).

$$TimePayload = (ByteSize \times TimePerPayloadByte) \tag{4.4}$$

4.4: Time of the Payload Formula

$$DelayInserted = \frac{(CurrentInterval - ConstantNoDelayTime - TimePayload) \times 960}{15.36} \tag{4.5}$$

4.5: Delay Inserted Formula

In this formulas:

- CurrentInterval: Actual time - Last covert packet recorded time (in milliseconds)
- ConstantNoDelayTime: The time a packet without payload takes to travel the network (constant 3.104 milliseconds)
- ByteSize: Packet's payload size (in bytes)
- TimePerPayloadByte: Constant achieved through the formula below, with the time each byte of the payload adds to the transmission time (constant 0.032 milliseconds)

With the amount of delay inserted into the message being available, a simple comparison with the default values was sufficient to determine the bit being covertly transmitted in the channel. If the delay embedded into the delay quantity is equal to amount set as the first transmission, the storing variables are reset and the program becomes ready to initiate the reception of the covert information. After a correct read, the program adds to the storage variable the transmitted bit and proceeds to next packet, restarting the process (figure 4.13). Depending on the technique used, the method will search different delay lists in order to retrieve the bit (or set of bits) sent covertly.

```
currentInterval = (millis - lastPacketTime) * 1000.0;
int16_t symbolsTmp = ((((currentInterval - 3.104 - (byteSize * 0.032)) * 960) / 15.36) + 0.5);
if(symbolsTmp == 1){
    rMessageInBits.none();
    rLetterCounter = 0;
    messageToDecode.clear();
} else if(currentInterval < 100){
    if(symbolsTmp == 0){
        messageTemp = diffChars[0];
    }
    if(symbolsTmp == 2+ALPHA){
        messageTemp = diffChars[1];
    }
    myfile.open ("results/writer" + std::to_string(so) + ".csv", std::ios_base::app);
    myfile << (std::to_string(currentInterval) + ",0" + "\n");
    myfile.close();
    char charMessageTmp = messageTemp;
    messageToDecode.append(1, charMessageTmp);
    rLetterCounter++;
    LOG_DEBUG("CHAR: " << messageTemp);
    LOG_DEBUG("MESSAGE:" << messageToDecode);
} else if(currentInterval < 40000) {
    myfile.open ("results/writer" + std::to_string(so) + ".csv", std::ios_base::app);
    myfile << ("0," + std::to_string(currentInterval) + "\n");
    myfile.close();
}

lastPacketTime = millis;
```

Figure 4.13: Decode Covert Information

In the original DSME model, this new module will be called in the MessageHelper component, present in the DSMEAdaptionLayer (figure 4.14). In the handleDataIndication method, responsible for dealing with a data message when it is sent to the node in question, a single call to the module is performed, sending the payload as a parameter in order to detect the message's source node.

```
void MessageHelper::handleDataIndication(mcps_sap::DATA_indication_parameters& params) {
    LOG_DEBUG("Received DATA message from MCPS.");

    this->dsmeAdaptionLayer.getCovertHelperReceiver().readCovertOnOff(params);

    this->dsmeAdaptionLayer.getGTSHelper().indicateReceivedMessage(params.msdu->getHeader().getSrcAddr().getShortAddress());
    receiveIndication(params.msdu);
    return;
}
```

Figure 4.14: Covert Receiver method injection

An implementation particularity was dealing with the end of a time slot, because the last packet in the time slot would still contain the covert bit, but as it never left the sender, it never reached the receiver, and the start of the next time slot would have a ridiculous interval, not corresponding to any of the configured intervals. To correct this, a mechanism was implemented that, in the event of the interval between packets being too high, the current packet is ignored by the receiver and the sender, when it also detects such a large interval, it replays the last transmission so that the string of information stays cohesive.

In order to furthermore create a perfect environment to ensure that the simulation data is reliable, a static scheduling of time slots was used, this one already being disponibilized by the developers of the model in use.

4.4.3 Pseudocode

Algorithm 4.1 Sender's algorithm:

$Delay \leftarrow 1$
$CharToSend \leftarrow MessageToSend[CharCounter]$
$BitToSend \leftarrow GetBitsetFromChar(CharToSend)[BitCounter]$
if $First$ is true **then**
  $StartDelay(CurrentTime + 3 \times Delay)$
else if $BitToSend$ is 0 **then**
  if $BitCounter$ is greater or equal than 8 **then**
    $CharCounter \leftarrow CharCounter + 1$
    $BitCounter \leftarrow 0$
  end if
  $SendMessage$
else if $BitToSend$ is 1 **then**
  if $BitCounter$ is greater or equal than 8 **then**
    $CharCounter \leftarrow CharCounter + 1$
    $BitCounter \leftarrow 0$
  end if
  $StartDelay(CurrentTime + 2 \times Delay)$
  $SendMessage$
end if

Algorithm 4.2 Receiver's algorithm:

$CurrentTime \leftarrow CurrentSimTime$
$CurrentInterval \leftarrow (CurrentTime - LastPacketTime) \times 1000$
$ByteSize \leftarrow MessageTotalSymbols - HeaderSize$
$Delay \leftarrow \frac{CurrentInterval - 3.104 - (byteSize \times 0.032) \times 960}{15.36}$
if $CurrentInterval$ is smaller than $MaxTime$ **then**
  if $Delay$ is 0 **then**
    $BitReceived \leftarrow 0$
    $MessageReceived \leftarrow MessageReceived + BitReceived$
  else if $Delay$ is 2) **then**
    $BitReceived \leftarrow 1$
    $MessageReceived \leftarrow MessageReceived + BitReceived$
  else if $Delay$ is 3 **then**
    $MessageReceived \leftarrow EmptyString$
  end if
end if
$LastPacketTime \leftarrow CurrentTime$

4.4.4 Work Methodology

After a successful implementation, an algorithm to retrieve analysable data from the simulations had to be created. Firstly, an input/output script in C++ was embedded into the covert channel implementation (on both the sender and the receiver ends) that retrieved several raw data, including the interval time between consecutive packets and the amount of information being transmitted (both covertly and openly) by the node responsible for sending

covert information. Then, with the assistance of the R programming language, a thorough analysis will be performed, with several graphs and other visual aids being generated in the process. These will help immensely in the interpretation of the data extrapolated from the simulations and in transferring that knowledge obtained to the thesis' conclusions.

4.4.5 Validation

To guarantee that the covert channel implementation performed and described earlier is fully functional, several conditions were verified to validate that the covert channel is working correctly. The main validation to be performed is the integrity of the data, that is, if the information covertly transmitted reached the receptor node without flaws. Furthermore, the ability to retrieve information (algorithm of section 4.4.4) of the ran simulations is to be validated also.

Referring to the first validation, the information defined in the code to be transmitted covertly is a sequence of 208 bits (figure 4.15).

```
void MessageDispatcher::initialize(void) {
    this->NOW.initialize(&(this->dsme.getPlatform()), &IDSMEPlatform::getSymbolCounter);
    messageToEncode = "0110100101101110011001100111011110110011001101101011000010110001101100001011011110010000001100011011011
    //messageInBits = getBitsetFromChar(messageToEncode[letterCounter]);
    currentACTElement = dsme.getMAC_PIB().macDSMEACT.end();
    std::ofstream myfile;
```

Figure 4.15: Message (partial) to be sent in the code implementation

After several seconds of simulation, the obtained result was printed by the OMNeT++ log system as seen in figure 4.16.

```
INFO (Ieee802154NarrowbandScalarRadio)Net802154.host[1].wlan[0].radio:Changing radio transmission state from IDLE to UNDEFINED.
** Event #224509  t=70.078688  Net802154.host[0].wlan[0].mac (DSME, id=31)  on appData-280 (inet::Packet, id=314303)
INFO:70.078688   1: Received valid frame     (44 B) (inet::SequenceChunk) length = 44 B(2|1|39|DATA|122) [0%, 0%, LQI 255]
INFO:70.078688   1: sending ACK
INFO (Ieee802154NarrowbandScalarRadio)Net802154.host[0].wlan[0].radio:Radio mode changed from RECEIVER to TRANSMITTER.
INFO (Ieee802154NarrowbandScalarRadio)Net802154.host[0].wlan[0].radio:Changing radio reception state from IDLE to UNDEFINED.
INFO (Ieee802154NarrowbandScalarRadio)Net802154.host[0].wlan[0].radio:Changing radio transmission state from UNDEFINED to IDLE.
INFO (Ieee802154NarrowbandScalarRadio)Net802154.host[0].wlan[0].radio:Changing radio received signal part from WHOLE to NONE.
INFO:70.078688   1: Received DATA message from MCPS.
INFO:70.078688   1: AAAAAAAAAAAAAAAAAAAAAAAAAAAA 120 2
INFO:70.078688   1: SYMBOLS ACT: 120
INFO:70.078688   1: SYMBOLS PRED: 2
INFO:70.078688   1: 70.078688
INFO:70.078688   1: LAST: 70.0743
INFO:70.078688   1: NOW: 70.0787
INFO:70.078688   1: INTERVAL: 4.416
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1: CHAR: 1
INFO:70.078688   1: MESSAGE:0110100101101110011001100111011110110011001101101011000010110001101100001011011110010000001100011011011000010110100101100001011011000010000001000000100000
INFO:70.078688   1:
INFO:70.078688   1:
INFO:70.078688   1:
```

Figure 4.16: Message received in the simulation

Using a web tool designed to compare strings and check for their equality, the confirmation that the simulation covert channel result was identical to the expected is presented in figure 4.17.

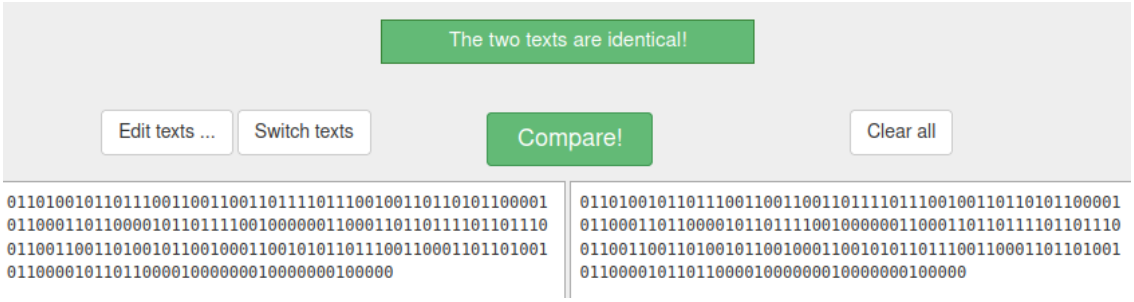


Figure 4.17: Comparison between message sent and received

Regarding the second validation, the generated output *.csv* file contained successfully extracted information of the covert channel, with metrics referring to the amount of information covertly transmitted, time of delays between packets and between time slots (figure 4.18).

	A	B	C	D	E	F
1	Time	CovertPackets	AllPackets	BytesSent	PacketInterval	GTSInterval
2	30.758272	1	2	0	3.136000	0
3	30.761408	2	3	0	4.416000	0
4	30.765824	3	4	0	4.416000	0
5	30.770240	4	5	0	0	966.016000
6	31.736256	4	6	0	3.136000	0
7	31.739392	5	7	0	4.416000	0
8	31.743808	6	8	0	3.136000	0
9	31.746944	7	9	0	3.136000	0
10	31.750080	8	10	1	4.416000	0
11	31.754496	9	11	1	3.136000	0
12	31.757632	10	12	1	4.416000	0
13	31.762048	11	13	1	0	957.248000
14	32.719296	11	14	1	4.416000	0
15	32.723712	12	15	1	3.136000	0
16	32.726848	13	16	1	4.416000	0
17	32.731264	14	17	1	4.416000	0
18	32.735680	15	18	1	4.416000	0
19	32.740096	16	19	2	3.136000	0
20	32.743232	17	20	2	0	959.104000
21	33.702336	17	21	2	3.136000	0

Figure 4.18: File exported with data to analyse

With both these conditions validated, an argument regarding the implementation and the success of it was considered resolved, with the covert channel being considered functional and ready to be further exploited and analysed thoroughly.

Chapter 5

Performance Analysis of Timing Covert Channels in the IEEE 802.15.4

This chapter presents the performance evaluation of several timing covert channel techniques over the GTS service of the IEEE 802.15.4 DSME protocol.

5.1 Metrics and Information Sources

To fully test the covert channel capabilities, the major point is naturally the integrity of the information. If the data being covertly sent is not correct upon arrival, the covert channel is not useful and defeats its main purpose. As we are considering covert channels of the "timing nature", timeliness is fundamental. For this the GTS service of the IEEE 802.15.4 constitutes the ideal medium for the setup. To analyse the performance of the techniques, we considered the following metrics:

- Covert channel concealment: detectability of the subliminal channel by a network warden, will be investigated based on transmission patterns and how often they repeat themselves (predictability).
- Transmission efficiency: impact of the channel presence in the network, in terms of transmission delays and overall number of packets sent with or without a covert channel present.
- Capacity: quantity of information that can be transmitted in a certain amount of time, to be compared not only in different environments, but also considering different covert channel techniques.

