Faculty of Sciences and Technology

Department of Informatics Engineering

Energy Efficient Mechanisms for 5G Networks

Luís Filipe Gouveia da Silva

Dissertation in the context of the Master in Informatics Engineering, Specialization in Software Engineering advised by Prof. Jorge Granjal and Prof. Marília Curado and presented to the Faculty of Sciences and Technology / Department of Informatics Engineering..

September 2019

1 2 9 0

UNIVERSIDADE Đ COIMBRA

This page is intentionally left blank.

Abstract

Fifth Generation (5G) will allow new functional and security requirements. Security is one of the most critical requirements regarding end-to-end secure communications. However, it has a price in terms of energy cost. Confidentiality, integrity, authentication and non-repudiation composes this security, especially at Wireless Sensor Networks (WSN), and these are the security requirements more essential and required at the Internet of Things (IoT). Some works handle the energy-security trade-off at a validation way. However, the more interesting are the ones that tackle this trade-off in an adaptation manner. A lack of works regarding this adaptation of energy cost and security trade-off is a reality, in particular at WSN using low-power and short-range technologies. This lack occurs in Constrained Application Protocol (CoAP) environments using standards such as Datagram Transport Layer Security (DTLS), IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) or the IEEE Standard for Low-Rate Wireless Networks (IEEE 802.15.4).

This thesis proposes a controller to handle energy and security trade-off, besides other variables such as applications requirements, attacks occurrences or the battery level. The primary role of this controller is to perform security adaptation at run-time considering the previous aspects. It is the essential component of the presented architecture regarding its purpose. The main objective is the selection of the optimal energy-aware security profile.

A set of security algorithms was created, gathering algorithms such as Advanced Encryption Standard (AES), Triple Data Encryption Standard (3DES), Secure Hash Algorithm 1 (SHA1) and Secure Hash Algorithm 2 (SHA2). Then, an energy analysis was made of each algorithm with various configurations. This analysis was initial work to stipulates static security profiles to be used with applications and to perform security adaptation with the controller.

The controller creation was conducted with a given proposal followed by a formalisation. Then, pseudo-code was presented and explained. Finally, an analytical validation was performed. The purpose of this validation was the controller validation role using simulated variables. Then a presentation regarding future works with this component was presented given the enormous potential of the given controller.

Keywords

5G, 6LoWPAN, CoAP, DTLS, IEEE 802.15.4, IoT, WSN

This page is intentionally left blank.

Resumo

O 5G permitirá o estabelecimento de novos requisitos funcionais e de segurança. Estes últimos são dos requisitos mais importantes relativamente à segurança ponto a ponto nas comunicações. No entanto, esta segurança tem um preço em termos de custo energético. A confidencialidade, a integridade, a autenticação e o não repúdio compõem esta segurança, especialmente nas Redes de Sensores Sem Fios (RSSF). Estes são os requisitos de segurança mais importantes e exigidos na Internet das Coisas (IoT). Alguns trabalhos lidam com o compromisso entre energia e segurança de uma maneira de validação. No entanto, os trabalhos mais interessantes são aqueles que lidam com este compromisso de uma maneira de adaptação. A falta de trabalhos relacionados com esta adaptação do custo energético e do compromisso da segurança é uma realidade, particularmente nas RSSF, utilizando tecnologias de baixo consumo e de curto alcance. Esta lacuna ocorre em ambientes CoAP usando protocolos tais como o DTLS, 6LoWPAN ou o IEEE 802.15.4.

Esta tese propõe um controlador para lidar com a energia e o compromisso da segurança, além de também lidar com outras variáveis, tais como os requisitos das aplicações, a ocorrência de ataques ou o nível de bateria. A função principal deste controlador é efetuar a adaptação da segurança em tempo real, considerando os aspetos anteriores. Este é o principal componente da arquitetura apresentada tendo em conta a sua função. O objetivo principal é a seleção do melhor perfil de segurança.