All of this metrics will be retrieved from the simulations performed on the modified model by an algorithm embedded that extracts the desired information and stores it in a file in an orderly fashion. With this values, graphs and other visual aids will be crafted to improve and further assist the comprehension of the impacts that are inflicted on the network by the covert channel.

5.2 Simulation Setup

The implementation performed in the openDSME model [Köstler et al. 2016] was run in the OMNeT++ event simulation framework, supported by the INET platform. We consider a star network topology with 11 nodes, organized in a circular shape, where the distance between the exterior nodes and the node[0], i.e. the PAN Coordinator, is approximately similar. This node, being placed at the center, acts as sink as presented in figure 5.1. All the data sender nodes will be scheduled according to an incrementing schedule (node[1] gets slot 1, node[2] gets slot 2, etc.). This data begins being generated in the 30th second of the simulation to ensure the already flowing state. From this second, the simulation will run for 200 seconds, recording several metrics of interest described below. The default setup of the simulation (unless changed for a metric comparison) was a packet payload length of 1 byte, a rate of 0.01 seconds per packet created and a channel bitrate of 250k bits per second. In terms of the protocol standard metrics, the beacon order is set to 6, multi superframe order is 4 and the superframe order is also 4.

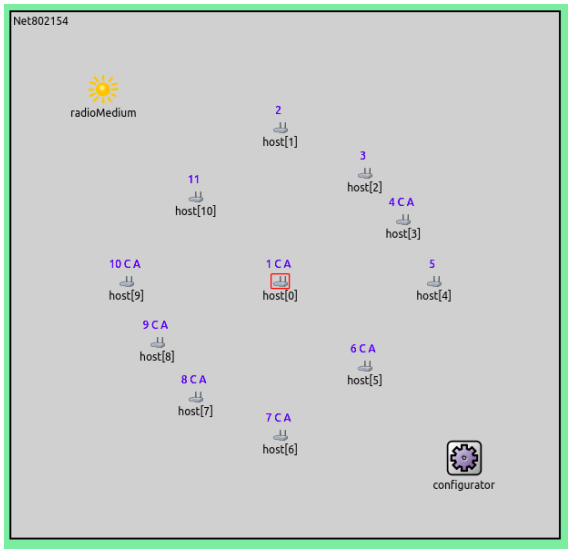


Figure 5.1: OMNeT++ Simulator Model

From each simulation the retrieved metrics are the number of bytes transmitted, the number of packets from which the covert bit was read (important since the last packet in each guaranteed time slot is enable to transmit a readable delay), the information sent covertly (in bytes), the interval between packets in the guaranteed time slot and the interval between time slots in which the covert node is transmitting.

5.3 ON/OFF CC Technique

We consider a single ON/OFF covert channel has been established on node[1] to exfiltrate data, corresponding to the slot indicated in the figure 5.2.

5.3.1 Impact of SO upon the Covert Channel

We consider $MO = SO$ for all test cases and $BO = SO+1$, so to guarantee that we allocate two superframes per multi-superframe. This is enough to schedule all node's transmissions

in a single multi-superframe, with one GTS slot for each node, as presented in figure 5.2.

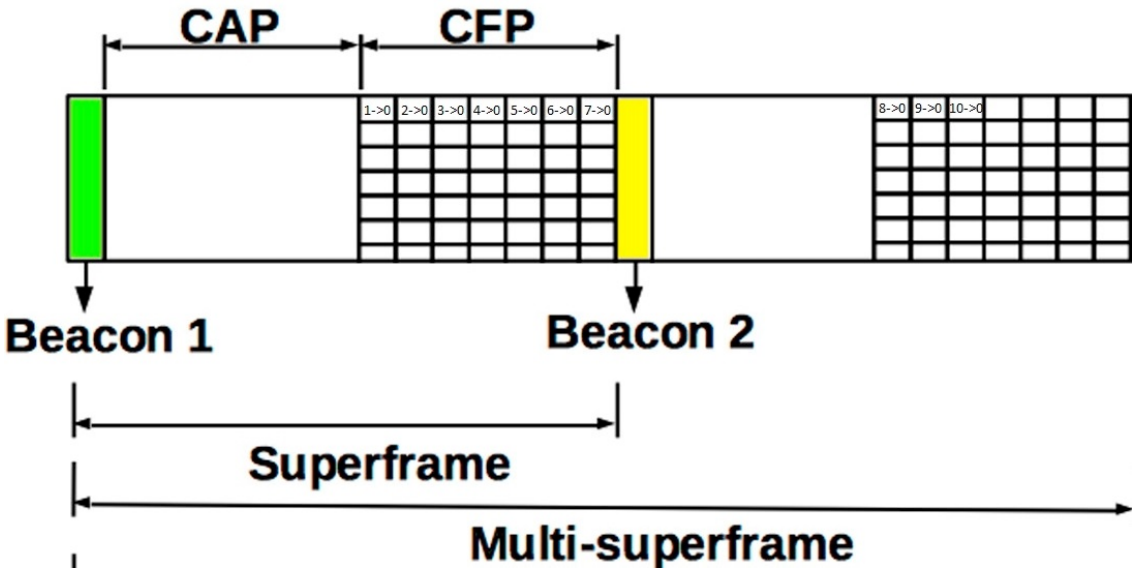


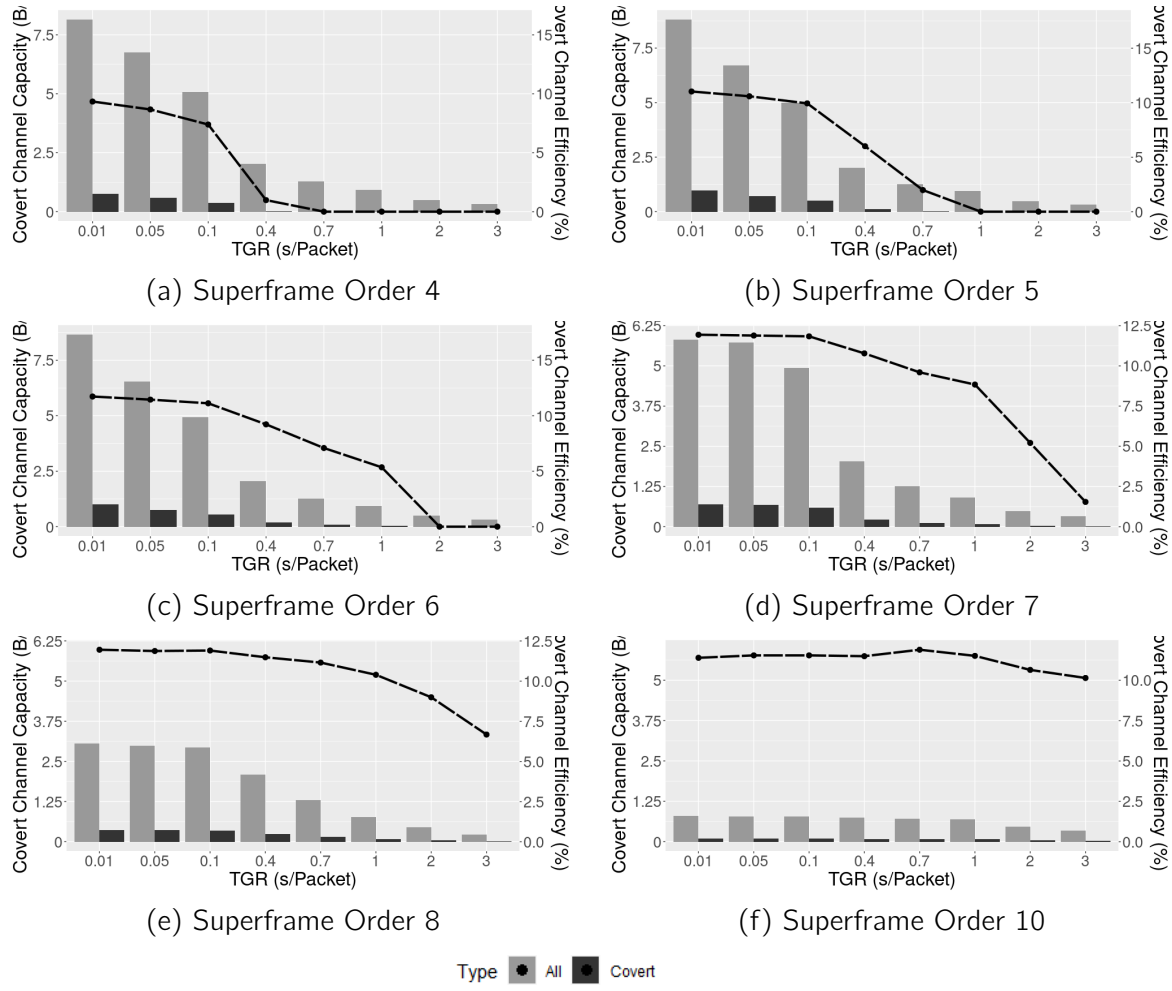
Figure 5.2: Superframe schedule (based of [Queiroz et al. 2017])

To evaluate the behaviour of the covert channel we consider varying traffic rates from 0.01 to 3 seconds, and the MO, SO, BO combinations represented in table 5.1.

	SO	MO	BO
A	4	5	6
B	5	6	7
C	6	7	8
D	7	8	9
E	8	9	10
F	10	11	12
G	12	13	14

Table 5.1: Network DSME configurations

The graphs in the figure 5.3 present a comparison between the overall traffic transmitted by node[1] (in red) and the covert channel traffic in KB/s in blue for each setting at different traffic generation rates. As expected, the covert channel traffic is directly dependent of the traffic generation rate. Being a timing covert channel, the more traffic the node generates the more information can be passed in the covert channel, however, SO plays an important role in the network, particularly in defining the time slot length and periodicity, which plays a critical role in establishing the timing covert channel performance.



(a) Superframe Order 4

(b) Superframe Order 5

(c) Superframe Order 6

(d) Superframe Order 7

(e) Superframe Order 8

(f) Superframe Order 10

Figure 5.3: Regular vs Covert channel traffic in several Superframe Orders

Interestingly, the covert channel efficiency in regards to the amount of legitimate traffic sent, increases as the time slot length increases. It is clear that for lower SOs ($SO = 4$) only a small portion of the amount of traffic sent is relevant for covert traffic, an amount which increases with the network SO. This is tightly connected with time slot length. The reduced bandwidth of the $SO = 4$ time slot (15.36 ms) forces the ON/OFF covert channel to resume service on the next superframe, discarding the last sent packet. Naturally, the amount of packets discarded will be higher the more resets the covert channel mechanism has to perform. At the most, the ON/OFF covert channel was able to achieve a performance of 11.9% in regards to the overall legitimate traffic transmitted, in other words, at the most, approximately 12% of legitimate traffic could be effectively used to convey the covert information (for $SO = 6$ and above). Such efficiency decreases, as expected, as the traffic rate decreases. However, for larger SO the network can keep a good efficiency even when the amount of traffic decreases to 1 packet every 3 s. ($SO = 10$, above 10%). In what follows, this behaviour is explained in depth.

It is clear that choosing a good combination between SO and the traffic-rate is fundamental. For lower SO values e.g. $SO = 4$ and $SO = 6$, as traffic decreases, no covert information can be passed at all. This is because there are not enough packets accumulated

in each slot for transmission, and instead, the generated packet is dispatched almost immediately in the first opportunity, while the next superframes remain unoccupied until a new packet is generated. However, for higher SO values, the channel can better cope with lower traffic as, due to its lower slot periodicity (longer superframes), packets get accumulated awaiting service. Also given the higher slot length, more packets can be transmitted in a single slot.

This GTS' bursty traffic property can be easily noticed in figure 5.4 which compares $SO = 4$ with $SO = 10$ at different traffic generation rates.

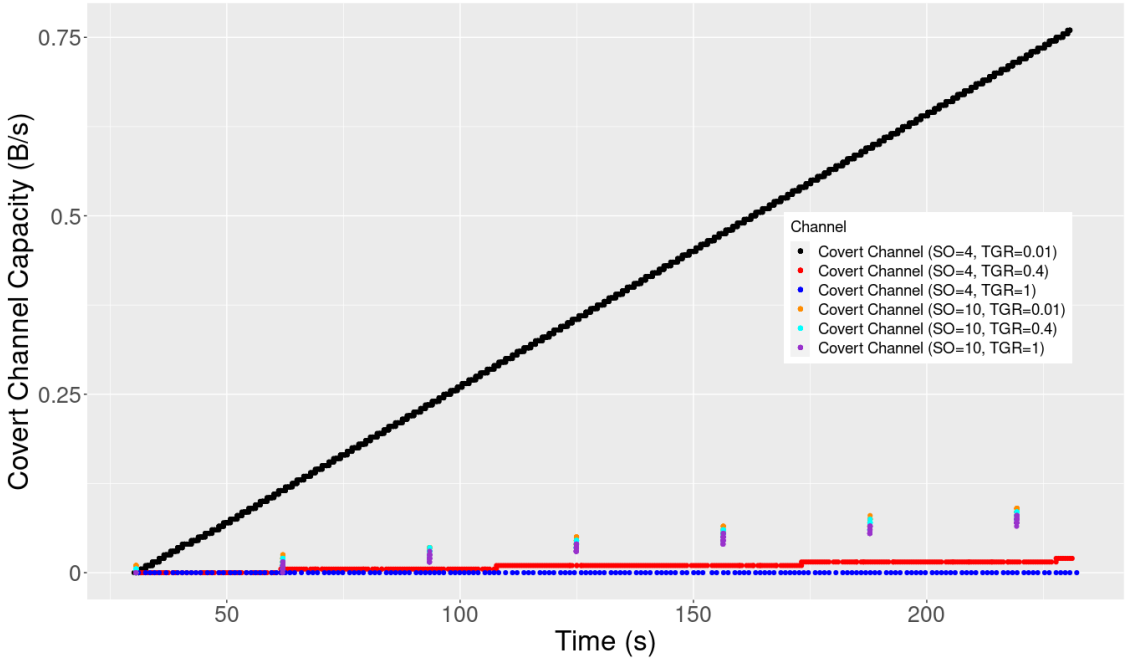


Figure 5.4: Covert Bytes transmitted over time by Superframe Order and Traffic Generation Rate

As shown, as simulation time progresses, for lower amounts of traffic ($TGR = 0.4$), $SO = 4$ cannot cope with covert transmissions. Interestingly, it greatly surpasses the higher SO values ($SO = 10$) for high volumes of traffic. This is because of its slot frequency, which although with smaller slots, are more frequent, resulting in more transmission opportunities. For such higher traffic volumes, higher SOs on the other hand, tend to decrease covert channel capacity, as the introduced delay becomes quite significant, lowering the overall covert traffic sent during the same period. However, they still manage to support covert traffic due to their bursty behaviour, which lets them accumulate packets for the next timeslot. Another important component to consider is slot length.

Hence, regarding overall covert channel capacity, we now compare the overall results for different traffic rates and SO combinations. As shown, due to above mentioned, SO/traffic-rate balance, $SO = 6$ seems to provide the best results for $TGR = [0.01, 0.05]$, achieving 187 transmitted bytes in the covert traffic. However, for a traffic generation rate of 0.1s, $SO = 7$ seems to be more suited as presented in figure 5.5, achieving 119 transmitted bytes. And $SO = 8$ achieves the best covert channel capacity at $TGR = 0.4$.

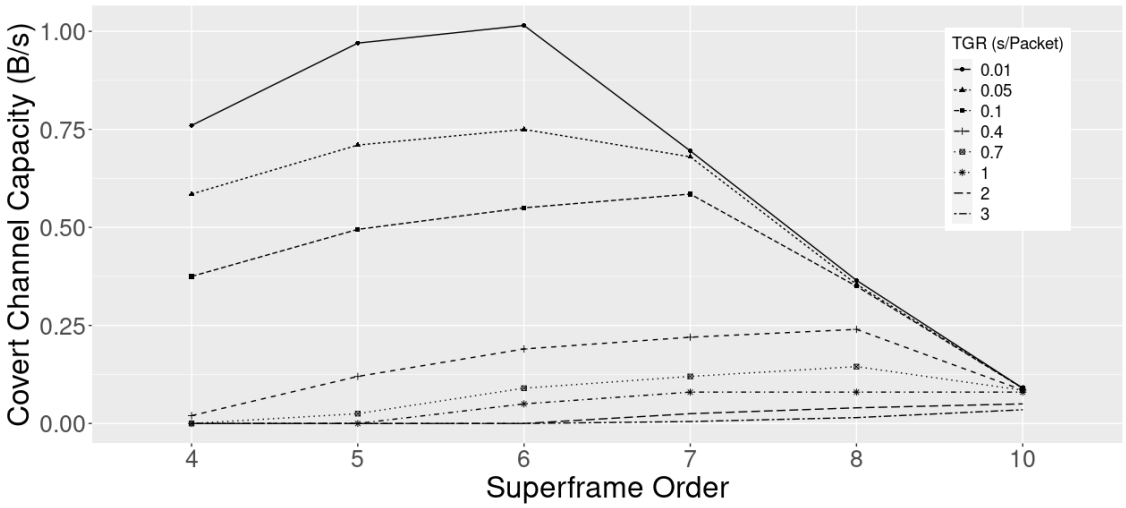


Figure 5.5: Total Covert Bytes transmitted by Traffic Generation Rate and Superframe Order

This is related to the slot length. Although for $SO = 6$, there is a higher delay between slots than for lower SOs, its larger slot size allows for more traffic to be transmitted per slot, which increases covert channel capacity. This is until traffic generation rate slows down to $TGR = 0.1s$. Then, part of the slot bandwidth becomes wasted and $SO = 7$ achieves better results at the expense of higher delay. Its larger time slot size of 122.88 ms is enough to better accommodate the generated traffic, i.e. minimizing wasted bandwidth. Figure 5.6 depicts the delay between the covert channel time slots for different SOs. Delay is even worsened as two superframes are needed to accommodate all nodes, which was a network requirement.

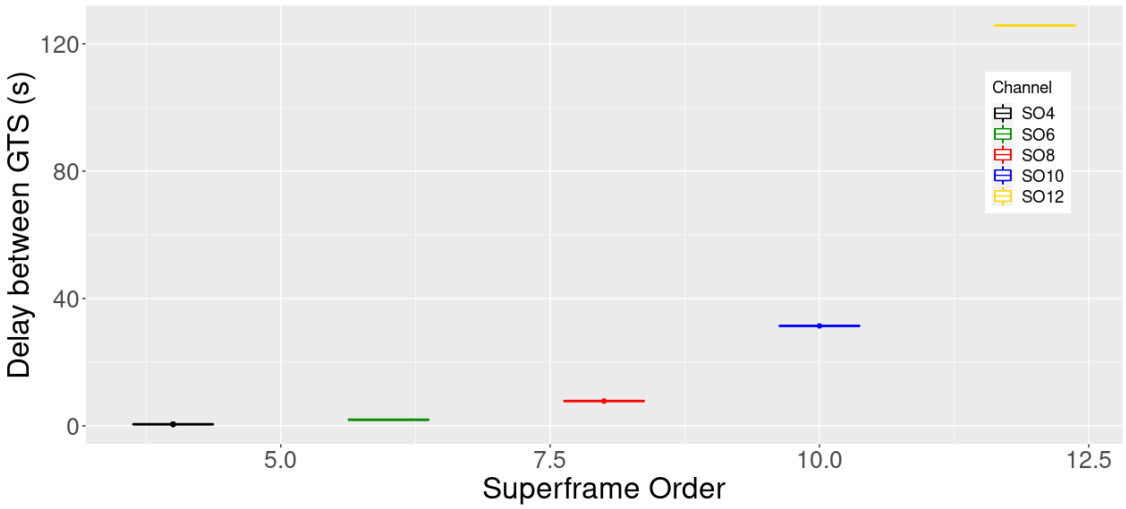


Figure 5.6: Interval between slots by Superframe Order

Interestingly, higher SOs, on the other hand, result in lower covert traffic capacity, as already explained, and appear less sensitive to variations in traffic generation rate. For $SO = 10$, in $TGR = [0.01, 0.7]$, the covert channel throughput is almost independent of the traffic rate, and for $SO = 12$ the covert channel capacity is approximately the same

for all traffic generation rates. This is consequence of the large time slot length's capacity, which is able to allocate and transmit all of the generated traffic during the multi-superframe period. Hence, its covert channel capacity and efficiency remain approximately constant.

As presented, the timing covert channel performance is highly dependent on the overall balance between slot length, slot periodicity, specified by the SO value, and traffic generation rate.

Using low SO values with low packet generation rates can effectively defeat such timing covert channel implementations, or greatly reduce their performance, while still guaranteeing low legitimate traffic delays. This is, of course, at the expense of higher energy consumption, as superframes are much more frequent. On the other hand, higher SO values, although they are unable to provide high covert channel throughput, they can accommodate greater shifts in traffic generation rate even when applications generate very little traffic. Interestingly, for very low traffic rates, it seems one could expect $SO = 4$ would bring the best results due to the high slot frequency. However, this is not the case. As SO reduces, slot length also shortens, thus decreasing the available bandwidth (figure 5.7). While less packets are transmitted per timeslot, reducing covert channel capacity, frequent transmissions lead to more resets of the ON/OFF covert channel mechanism, which causes the last packet to be dismissed, decreasing covert channel efficiency.

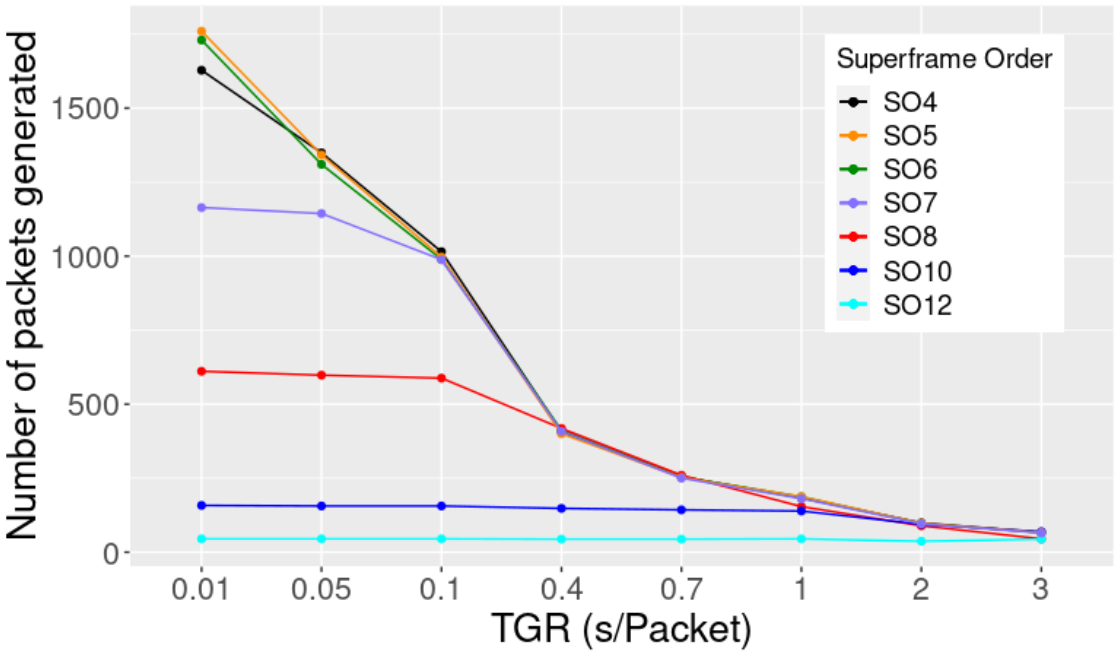


Figure 5.7: Packets transmitted by Superframe Order and Traffic Generation Rate

5.3.2 Impact of Packet length

While keeping the same network setup, we now analyse the impact of different packet lengths on the covert channel performance. Due to the limited available bandwidth established by the correspondent SO value, it is expected that packet length also impacts the overall covert channel performance. Indeed, as shown in figure 5.8 for $SO = 6$, this impact is particularly visible for lower traffic generation periods.