Um conjunto de algoritmos de segurança foi criado, reunindo algoritmos tais como o AES, o 3DES, o SHA1 e o SHA2. Depois, uma análise de consumo energético foi efetuada para cada um destes algoritmos e respetivas configurações. Esta análise foi trabalho prévio, de modo a serem estipulados perfis estáticos de segurança a serem utilizados com aplicações e na adaptação da segurança com o controlador.

A criação do controlador foi guiada com a disponibilização de uma proposta seguida de uma formalização. Depois foi apresentado e explicado o pseudocódigo do controlador. Finalmente uma validação analítica foi realizada. Esta, teve como objetivo a validação do comportamento do controlador utilizando a simulação de variáveis. Depois foi apresentada uma exposição relacionada com o trabalho futuro e o controlador, tendo em conta o grande potencial deste componente.

Palavras-Chave

5G, 6LoWPAN, CoAP, DTLS, IEEE 802.15.4, IoT, WSN

This page is intentionally left blank.

Acknowledgements

I would like to thank Professors Jorge Granjal and Marília Curado for their guidance and valuable suggestions on this journey. To all my family, especially my parents and brothers, who always supported me, even in difficult times. To Alexandra, thank you for your support, encouragement and for letting me be part of your life. To my friends and colleagues, thank you for the help and knowledge exchange.

The work presented in this dissertation was partially carried out in the scope of the MobiWise project: From mobile sensing to mobility advising (P2020 SAICTPAC/0011/2015), co-financed by COMPETE 2020, Portugal 2020 - Operational Program for Competitiveness and Internationalization (POCI), European Union's ERDF (European Regional Develop- ment Fund), and the Portuguese Foundation for Science and Technology (FCT).

Cofinanciado por:

This page is intentionally left blank.

Contents

Acronyms

3DES Triple Data Encryption Standard. iii, v, 35, 40, 43, 45, 52

5G Fifth Generation. iii, 1, 25, 26, 33

6LoWPAN IPv6 over Low-Power Wireless Personal Area Networks. iii, v, 1, 8, 9, 11, 13, 14, 31, 32, 34, 38

6TiSCH IPv6 over the time slotted channel hopping mode of IEEE 802.15.4e. 8

A Authentication. 50, 51

ABP Activation by Personalization. 14

ACL Access Control Lists. 10

AES Advanced Encryption Standard. iii, v, 9, 10, 13–15, 35, 40, 43, 45, 52

AH Authentication Header. 10

AMQP Advanced Message Queuing Protocol. 13

BLE Bluetooth Low Energy. 13, 25

C Confidentiality. 50, 51

CA Certificate Authorities. 25

CCM Cipher Block Chaining-Message Authentication Code. 13, 15

CCMP Counter Mode CBC-MAC Protocol. 12

CoAP Constrained Application Protocol. iii, v, 1, 2, 9, 15, 19–23, 28, 29, 31, 32, 34, 38, 49, 58, 68, 72, 73, 89, 90

CTS Clear to Send. 19

D2D Device-to-Device. 1

DES Data Encryption Standard. 40

DODAG Destination Oriented Direct Acyclic Graph. 11

DoS Denial of Service. 9, 18, 21, 22, 31

DTLS Datagram Transport Layer Security. iii, v, 1, 2, 9, 14, 21, 22, 24, 25, 31, 32, 34, 36, 37

DVR Distance Vector Routing. 14

ECB Electronic Codebook. 43

eeDTLS Energy-Efficient DTLS. 24

ESP Encapsulating Security Payload. 10

EU Europe Union. 1

GCC GNU Compiler Collection. 45

GSMA Group Special Mobile Association. 1

HCI Host Controller Interface. 13

HTTP Hypertext Transfer Protocol. 19

I Integrity. 50, 51

ICMP Internet Control Message Protocol. 11

IDS Intrusion Detection System. 25, 26, 34, 36

IEEE Institute of Electrical and Electronics Engineers. 5, 8

IEEE 802.15.4 IEEE Standard for Low-Rate Wireless Networks. iii, v, 1, 8, 9, 13–15, 25, 31, 38

IEEE 802.15.6 IEEE Standard for Local and metropolitan area networks - Part 15.6: Wireless Body Area Networks. 10

IEEE C37.1-2007 IEEE Standard for SCADA and Automation Systems. 69

IETF Internet Engineering Task Force. 5, 8–12

IoT Internet of Things. iii, xvii, 2, 3, 5–10, 12, 13, 15, 24, 25, 29, 31, 32, 89, 91

IPSec Internet Protocol Security. 10

ISA International Society of Automation. 13

ISM Industrial Scientific Medical. 10, 13, 15

ISO International Organization for Standardization. 10

LoRaWAN Long Range Wide-Area Network. 13, 14

LPWAN Low-Power Wide-Area Networks. 14

M2M Machine-to-Machine. 1, 9, 14, 15, 19, 31

MAC Medium Access Control. 9, 10, 13, 14

MCC Mobile Cloud Computing. 26

MIC Message Integrity Check. 13, 14

MIMO Multiple-Input and Multiple-Output. 25

mmWave Millimetre Wave. 1, 25

MQTT Message Queuing Telemetry Transport. 10–12

MQTT-SN MQTT For Sensor Networks. 11

NFC Near Field Communications. 8, 14, 23

NGMN Next Generation Mobile Networks. 1

NR Non-repudiation. 50, 51

OASIS Organization for the Advancement of Structured Information Standards. 10, 13

OS Operating System. 38, 39, 90, 91

OSCORE Object Security for Constrained RESTful Environments. 22

OTAA Over The Air Activation. 14

QoS Quality of Service. 6, 10, 11

RADIUS Remote Authentication Dial-In User Service. 12

RF Radio Frequency. 13

ROLL Routing Over Low-power and Lossy Networks. 11

RPL IPv6 Routing Protocol for Low-Power and Lossy Networks. 11, 34, 38

RTD Signature Record Type Definition. 14

RTS Request to Send. 19

RTT Round-Trip-Time. 23

SA Security Association. 36

SADB Security Association Database. 36

SASL Simple Authentication and Security Layer. 13

SCADA Supervisory Control and Data Acquisition. 69

SHA Secure Hash Algorithm. 45

SHA1 Secure Hash Algorithm 1. iii, v, 40, 43, 52

SHA2 Secure Hash Algorithm 2. iii, v, 35, 40, 43, 52

SIG Bluetooth Special Interest Group. 13

SP Security Profile. 53–55, 58, 60–62, 72, 81

SSL Secure Sockets Layer. 12

TCP Transmission Control Protocol. 11–13

TDMA Time Division Multiple Access. 14

TKIP Temporal Key Integrity Protocol. 12

TLS Transport Layer Security. 9, 11–13, 31

TSCH Time-Slotted Channel Hopping. 8

UDP User Datagram Protocol. 9, 14, 21, 22, 31

URI Uniform Resource Identifier. 9, 20, 21, 31

WEP Wireless Equivalent Privacy. 12

WPA Wifi Protected Access. 12

WSN Wireless Sensor Networks. iii, 1, 2, 5, 7, 15, 17, 18, 25, 26, 28, 29, 34, 35, 38, 40, 68, 89, 90

XML Extensible Markup Language. 12

XMPP Extensible Messaging and Presence Protocol. 12

List of Figures

List of Tables

This page is intentionally left blank.

Chapter 1

Introduction

In this chapter, the context and motivation are introduced. Also, the structure of the thesis is presented, besides the objectives and the challenges.

1.1 Context and motivation

Many entities are performing research in the Fifth Generation (5G) network. Some belong to the Industry and others to the Academic field. Institutions like the Europe Union (EU) [Commission, 2018], the Next Generation Mobile Networks (NGMN) [Networks, 2018] or the Group Special Mobile Association (GSMA) [Association, 2018] are encouraging 5G network development. This development is being performed across partnerships between the Industry and the Academic field.