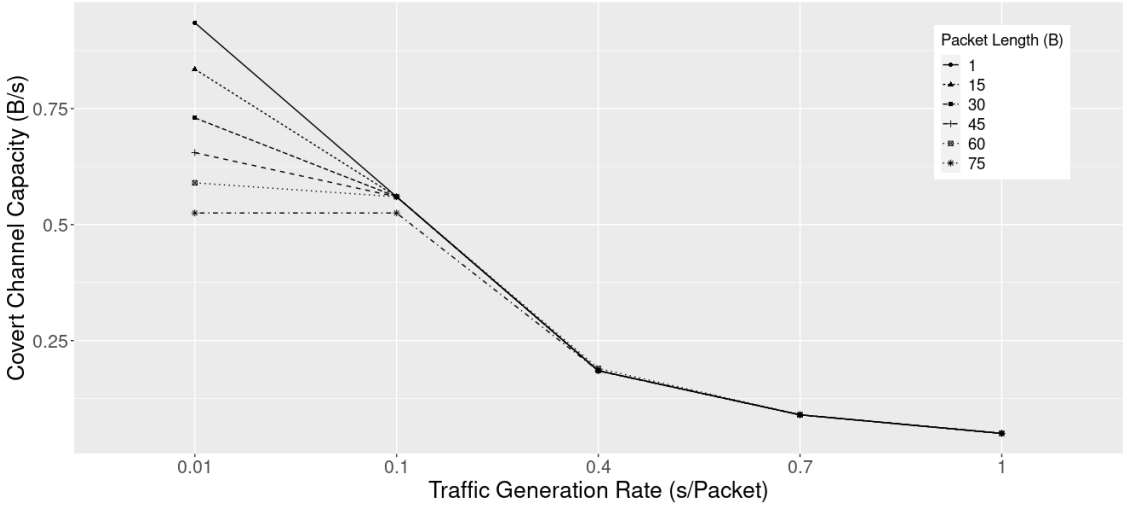


Figure 5.8: Total Covert Bytes transmitted by Packet Length and Traffic Generation Rate for SO 6

For $TGR = 0.01s$, an increase on packet length dramatically decreases the covert channel capacity, as the larger each packet, the less traffic can fit inside a time slot, thus decreasing the available traffic to rely on for the timing ON/OFF covert channel. As the amount of traffic decreases, the longer packets decrease their impact upon the covert channel capacity. This is because the number of packets transmitted in each slot is already so reduced, that the longer packet size does not significantly change the amount transmitted per slot.

The effect of increased packet lengths on covert channel capacity is particularly worse for lower SOs, as shown in figure 5.9 for $TGR = 0.01s$. The already shorter time slots, will now further reduce service by half, for the case of 75 bytes packet length. The dramatic decrease of packets effectively transmitted naturally reduces the capacity of the timing covert channel as already analysed in the previous section. However, for larger SO, the effect of packet length is negligible, even at such high traffic generation rates. This immunity is related to the large available time slot size, which is more than sufficient to accommodate all that generated traffic.

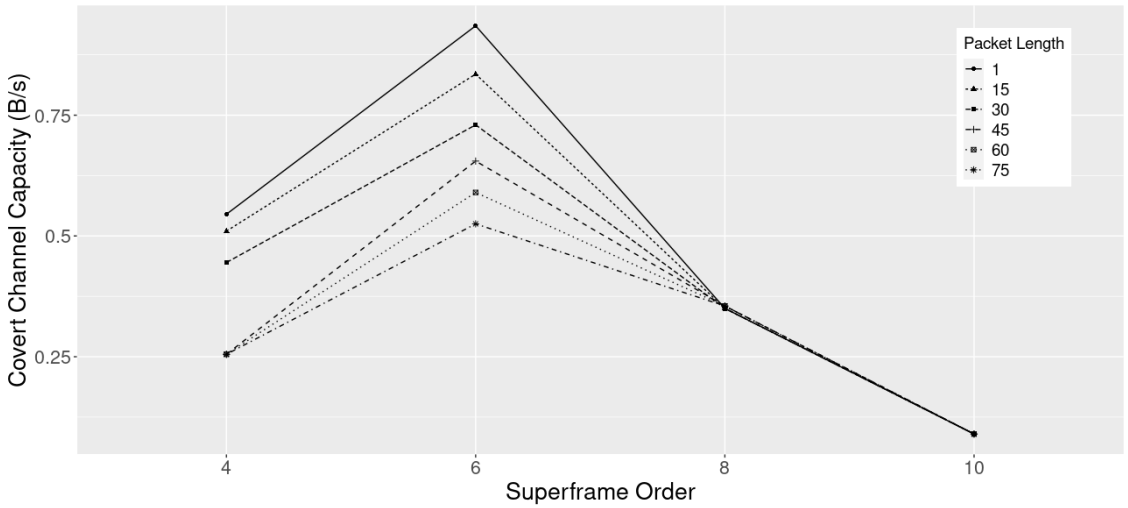


Figure 5.9: Total Covert Bytes transmitted by Superframe Order and Packet Length

5.3.3 Impact of CAP Reduction

The introduction of the CAP Reduction with the new DSME MAC behaviour came with a fundamental improvement, that being the highly increased number of GTS slots available, replacing the CAP section of the superframes, resulting in an increase of GTS service capacity. To analyse the effect of CAP reduction upon the timing channel, we now consider a total of 15 nodes that are able to transmit time-constrained information using GTS mechanisms. As in the analysis above, only one node is compromised and implements a binary timing covert channel. To fit all the nodes in the multi-superframe, we use $MO = SO + 2$ in order to encompass at least three superframes, i.e. four due to protocol limitations as presented in Fig. 5.2. We run several simulations for each network setting as shown in Tab. 5.1 Set 2 and Set 3, with and without CAP reduction. For each, we vary the traffic generation rate and analyse the covert channel capacity. Fig. 5.10 presents the results with and without CAP reduction.

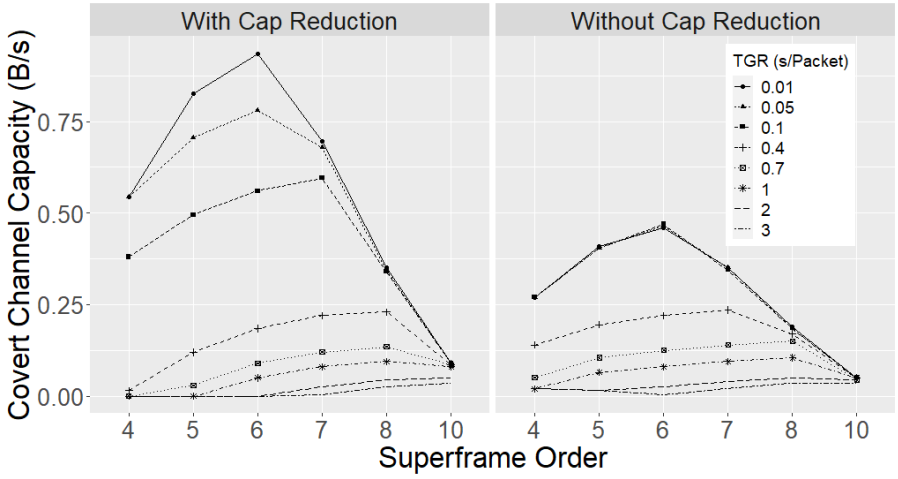


Figure 5.10: CC Capacity by Superframe Order shifting MO settings and CAP Reduction

As presented in Fig. 5.10, with CAP reduction disabled, covert traffic capacity is limited by the greater delay between each transmission opportunity i.e. GTS timeslot periodicity, imposed by the higher MO. Encompassing more superframes in the multi-superframe increases delay for the overt traffic. By turning on the CAP Reduction feature, it increases the available number of GTS slots. This naturally grants GTS service to these nodes with lower MOs, as more can now fit into a single superframe. Hence, the fundamental factor at play is indeed the network throughput per MO. Therefore, it is expected that as the overt traffic increases, the underlying cover traffic follows the same pattern. This is true throughout the SO range as shown in Fig. 5.10 for frequent traffic. For $SO = 6$, for instance, at $TGR = 0.01s$ the covert channel capacity is doubled in regards to the disabled CAP reduction MO setting. This is because during the same period of the MO, with CAP reduction we have two transmission opportunities as shown in Fig. 5.2 (slot marked in yellow) as opposed to one with the other settings. For lower amounts of generated overt traffic i.e. below 0.4s, we see only a slight increase in the covert channel capacity for SO above 6 when no CAP reduction is used. This is because the generated traffic is now so sparse, the delay introduced by the longer MO setting without CAP reduction, helps packing more overt traffic in each timeslot. This increases the Covert Channel Capacity. But for lower SOs, i.e. $SO = 4$ and $SO = 5$, the Covert Traffic Capacity improves significantly when no CAP reduction and higher MO is used, for low amount of generated traffic. In this case, as we turn CAP reduction on, we can further reduce the period between each GTS slot i.e. decreasing MO, as two superframes per multi-superframes suffice. This shortened delay results in a faster transmission of the generated packets, which in turn decreases the probability of multiple transmissions in the same time slot, needed to support the timing channel. Hence, the decrease in Covert Channel capacity with CAP reduction. This is the same behaviour we noticed in the Superframe Order Impact subsection. Conversely, when CAP reduction is disabled, the higher MO allows more overt traffic in each GTS slot, which significantly increases the CC Capacity.

5.4 Analysis of Additional CC Techniques

To better understand the performance limits of the timing CC over this protocol, other relevant techniques were analysed. The Time Replay and the L-Bits to N-Packets techniques provide additional efficiency via improved coding of the covert data, achieved by enabling additional intervals to code 2, 4 and 8 bits at a time.

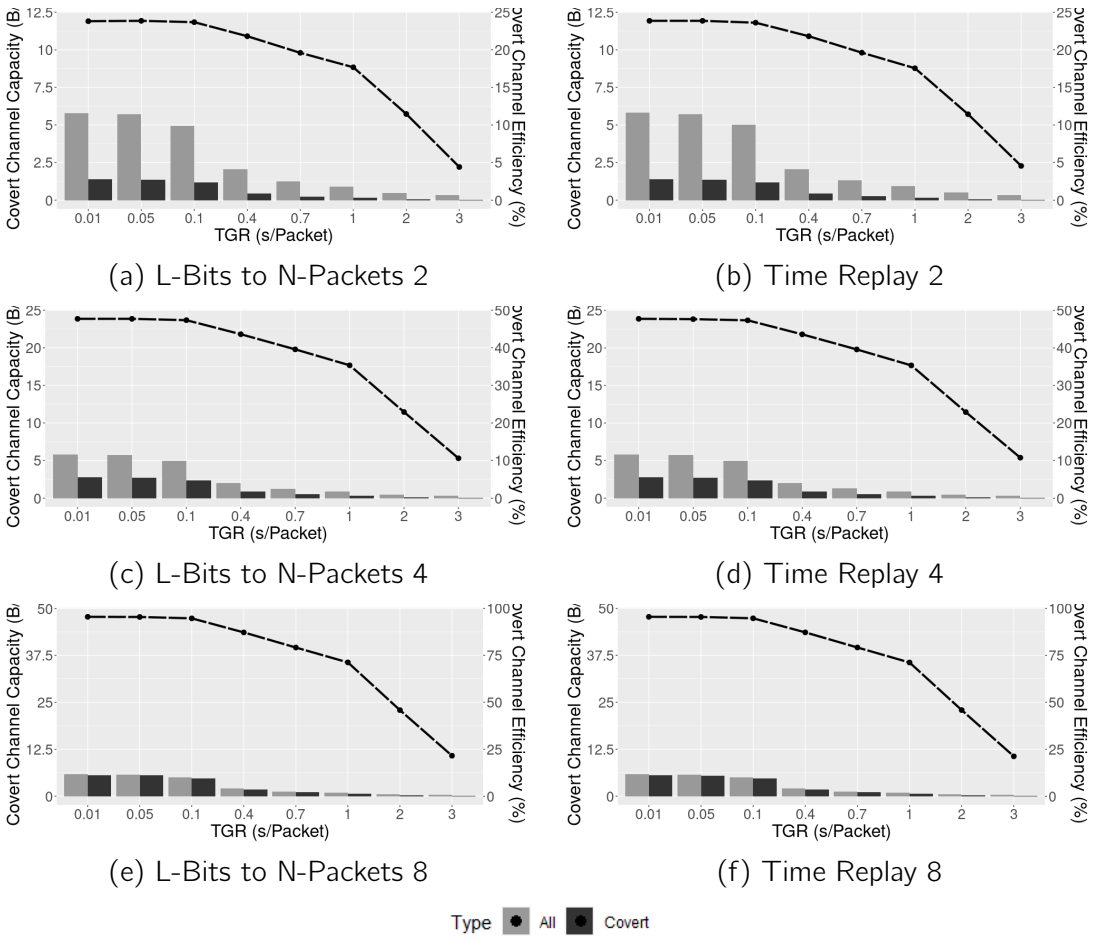


Figure 5.11: CC Techniques capacity and efficiency comparison for Superframe Order 7

In figure 5.13, from (a) to (f), a comparison between the two techniques is presented. The tests included a thorough comparison using superframe order as a variable, from which the data presented in figure 5.11 is a summary comparing only the best configurations achieved from each technique. An evident difference between them is the amount of covert data being transmitted, with this one being significantly higher when the amount of bits being passed in each transmission is also larger. With this, the covert channel efficiency (information passed through the regular channel / information passed through the covert channel) is also higher, with the techniques transmitting 8 bits of covert information per packet achieving efficiency rates of 96%.

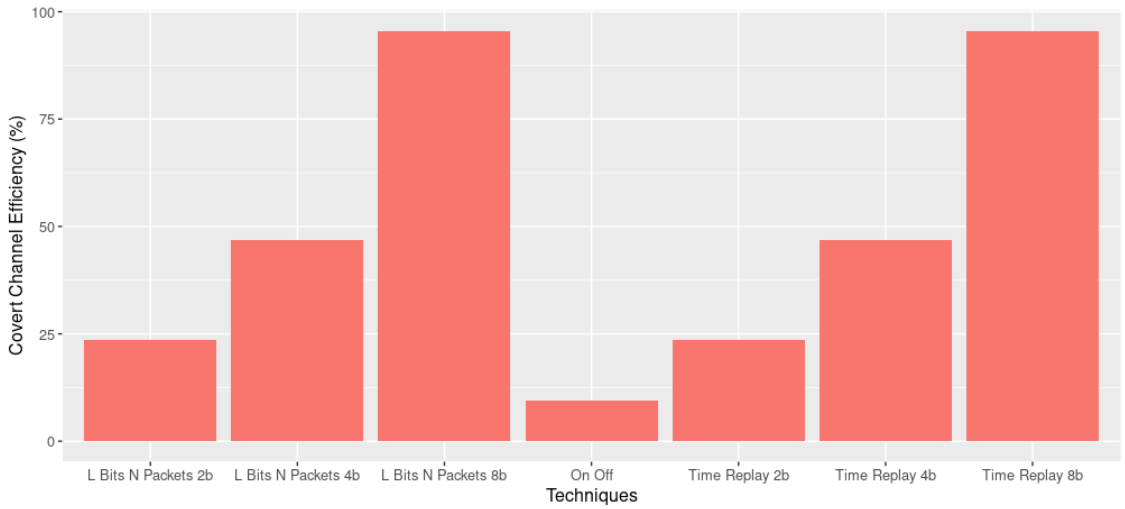
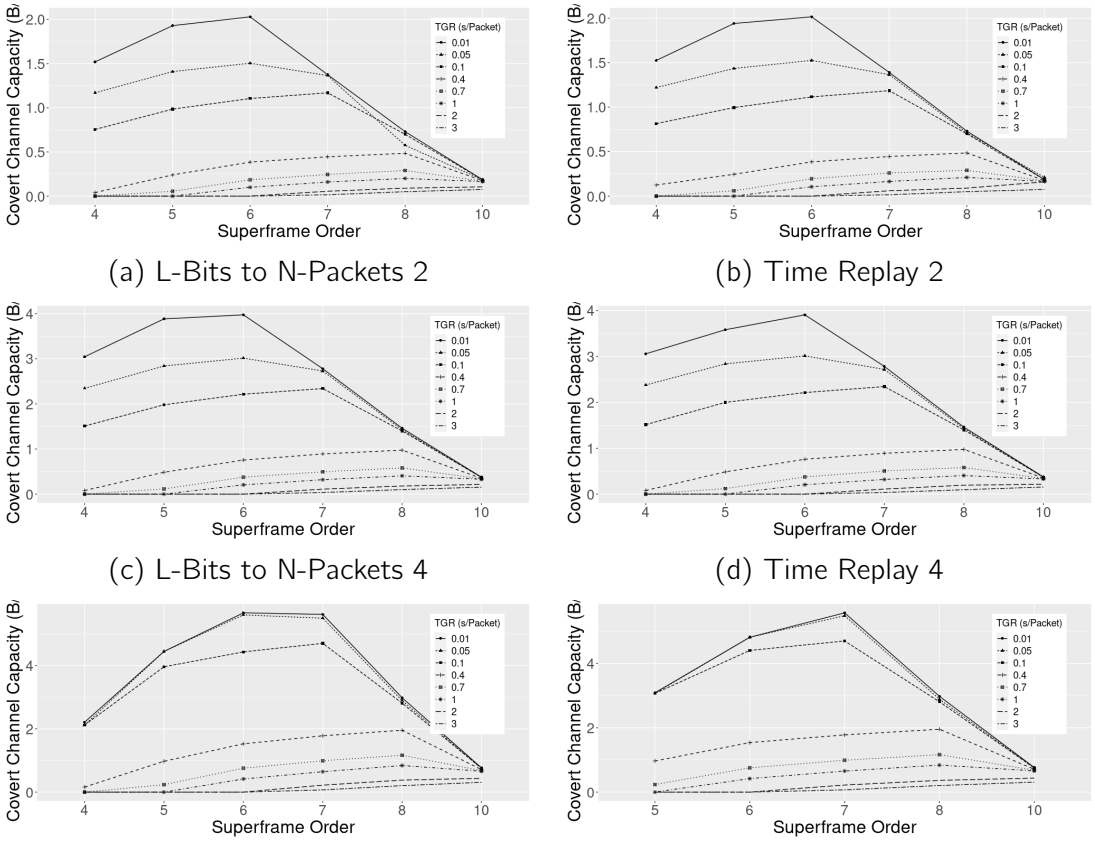


Figure 5.12: CC Techniques efficiency comparison

Comparing all techniques in regards to CC efficiency, as shown in figure 5.12, the 8 bit techniques present a clear encoding only a singular bit with every transmission, is the weakest covert channel analysed in this Thesis, possessing an efficiency of little over 10%.

Regarding CC capacity, as presented in figure 5.13, the techniques that encode 8 covert bits per CC interval present the highest capacity. The abundance of traffic in the network allows for a maximization of the covert traffic quantity and, therefore, the highest recorded values are related to a traffic generation rate of 0.01 seconds per packet generated. Comparing superframe orders, a conclusion can be reached that an extremely high or low superframe order can negatively impact the transmission quantity due to the reduced size of the guaranteed time slots (lower superframe orders) and to the very high period that separates time slots that transmit covert traffic. Generally, from the graphs observed, the configuration to fully maximize the covert channel to its full capacities would be the usage of an 8 bit technique (either Time Replay or L-Nits to N-Packets), with Superframe Order 7 and a Traffic Generation Rate of 0.01 seconds. Again, higher SO work better with reduced TGR as the larger time slots accommodate several packets. On the contrary, with reduced TGRs, low SO do not support adequate number of underlying traffic to maintain a high capacity CC channel.

To further increase the understanding of how the techniques differentiate between them, the graph in figure 5.14 presents the time interval between consecutive covert transmissions, that is, the amount of delay inserted into each packet to transmit the desired information. As expected, the delay inserted into the packets increments as the amount of encoded CC information also increments. The difference between techniques and their functioning is evident here, where it is possible to observe the impact if the redundant intervals of the Time Replay techniques in comparison to the L-Bits to N-Packets variants.



(a) L-Bits to N-Packets 2

(b) Time Replay 2

(c) L-Bits to N-Packets 4

(d) Time Replay 4

(e) L-Bits to N-Packets 8

(f) Time Replay 8

Figure 5.13: CC Techniques capacity comparison for different Superframe Orders and Traffic Generation Rates

As observed for the On/Off technique, the principal and most determining factors for the performance of the covert channel were the superframe order and the traffic generation rate. By relying only on the studied network settings one can severely limit and mitigate the covert channel or maximize its capacity. For the later, the recommendation is to keep a very high load of traffic in the network in order to allow for the highest number of underlying legit transmissions. Combining that with a mid-range superframe order ($5 \leq SO \leq 7$) and the network is in prime conditions to maintain a timing covert channel with a high capacity of information sharing. On the contrary, a configuration developed to restrain the usage of timing covert channels in a network depends on the traffic restrictions in place. If traffic can be kept to a low rate ($0.7 \leq TGR$), then a superframe order of 4 will not allow for a timing covert channel to successfully pass information covertly in the network. If such traffic limitations cannot be placed, then a full stoppage of covert traffic is not guaranteed, but its information sharing capabilities can be severely impaired by a configuration featuring an extremely high superframe order ($10 \leq SO$), guaranteeing that the slots will be very large but simultaneously highly spaced-out time wise.

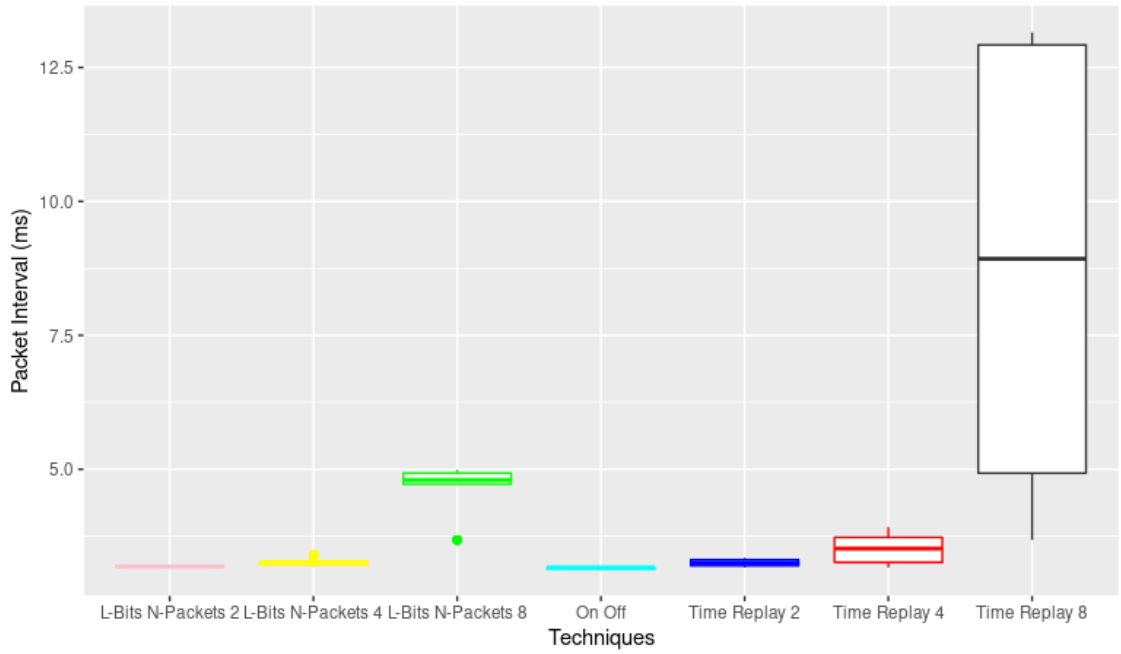


Figure 5.14: CC Techniques intervals comparison

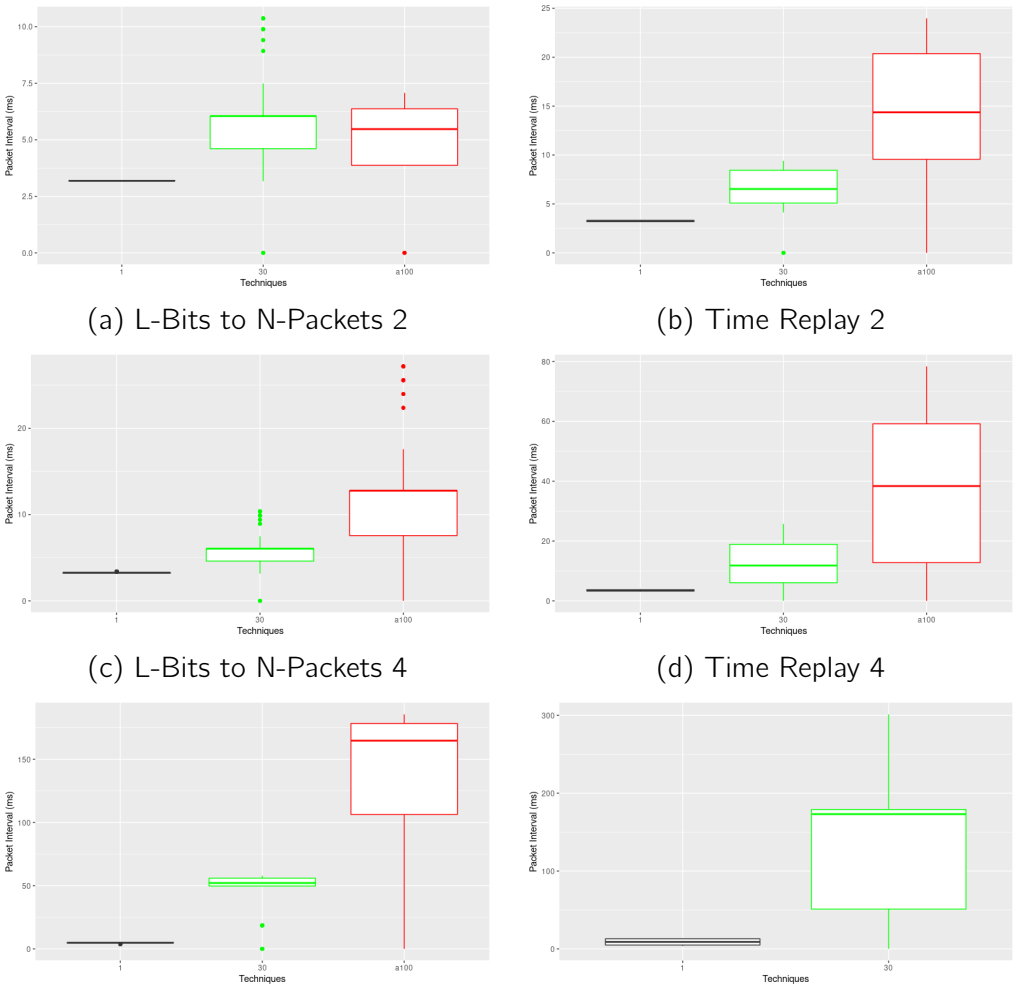
5.4.1 Impact of the CC encoding interval (α)

As referred in the Implementation section of this Thesis, the timing covert channels features a minimum interval needed to encode information. Such value was computed as multiples of one symbol (around 0.032 ms).