5G will ensure several functional requirements and quality attributes [Albreem, 2015]. Faster speeds, lower latency, performance, scalability, and reliability, security or energy efficiency represent an evolution of the user experience. In consideration of these qualities attributes, it will be possible to enable new Device-to-Device (D2D) or Machine-to-Machine (M2M) applications that will impact Industry and Consumers [Elena, 2014].

With new technologies, new problems and threats are emerging. These topics are related to the performance and security of the network and devices. Signal attenuation and power consumption, bearing in mind Millimetre Wave (mmWave) or the security of data hosted in virtualised environments, depending on the isolation degree of these environments, are topics encountered in 5G. The changing ecosystem is another driver for 5G security. It is because the 5G network will be composed of many and different stakeholders providing specialised services. These points justify why the hop by hop security, present today in traditional networks, may not be efficient in 5G environments [Dr. Muhammad, 2017].

This master thesis addresses security and energy consumption in 5G networks, particularly at Wireless Sensor Networks (WSN) regarding the topics presented. These networks are composed of resource-constrained devices mainly related to energy consumption, processing power or storage capacity. That is why it will be addressed low-power and short-range technologies, such as IEEE Standard for Low-Rate Wireless Networks (IEEE 802.15.4), IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN), Datagram Transport Layer Security (DTLS) or Constrained Application Protocol (CoAP). Nowadays, security-energy trade-off is an exciting challenge, and the secret is to achieve a middle ground. Some works were presented, but the most attractive are the ones that approach the energy-security trade-off, not in a validation way, but in an adaptive way.

The works that approach the energy-security trade-off, in an adaptive way, are scarce. However, when they are performed, they do not consider just technologies with low-power and short-range. Another aspect is regarding the way that they perform security. Usually, this security is not made at upper layers, but it is performed at lower ones. As an example, CoAP is typically used with the DTLS. If the security was conducted in the application layer, there would be a greater granularity concerning its use. For example, an application could choose a security type or otherwise could select none. Adaptation could mean the change of a specific configuration. This adaptation might be made to respond to a given context managing in this way the energy consumption and security.

1.2 Objectives and challenges

The main objective of this thesis is to create a framework that allows security adaptation at run-time. Also, this framework must take into account the usage of the low-power and short-range technologies previously introduced. Also, it must take into consideration a given context, such as the requirements of applications or the network status. The expected outcomes are:

- The measurement of the security algorithms to rank and used them in a given context.
- The resistance of external attacks.
- The accomplishment of secure communications using the rank of the previous algorithms.
- The measure completion of the node energy cost regarding static and adaptive security.
- The change fulfilment of the security algorithm in use considering various variables.
- The measure achievement of the computational effort of each security algorithm as example memory or CPU usage.

The principal contribution of this thesis is the creation of this framework to perform the security adaptation at run-time. The technologies to take into account are used at IoT, concretely at WSN. The context is related with the used variables necessary to take into account this security adaptation. This creation takes into account CoAP environments besides the standard security used with this standard. The selection of the optimal energy-aware security profile, given by this framework, is the expected result.

The final result achieved was the creation of an architecture module, named controller, which performs security adaptation and handles the security-energy trade-off, taking into account the previous context. A performed analytical validation validated this module, and the results are presented in chapter 5. Also, a prioritisation of the security algorithms with real energy consumption costs was made.

1.3 Work plan

An initial work plan was performed and is presented at appendices sections. Annexe A 6.1 displays the fulfilled tasks regarding the first semester, whereas annexe B 6.2 exposes the planned activities related to the second semester.

The blue pieces represent the research and scientific articles readings. The green ones are the meetings performed, and the single yellow task is the thesis writing work. Beyond these tasks, the Gant chart also presents the classes and exams of the first six month period. These aspects are regarding the first semester 6.1.