All the results presented so far feature an alpha value of 1 symbol, that was found as the lowest possible to successfully encode a piece of covert information. But there was a question as to what would be the impact of variations of this value upon the covert channel. So, using a set of 3 possible different alpha values in symbols ($\alpha = 1$, $\alpha = 30$, $\alpha = 100$), we evaluated its impact upon all techniques.

From figure 5.15, in graphs (a) to (f), conclusions can be observed from the start as to the difference in between the packet interval times, that being, the amount of delay induced into each packet in order to encode hidden information into it.

Relevant to notice in graph (f) of the same figure 5.15 that, it was not possible to fit into a time slot, packets which presented a CC interval of 100 symbols, in the Time Replay technique if it encoded 8 covert bits at a time. Given its 3 interval redundancy, the worst case scenario in that technique (information encoded using the biggest interval possible) would add 76500 symbols of delay (255 possible combinations of 8 bits * 100 alpha delay * 3 intervals to create redundancy) to a single packet, that resulted in approximately 1.3 seconds, and not even a time slot in the superframe 10 scenario could accommodate such a large delay between consecutive packets.



(a) L-Bits to N-Packets 2

(b) Time Replay 2

(c) L-Bits to N-Packets 4

(d) Time Replay 4

(e) L-Bits to N-Packets 8

(f) Time Replay 8

Figure 5.15: CC Techniques packet delay time comparison for different values of α

Naturaly, this value plays a significant role in the resulting delay when encoding the covert information. In figure 5.16 we present the covert channel capacity of all implemented techniques for different alpha settings.

From a first glance, the difference between the results where $\alpha = 1$ to the following is uncanny. The best possible configuration observed was when the alpha was 1 symbol, superframe order 6 on the L-Bits to N-Packets passing 8 covert bits at a time, presenting a covert channel capacity of approximately 5.67 Bytes per second. Using a 30 symbol minimum delay unit (alpha), the L-Bits to N-Packets passing 4 covert bits easily surpassed the competition, having its superframe order 7 configuration achieving a 2.24 Bytes per second covert channel capacity. Given the drastic increase in delay between packets to encode covert information, some techniques (both the Time Replay and the L-Bits to N-Packets passing 8 covert bits per transmission) have to resume transmission on the next superframe, as they cannot fit in a single timeslot. For a 100 symbol alpha, the covert channel capacities are overall very low, with all techniques being severely impacted by this abrupt change in the alpha. In this group of data, the best configuration presented again

the L-Bits to N-Packets technique, but with a higher superframe order of 8. It successfully produced a covert channel capable of a capacity of 1.11 Bytes per second. Clearly, the On/Off technique seems much less sensitive to changes in α. This is because only the minimum interval is used in every case, as a single covert bit gets encoded at a time.

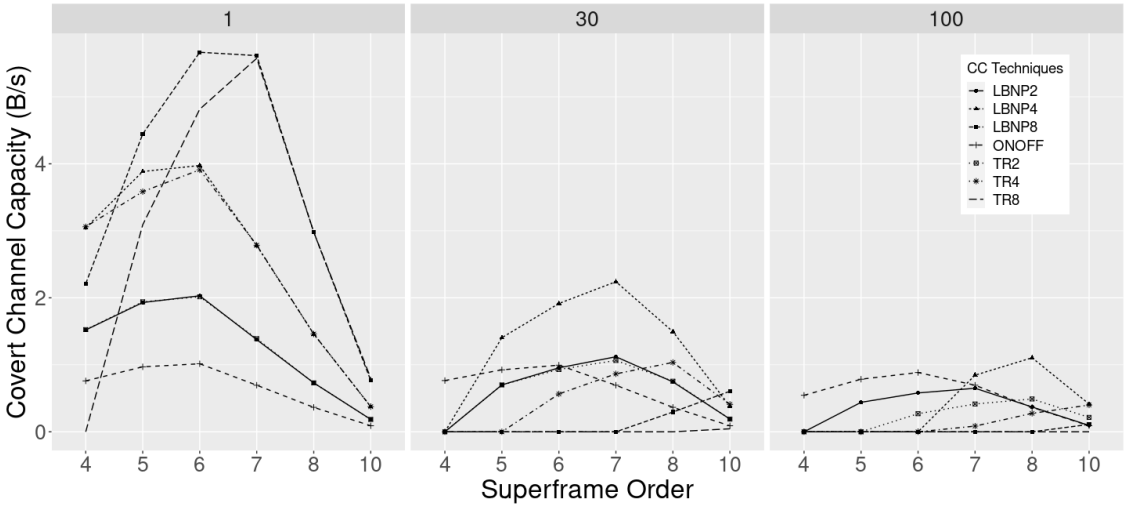


Figure 5.16: CC Techniques comparison on α change

Chapter 6

Conclusions

6.1 Results

Given the obtained results it is clear that such covert channels are very capable of being implemented in the DSME MAC behaviour of the 802.15.4 protocol. Such covert channels can present themselves in various ways in several techniques, some with great effectiveness. Coherent results were obtained and interesting conclusions were retrieved, such as the best configurations for a covert channel and the insight for a proper covert channel mitigation by relying on tuning network settings.

For instance, to fully maximize the transmissions being performed in a network covert channel, its superframe order should be low, with the highest transmitting one ($SO = 6$) with the most amount of traffic running in the simulation ($TGR = 0.01$). In terms of the best channel configuration to extinguish network timing covert channels, higher superframe orders transmit the lowest amount of data, but regardless of the traffic present in the network, they are still consistent in information transmission. But, for scarce traffic in the simulation scenario, the lowest superframe orders ($SO = 4$ OR $SO = 5$) struggle in transmitting information covertly.

The impact of different packet payload lengths was also measured and compared according to the previous metrics. The payload length caused the most impact on networks with high traffic, whereas the smallest packets could pass more information covertly than the chunkier ones due to more frequent transmissions. But, as the traffic in the simulation scenario diminished, the impact of different payload packet lengths was viewed as insignificant and unable to impact the covert channel in a significant fashion.

Introduced with the addition of DSME as a MAC behaviour to the 802.15.4 protocol, the CAP reduction was a new feature that was also target of testing and evaluation on covert channel performance. A network with CAP reduction enables a scenario where the superframes (aside from the first one) are composed of only guaranteed time slots. This results in a much more capable covert channel, that with certain high performance configurations, such as low superframe orders combined with a high traffic support ($TGR = 0.01; 0.05$), can pass almost double the covert information than a network without CAP reduction can. But with higher superframe orders and a deficiency of traffic in the network to support the timing covert channel, the effects of CAP reduction are sparse and barely visible.

In term of the different covert channel techniques, the best performing ones were the Time Replay and with the L-Bits to N-Packets encoding 8 bits of covert information per

packet interval. They presented very similar performance, and in regards to covert channel efficiency, reach 96%. This opens new possibilities to explore further usage of such timing covert channels in this protocol.

However, the minimum CC time interval must be chosen in a way not to jeopardize the CC capacity, by keeping it as low as possible, while maintaining a good decoding quality of the received covert information. The usage of a $\alpha = 1$ as the base delay unit is the best configuration possible of this metric and is able to achieve the highest covert channel capacities.

6.2 Objectives

Regarding the main Thesis' objectives described in the initial chapter, all were successfully accomplished. An ample and thorough survey of both covert channel techniques and their implementations was conducted and presented, focusing relevant covert channels in the desired network communication protocol. The standard was studied and over-viewed, in order to explore the possibility of covert channel implementations, in particular, of the presented techniques. The final simulation platform and its model were obtained through a decision methodology (AHP). From there, the final application design was created, with the addition of modules to an already existent and tested simulation model, presenting different alternatives and approaches. After properly implementing the covert communication module, several simulations were run to explore the performance limits of the TCC techniques.

This work enabled the publication of a scientific article. The paper "Exploring Timing Covert Channel Performance over the IEEE 802.15.4" [Severino, Rodrigues, and Ferreira 2022], was accepted in the 27th International Conference on Emerging Technologies and Factory Automation (ETFA).

6.3 Future Work

The model achieved with the works performed in this Thesis is already deeply tested and functional, but several features can still be added. Mainly, the implementation of different covert channel techniques, not only of the timing variant, but storage ones also. The additional techniques could bring additional covert communication opportunities to increase the flexibility of such tool.

We expect such covert channels can be used in authentication and authorization mechanisms. A preliminary study of the state of the art was already performed, and this possibility will be research next. The contents of this Thesis are fundamental to enable such mechanisms, by leveraging already implemented covert channel techniques.

Bibliography

Alkama, Lynda and Louiza Bouallouche-Medjkoune (Jan. 2021). "IEEE 802.15.4 historical revolution versions: A survey". en. In: *Computing* 103.1, pp. 99–131. issn: 1436-5057. doi: 10.1007/s00607-020-00844-3. url: https://doi.org/10.1007/s00607-020-00844-3 (visited on 02/13/2022).

Alonso, José Antonio and Mª Teresa Lamata (2006). *Consistency in the Analytic Hierarchy Process: A new approach*.

Baronti, Paolo et al. (May 2007). "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards". en. In: *Computer Communications* 30.7, pp. 1655–1695. issn: 01403664. doi: 10.1016/j.comcom.2006.12.020. url: https://linkinghub.elsevier.com/retrieve/pii/S0140366406004749 (visited on 02/13/2022).

Battaglia, Filippo et al. (Jan. 2020). "Novel Extensions to Enhance Scalability and Reliability of the IEEE 802.15.4-DSME Protocol". In: *Electronics* 9. doi: 10.3390/electronics9010126.

Ben Yaala, Sahar, Fabrice Theoleyre, and Ridha Bouallegue (June 2016). "Performance Study of Co-Located IEEE 802.15.4-TSCH Networks: Interference and Coexistence". In: *2016 IEEE Symposium on Computers and Communication (ISCC)*. 2016 IEEE Symposium on Computers and Communication (ISCC). Messina, Italy: IEEE, pp. 513–518. doi: 10.1109/ISCC.2016.7543790. url: https://hal.archives-ouvertes.fr/hal-02566004 (visited on 02/13/2022).

Bensky, Alan (Jan. 2019). "Chapter 12 - Wireless personal area networks". en. In: *Short-range Wireless Communication (Third Edition)*. Ed. by Alan Bensky. Newnes, pp. 317–360. isbn: 978-0-12-815405-2. doi: 10.1016/B978-0-12-815405-2.00012-9. url: https://www.sciencedirect.com/science/article/pii/B9780128154052000129 (visited on 02/13/2022).

Caviglione, Luca (Jan. 2021). "Trends and Challenges in Network Covert Channels Countermeasures". en. In: *Applied Sciences* 11.4. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute, p. 1641. issn: 2076-3417. doi: 10.3390/app11041641. url: https://www.mdpi.com/2076-3417/11/4/1641 (visited on 02/18/2022).

Caviglione, Luca, Alessio Merlo, and Mauro Migliardi (May 2018). "Covert Channels in IoT Deployments Through Data Hiding Techniques". In: *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pp. 559–563. doi: 10.1109/WAINA.2018.00144.

Choudhury, Nikumani et al. (2021). "A Beacon and GTS Scheduling Scheme for IEEE 802.15.4 DSME Networks". In: *IEEE Internet of Things Journal*. Conference Name: IEEE Internet of Things Journal, pp. 1–1. issn: 2327-4662. doi: 10.1109/JIOT.2021.3110866.

Cirani, Simone, Gianluigi Ferrari, and Luca Veltri (June 2013). "Enforcing Security Mechanisms in the IP-Based Internet of Things: An Algorithmic Overview". en. In: *Algorithms* 6.2. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute, pp. 197–226. issn: 1999-4893. doi: 10.3390/a6020197. url: https://www.mdpi.com/1999-4893/6/2/197 (visited on 02/18/2022).

Cortellessa, Vittorio, Antinisca Di Marco, and Paola Inverardi (2011). *Model-Based Software Performance Analysis*. en. Berlin, Heidelberg: Springer Berlin Heidelberg. isbn: 978-3-642-13621-4. doi: 10.1007/978-3-642-13621-4. url: http://link.springer.com/10.1007/978-3-642-13621-4 (visited on 02/16/2022).

Cotroneo, Domenico, Luigi De Simone, and Roberto Natella (Apr. 2021). *Timing Covert Channel Analysis of the VxWorks MILS Embedded Hypervisor under the Common Criteria Security Certification*.

Cramer, Ronald and Ivan Damgård (1997). "Fast and Secure Immunization Against Adaptive Man-in-the-Middle Impersonation". In: *Advances in Cryptology — EUROCRYPT '97*. Ed. by Walter Fumy. Red. by Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen. Vol. 1233. Series Title: Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 75–87. isbn: 978-3-540-62975-7. doi: 10.1007/3-540-69053-0_7. url: http://link.springer.com/10.1007/3-540-69053-0_7 (visited on 06/27/2022).

Cunha, André et al. (Oct. 2007). "Open-ZB: an open-source implementation of the IEEE 802.15.4/ZigBee protocol stack on TinyOS". In: pp. 1–12. doi: 10.1109/MOBHOC.2007.4428602.

Daneels, Glenn et al. (Nov. 2017). "ReSF: Recurrent Low-Latency Scheduling in IEEE 802.15.4e TSCH networks". In: *Ad Hoc Networks* 69. doi: 10.1016/j.adhoc.2017.11.002.

De Guglielmo, Domenico, Simone Brienza, and Giuseppe Anastasi (Aug. 2016). "IEEE 802.15.4e: A survey". en. In: *Computer Communications* 88, pp. 1–24. issn: 0140-3664. doi: 10.1016/j.comcom.2016.05.004. url: https://www.sciencedirect.com/science/article/pii/S0140366416301980 (visited on 02/13/2022).

Elsadig, Muawia A. and Yahia A. Fadlalla (Dec. 2018). "Packet Length Covert Channels Crashed". In: *Journal of Computer Science and Computational Mathematics*, pp. 59–66. issn: 22318879. doi: 10.20967/jcscm.2018.04.001. url: https://www.jcscm.net/cms/?action=showpaper&id=2050637.

Elsts, Atis, Simon Duquennoy, et al. (Nov. 2016). "Microsecond-Accuracy Time Synchronization Using the IEEE 802.15.4 TSCH Protocol". en. In: *2016 IEEE 41st Conference on Local Computer Networks Workshops (LCN Workshops)*. Dubai: IEEE, pp. 156–164. isbn: 978-1-5090-2347-9. doi: 10.1109/LCN.2016.042. url: http://ieeexplore.ieee.org/document/7856151/ (visited on 02/13/2022).

Elsts, Atis, Xenofon Fafoutis, et al. (June 2017). *Adaptive channel selection in IEEE 802.15.4 TSCH networks*. doi: 10.1109/GIOTS.2017.8016246.

Fafoutis, Xenofon et al. (Feb. 2018). "Adaptive static scheduling in IEEE 802.15.4 TSCH networks". en. In: *2018 IEEE 4th World Forum on Internet of Things (WF-IoT)*. Singapore: IEEE, pp. 263–268. isbn: 978-1-4673-9944-9. doi: 10.1109/WF-IoT.2018.8355114. url: https://ieeexplore.ieee.org/document/8355114/ (visited on 02/13/2022).

Farahani, Shahin (Jan. 2008). "Chapter 2 - ZigBee/IEEE 802.15.4 Networking Examples". en. In: *ZigBee Wireless Networks and Transceivers*. Ed. by Shahin Farahani. Burlington: Newnes, pp. 25–32. isbn: 978-0-7506-8393-7. doi: 10.1016/B978-0-7506-8393-7.00002-9. url: https://www.sciencedirect.com/science/article/pii/B9780750683937000029 (visited on 02/13/2022).

Frustaci, Mario et al. (Aug. 2018). "Evaluating Critical Security Issues of the IoT World: Present and Future Challenges". en. In: *IEEE Internet of Things Journal* 5.4, pp. 2483–2495. issn: 2327-4662, 2372-2541. doi: 10.1109/JIOT.2017.2767291. url: https://ieeexplore.ieee.org/document/8086136/ (visited on 02/18/2022).

Gamundani, Attlee, Amelia Phillips, and Hippolyte Muyingi (Nov. 2018). "An Overview of Potential Authentication Threats and Attacks on Internet of Things(IoT): A Focus on Smart Home Applications". In: doi: 10.1109/Cybermatics_2018.2018.00043.

Han, Jiaxuan et al. (2020). "Covert timing channel detection method based on time interval and payload length analysis". In: *Computers and Security* 97, p. 101952. issn: 01674048. doi: 10.1016/j.cose.2020.101952. url: https://doi.org/10.1016/j.cose.2020.101952.

Hou, Ningning and Yuanqing Zheng (Oct. 2020). "CloakLoRa: A Covert Channel over LoRa PHY". en. In: *2020 IEEE 28th International Conference on Network Protocols (ICNP)*. Madrid, Spain: IEEE, pp. 1–11. isbn: 978-1-72816-992-7. doi: 10.1109/ICNP49622.2020.9259364. url: https://ieeexplore.ieee.org/document/9259364/ (visited on 02/13/2022).

Hu, Jie, Chuang Lin, and Xiangyang Li (Aug. 2016). "Relationship Privacy Leakage in Network Traffics". In: *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–9. doi: 10.1109/ICCCN.2016.7568566.

Hwang, Kwang-il and Sung-wook Nam (May 2014). "Analysis and Enhancement of IEEE 802.15.4e DSME Beacon Scheduling Model". en. In: *Journal of Applied Mathematics* 2014. Publisher: Hindawi, e934610. issn: 1110-757X. doi: 10.1155/2014/934610. url: https://www.hindawi.com/journals/jam/2014/934610/ (visited on 02/13/2022).

IEEE (Apr. 2012). "IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer". In: *IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011)*. Conference Name: IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011), pp. 1–225. doi: 10.1109/IEEESTD.2012.6185525.

– (July 2020). "IEEE Standard for Low-Rate Wireless Networks". In: *IEEE Std 802.15.4-2020 (Revision of IEEE Std 802.15.4-2015)*. Conference Name: IEEE Std 802.15.4-2020 (Revision of IEEE Std 802.15.4-2015), pp. 1–800. doi: 10.1109/IEEESTD.2020.9144691.

Johnson, Daryl et al. (2010). "Covert channels in the HTTP network protocol: Channel characterization and detecting man-in-the-middle attacks". In: p. 11.

Johnson, Matthew, Peter Lutz, and Daryl Johnson (Dec. 2016). "Covert Channel Using Man-in-the-Middle over HTTPS". In: *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*. 2016 International Conference on Computational Science and Computational Intelligence (CSCI). Las Vegas, NV, USA: IEEE, pp. 917–922. isbn: 978-1-5090-5510-4. doi: 10.1109/CSCI.2016.0177. url: http://ieeexplore.ieee.org/document/7881470/ (visited on 06/27/2022).

Kauer, Florian, Maximilian Köstler, and Volker Turau (June 2018). "Reliable Wireless Multi-Hop Networks with Decentralized Slot Management: An Analysis of IEEE 802.15.4 DSME". In: *arXiv:1806.10521 [cs]*. arXiv: 1806.10521. url: http://arxiv.org/abs/1806.10521 (visited on 02/13/2022).

Khan, Atta ur Rehman, Sardar M. Bilal, and Mazliza Othman (2012). "Comparison of Different Network Simulators". en. In: p. 6.

Kim, Doohwan, Jae-Young Choi, and Jang-Eui Hong (Jan. 2017). "Evaluating energy efficiency of Internet of Things software architecture based on reusable software components". In: *International Journal of Distributed Sensor Networks* 13.1. Publisher: SAGE Publications, p. 1550147716682738. issn: 1550-1329. doi: 10.1177/1550147716682738. url: https://doi.org/10.1177/1550147716682738 (visited on 02/18/2022).

Kiyavash, Negar et al. (Mar. 2013). "A Timing Channel Spyware for the CSMA/CA Protocol". In: *IEEE Transactions on Information Forensics and Security* 8 (3), pp. 477–487.

issn: 1556-6013. doi: 10.1109/TIFS.2013.2238930. url: http://ieeexplore.ieee.org/document/6410028/.

Kolias, Constantinos et al. (Jan. 2017). "DDoS in the IoT: Mirai and other botnets". In: *Computer* 50, pp. 80–84. doi: 10.1109/MC.2017.201.

Köstler, Maximilian et al. (Sept. 2016). "Towards an Open Source Implementation of the IEEE 802.15.4 DSME Link Layer". In: *Proceedings of the 15. GI/ITG KuVS Fachgespräch Sensornetze*. Ed. by Juergen Scholz and Alexander von Bodisco. Augsburg, Germany: University of Applied Sciences Augsburg, Dept. of Computer Science, p. 4.

kourzanov (Feb. 2022). *EIT-ICT-RICH/ns-3-dev-TSCH*. original-date: 2015-01-08T10:11:53Z. url: https://github.com/EIT-ICT-RICH/ns-3-dev-TSCH (visited on 02/20/2022).

Krebs, Brian (2017). *Mirai IoT Botnet Co-Authors Plead Guilty – Krebs on Security*. en-US. url: https://krebsonsecurity.com/2017/12/mirai-iot-botnet-co-authors-plead-guilty/ (visited on 02/18/2022).

Krueger, Leo and Shudrenko Yevhenii (Feb. 2022). *TSCH*. original-date: 2019-11-22T16:08:26Z. url: https://github.com/ComNetsHH/omnetpp-tsch (visited on 02/13/2022).

Kurunathan, John Harrison (2021). "Improving QoS for IEEE 802.15.4e DSME Networks". en. In: p. 158.

Lampson, Butler W. (Oct. 1973). "A note on the confinement problem". en. In: *Communications of the ACM* 16.10, pp. 613–615. issn: 0001-0782, 1557-7317. doi: 10.1145/362375.362389. url: https://dl.acm.org/doi/10.1145/362375.362389 (visited on 02/13/2022).

Larson, Selena (July 2017). *A smart fish tank left a casino vulnerable to hackers*. url: https://money.cnn.com/2017/07/19/technology/fish-tank-hack-darktrace/index.html (visited on 02/18/2022).

Lin, Jie et al. (Oct. 2017). "A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications". en. In: *IEEE Internet of Things Journal* 4.5, pp. 1125–1142. issn: 2327-4662, 2372-2541. doi: 10.1109/JIOT.2017.2683200. url: https://ieeexplore.ieee.org/document/7879243/ (visited on 02/18/2022).

Liu, Kun, YunRui Bi, and Di Liu (Jan. 2020). "Internet of Things based acquisition system of industrial intelligent bar code for smart city applications". en. In: *Computer Communications* 150, pp. 325–333. issn: 0140-3664. doi: 10.1016/j.comcom.2019.11.044. url: https://www.sciencedirect.com/science/article/pii/S0140366419312794 (visited on 02/13/2022).

Liu, Zhihong et al. (Apr. 2020). "Covert Wireless Communication in IoT Network: From AWGN Channel to THz Band". In: *IEEE Internet of Things Journal* 7.4. Conference Name: IEEE Internet of Things Journal, pp. 3378–3388. issn: 2327-4662. doi: 10.1109/JIOT.2020.2968153.

Lu, Shoupu et al. (Oct. 2019). "A Novel Timing-based Network Covert Channel Detection Method". In: *Journal of Physics: Conference Series* 1325 (1), p. 012050. issn: 1742-6588. doi: 10.1088/1742-6596/1325/1/012050. url: https://iopscience.iop.org/article/10.1088/1742-6596/1325/1/012050.

Martins, David and Hervé Guyennet (2010). "Steganography in MAC layers of 802.15.4 protocol for securing wireless sensor networks". In: *Proceedings - 2010 2nd International Conference on Multimedia Information Networking and Security, MINES 2010*, pp. 824–828. doi: 10.1109/MINES.2010.175.

Masood, Tariq and Paul Sonntag (Oct. 2020). "Industry 4.0: Adoption challenges and benefits for SMEs". en. In: *Computers in Industry* 121, p. 103261. issn: 0166-3615. doi: 10.1016/j.compind.2020.103261. url: https://www.sciencedirect.com/science/article/pii/S0166361520304954 (visited on 02/15/2022).

Meng, Mei et al. (Feb. 2018). "Scheduling for Data Transmission in Multi-Hop IEEE 802.15.4e TSCH Networks". en. In: *Mobile Networks and Applications* 23.1, pp. 119–125. issn: 1572-8153. doi: 10.1007/s11036-017-0887-9. url: https://doi.org/10.1007/s11036-017-0887-9 (visited on 02/13/2022).

Meyer, Florian, Phil Malessa, et al. (Sept. 2021). "Are Group Acknowledgements Worth Anything in IEEE 802.15.4 DSME: A Comparative Analysis". In: *arXiv:2109.06267 [cs]*. arXiv: 2109.06267. url: http://arxiv.org/abs/2109.06267 (visited on 02/13/2022).

Meyer, Florian, Ivonne Mantilla-González, and Volker Turau (2021). "Sending multiple packets per guaranteed time slot in IEEE 802.15.4 DSME: Analysis and evaluation". en. In: *Internet Technology Letters* 4.4. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/itl2.167, e167. issn: 2476-1508. doi: 10.1002/itl2.167. url: https://onlinelibrary.wiley.com/doi/abs/10.1002/itl2.167 (visited on 02/13/2022).

Muñuzuri, Jesús et al. (Jan. 2020). "Using IoT data and applications to improve port-based intermodal supply chains". en. In: *Computers & Industrial Engineering* 139, p. 105668. issn: 0360-8352. doi: 10.1016/j.cie.2019.01.042. url: https://www.sciencedirect.com/science/article/pii/S0360835219300488 (visited on 02/13/2022).

Nain, Ajay Kumar and P. Rajalakshmi (2017). "A reliable covert channel over IEEE 802.15.4 using steganography". In: *2016 IEEE 3rd World Forum on Internet of Things, WF-IoT 2016*, pp. 711–716. doi: 10.1109/WF-IoT.2016.7845486.

NS3 simulations (Sept. 2017). *Wireless Sensor Network Simulation in NS3 | Wireless Sensor Network Simulation in NS3 projects*. url: https://www.youtube.com/watch?v=_S9kGO9j9BU (visited on 02/16/2022).

nsnam (2022). *ns-3*. en. url: https://www.nsnam.org/ (visited on 02/13/2022).

OMNeT++ (2022). *OMNeT++ Discrete Event Simulator*. url: https://omnetpp.org/ (visited on 02/15/2022).

Osterlind, Fredrik et al. (Nov. 2006). "Cross-Level Sensor Network Simulation with COOJA". en. In: *Proceedings. 2006 31st IEEE Conference on Local Computer Networks*. ISSN: 0742-1303. Embassy Suites Hotel, Tampa, FL, USA: IEEE, pp. 641–648. isbn: 978-1-4244-0418-6. doi: 10.1109/LCN.2006.322172. url: http://ieeexplore.ieee.org/document/4116633/ (visited on 02/13/2022).

Pfitzmann, Birgit (May 1996). "Information Hiding Terminology - Results of an Informal Plenary Meeting and Additional Proposals". In: *Proceedings of the First International Workshop on Information Hiding*. Berlin, Heidelberg: Springer-Verlag, pp. 347–350. isbn: 978-3-540-61996-3. (Visited on 02/18/2022).

Queiroz, Diego V. et al. (Nov. 2017). "Survey and systematic mapping of industrial Wireless Sensor Networks". en. In: *Journal of Network and Computer Applications* 97, pp. 96–125. issn: 1084-8045. doi: 10.1016/j.jnca.2017.08.019. url: https://www.sciencedirect.com/science/article/pii/S1084804517302771 (visited on 02/17/2022).

Rawlinson, Kristi (2014). *HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack*. url: https://www.hp.com/us-en/hp-news/press-release.html?id=1744676 (visited on 02/18/2022).

Riahi Sfar, Arbia et al. (Apr. 2018). "A roadmap for security challenges in the Internet of Things". en. In: *Digital Communications and Networks* 4.2, pp. 118–137. issn: 2352-8648. doi: 10.1016/j.dcan.2017.04.003. url: https://www.sciencedirect.com/science/article/pii/S2352864817300214 (visited on 02/18/2022).

Rowland, Craig H. (May 1997). "Covert channels in the TCP/IP protocol suite". en. In: *First Monday*. issn: 1396-0466. doi: 10.5210/fm.v2i5.528. url: https://journals.uic.edu/ojs/index.php/fm/article/view/528 (visited on 02/18/2022).

Saaty, Thomas L. (1994). "How to Make a Decision: The Analytic Hierarchy Process". en. In: *Interfaces* 24.6, pp. 19–43. url: http://www.jstor.org/stable/25061950.

Sahoo, Prasan, Sudhir Pattanaik, and Shih-Lin Wu (Dec. 2019). "A Novel Synchronous MAC Protocol for Wireless Sensor Networks with Performance Analysis". In: *Sensors* 19, p. 5394. doi: 10.3390/s19245394.

Severino, Ricardo, Joao Rodrigues, and Luis Lino Ferreira (2022). "Exploring Timing Covert Channel Performance over the IEEE 802.15.4". en. In: p. 8.

Shah, Gaurav, Andres Molina, and Matt Blaze (2006). "Keyboards and Covert Channels". In: p. 17.

Shih, Chao-Fang, Ariton E. Xhafa, and Jianwei Zhou (June 2015). "Practical frequency hopping sequence design for interference avoidance in 802.15.4e TSCH networks". In: *2015 IEEE International Conference on Communications (ICC)*. ISSN: 1938-1883, pp. 6494–6499. doi: 10.1109/ICC.2015.7249359.

Simmons, Gustavus J. (1984). "The Prisoners' Problem and the Subliminal Channel". en. In: *Advances in Cryptology: Proceedings of Crypto 83*. Ed. by David Chaum. Boston, MA: Springer US, pp. 51–67. isbn: 978-1-4684-4730-9. doi: 10.1007/978-1-4684-4730-9_5. url: https://doi.org/10.1007/978-1-4684-4730-9_5 (visited on 02/15/2022).

Simon, Scott (Oct. 2016). "'Internet Of Things' Hacking Attack Led To Widespread Outage Of Popular Websites". en. In: *NPR*. url: https://www.npr.org/2016/10/22/498954197/internet-outage-update-internet-of-things-hacking-attack-led-to-outage-of-popula (visited on 02/18/2022).

Sinha, Akash, Gulshan Shrivastava, and Prabhat Kumar (Sept. 2019). "Architecting user-centric internet of things for smart agriculture". en. In: *Sustainable Computing: Informatics and Systems* 23, pp. 88–102. issn: 2210-5379. doi: 10.1016/j.suscom.2019.07.001. url: https://www.sciencedirect.com/science/article/pii/S2210537919300137 (visited on 02/13/2022).

Sitanayah, Lanny, Cormac Sreenan, and Szymon Fedor (Nov. 2013). "A Cooja-based tool for maintaining sensor network coverage requirements in a building". In: doi: 10.1145/2517351.2517390.

Stanislav, Mark and Tod Beardsley (2015). "HACKING IoT: A Case Study on Baby Monitor Exposures and Vulnerabilities". en. In: p. 17.

Stanislowski, David et al. (Feb. 2014). "Adaptive Synchronization in IEEE802.15.4e Networks". In: *IEEE Transactions on Industrial Informatics* 10.1. Conference Name: IEEE Transactions on Industrial Informatics, pp. 795–802. issn: 1941-0050. doi: 10.1109/TII.2013.2255062.

Statista (2022). *Number of IoT devices 2015-2025*. en. url: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ (visited on 02/15/2022).

Sullivan, Kevin J et al. (2001). "The Structure and Value of Modularity in Software Design". en. In: p. 10.

Tahmasbi, Fatemeh, Neda Moghim, and Mojtaba Mahdavi (Nov. 2016). "Adaptive ternary timing covert channel in IEEE 802.11". In: *Security and Communication Networks* 9 (16), pp. 3388–3400. issn: 19390114. doi: 10.1002/sec.1545. url: https://onlinelibrary.wiley.com/doi/10.1002/sec.1545.

Tan, Yu-an et al. (Dec. 2018). "Covert Timing Channels for IoT over Mobile Networks". In: *IEEE Wireless Communications* 25 (6), pp. 38–44. issn: 1536-1284. doi: 10.1109/MWC.2017.1800062. url: https://ieeexplore.ieee.org/document/8600755/.

Tian, Jing et al. (Aug. 2020). "A Survey of Key Technologies for Constructing Network Covert Channel". en. In: *Security and Communication Networks* 2020. Publisher: Hindawi,

e8892896. issn: 1939-0114. doi: 10.1155/2020/8892896. url: https://www.hindawi.com/journals/scn/2020/8892896/ (visited on 02/13/2022).

Tuptuk, Nilufer and Stephen Hailes (Mar. 2015). "Covert channel attacks in pervasive computing". In: *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pp. 236–242. doi: 10.1109/PERCOM.2015.7146534.

Al-Turjman, Fadi, Muhammad Hassan Nawaz, and Umit Deniz Ulusar (Jan. 2020). "Intelligence in the Internet of Medical Things era: A systematic review of current and future trends". en. In: *Computer Communications* 150, pp. 644–660. issn: 0140-3664. doi: 10.1016/j.comcom.2019.12.030. url: https://www.sciencedirect.com/science/article/pii/S0140366419313337 (visited on 02/13/2022).

U.S. Department of Defense (2016). *DoD Policy Recommendations for The Internet of Things (IoT)*.

Valero, M., Anu Bourgeois, and Raheem Beyah (June 2010). "DEEP: A deployable energy efficient 802.15.4 MAC protocol for sensor networks". In: pp. 1–6. doi: 10.1109/ICC.2010.5501955.

Varga, András (2003). *OMNeT++ Manual*. 2.3. url: http://src.gnu-darwin.org/ports/science/omnetpp/work/omnetpp-2.3p1/doc/usman.pdf (visited on 02/13/2022).

Vogli, Elvis et al. (Feb. 2018). "Fast network joining algorithms in industrial IEEE 802.15.4 deployments". en. In: *Ad Hoc Networks* 69, pp. 65–75. issn: 15708705. doi: 10.1016/j.adhoc.2017.10.013. url: https://linkinghub.elsevier.com/retrieve/pii/S1570870517301853 (visited on 02/13/2022).

Wang, Jia-huai (July 2020). "T cell receptors, mechanosensors, catch bonds and immunotherapy". en. In: *Progress in Biophysics and Molecular Biology* 153, pp. 23–27. issn: 0079-6107. doi: 10.1016/j.pbiomolbio.2020.01.001. url: https://www.sciencedirect.com/science/article/pii/S0079610720300080 (visited on 02/13/2022).

Weingartner, E., H. vom Lehn, and K. Wehrle (June 2009). "A Performance Comparison of Recent Network Simulators". en. In: *2009 IEEE International Conference on Communications*. Dresden, Germany: IEEE, pp. 1–5. doi: 10.1109/ICC.2009.5198657. url: http://ieeexplore.ieee.org/document/5198657/ (visited on 02/13/2022).

Wendzel, Steffen et al. (Apr. 2015). "A Pattern-based Survey and Categorization of Network Covert Channel Techniques". In: *ACM Computing Surveys* 47.3. arXiv: 1406.2901, pp. 1–26. issn: 0360-0300, 1557-7341. doi: 10.1145/2684195. url: http://arxiv.org/abs/1406.2901 (visited on 02/18/2022).

Wolf, Manfred (1989). "Covert channels in LAN protocols". en. In: *Local Area Network Security*. Ed. by Thomas A. Berson and Thomas Beth. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 89–101. isbn: 978-3-540-46802-8. doi: 10.1007/3-540-51754-5_33.

Xie, Dong, Jiang Li, and Huisheng Gao (Mar. 2020). "Comparison and Analysis of Simulation methods for TSN Performance". en. In: *IOP Conference Series: Materials Science and Engineering* 768.5, p. 052061. issn: 1757-8981, 1757-899X. doi: 10.1088/1757-899X/768/5/052061. url: https://iopscience.iop.org/article/10.1088/1757-899X/768/5/052061 (visited on 02/13/2022).

Xie, Haijiang and Jizhong Zhao (2015). "A lightweight identity authentication method by exploiting network covert channel". In: p. 10.

Ying, Xuhang et al. (Mar. 2019). "TACAN: Transmitter Authentication through Covert Channels in Controller Area Networks". In: *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, pp. 23–34. url: http://arxiv.org/abs/1903.05231.

Zander, Sebastian and Grenville Armitage (Jan. 2008). "CCHEF–Covert Channels Evaluation Framework Design and Implementation". In.

Zarrad, Anis and Izzat Alsmadi (Oct. 2017). "Evaluating network test scenarios for network simulators systems". In: *International Journal of Distributed Sensor Networks* 13.10. Publisher: SAGE Publications, p. 1550147717738216. issn: 1550-1329. doi: 10.1177/1550147717738216. url: https://doi.org/10.1177/1550147717738216 (visited on 02/13/2022).

Zhang, Xiaosong et al. (Jan. 2019). "A packet-reordering covert channel over VoLTE voice and video traffics". In: *Journal of Network and Computer Applications* 126, pp. 29–38. issn: 10848045. doi: 10.1016/j.jnca.2018.11.001. url: https://linkinghub.elsevier.com/retrieve/pii/S1084804518303527.