In annexe B 6.2, the green task represents the paper writing. The orange colour represents the thesis writing task. The jobs with the blue colour are the ones that compose the system realisation. This group of activities is composed of eight elements. This work plan was presented in an intermediate goal.

1.4 Structure of the thesis

In this section, the dissertation structure is presented. This structure is composed of six chapters, besides the bibliography and the appendices. The thesis starts with an overview regarding the IoT. Then, requirements and one architecture are presented beside a proposal to handle the main architecture component. Finally, this proposal is analytically validated. The following points accurately display the chapters regarding this line of thinking:

- Chapter 1: This chapter presents the context and motivation of the thesis. Also, objectives and challenges are presented beside the work contributions performed at the dissertation. All these points prepare the reader regarding security and energy consumption area beside the problem to be solved.
- Chapter 2: A state of the art is presented in this chapter regarding the IoT overview and works to handle the energy-security trade-off. Concretely, the main requirements, attacks and used protocols are presented. Then, works bearing in mind the energy, the security and the energy-security adaptation trade-off are exposed and discussed.
- Chapter 3: The requirements and one architecture are presented in this chapter, besides the specifications of the components. The gaps identified in the previous thesis chapter contributed to the architecture and objectives creation.
- Chapter 4: In this chapter, the controller proposal and one formalisation are presented, given the identified variables to be handled by it. Also, energy experiments were conducted to prioritise security algorithms. Finally, an algorithm to select the optimal energy-aware security profile is presented.
- Chapter 5: This chapter presents the analytical validation of the controller algorithm presented in the previous part of the thesis. In this part, the behaviour of the controller is analysed using simulated variables.
- Chapter 6: The conclusions and future work are presented in this chapter. Also, it shows the contributions made besides the objectives fulfilled.

This page is intentionally left blank.

Chapter 2

State of the art

This chapter presents the Internet of Things (IoT) overview, more concretely the presentation of the primary requirements, the attacks and the fundamental protocols used in Wireless Sensor Networks (WSN). Then, works are presented, bearing in mind the security energy trade-off in these networks. In conclusion, one comparison and discussion of these works are made.

2.1 IoT Overview

In this section, a presentation about the IoT is made. Initially, an overview of the prominent requirements is presented. These needs are organised at network or security requirements. After these requisites exposition, an overview related to attacks, and a presentation regarding the protocols are made. Here, it is taken into account open regulations, standardised mainly by entities such as Institute of Electrical and Electronics Engineers (IEEE) [Engineers and Electronics, 2018] or Internet Engineering Task Force (IETF) [Force, 2018], and industry standards. Also, in tables 2.3 and 2.4, these standards are presented, bearing in mind some requirements satisfaction.

2.1.1 IoT Requirements

Network requirements

The work [Rault et al., 2014] presents a survey about IoT fields, requirements, and low power WSN standards. The authors concluded that the main WSN requirements, transversal to all IoT areas, such as healthcare, Industry or environment and agriculture, are the following:

- Coverage - this need is related to nodes deployment. Sometimes, it may not be possible to perform this deployment in an optimally way, or the weather conditions can lead to the breakdown of inter-node lines.

- Latency - it takes into account the delay in a WSN. This delay must be minimal to mitigate the range between the detection and the mitigation of an anomaly or to report an alarm in an emergency application;

- Mobility - this requirement is transversal to many fields. In Agriculture, an animal can be monitored, or at a transport system, a vehicle can communicate with another;
- Quality of Service (QoS) - the need for traffic prioritisation is vital in some IoT areas, such as healthcare or transportation system. The critical data must have high priority in comparison with with lesser key data.
- Robustness - this need is related to where the devices operate. Many devices, bearing in mind areas such as the underground and underwater sensor networks, work under extreme conditions, and lower propagation speed, noise or path loss must be addressed. A network must keep the basic functionality even under the failure of components.
- Scalability - it is related to the enormous number of nodes in a controlled area. The primary objective is ensuring full coverage and communications;

Security requirements

The work [Rault et al., 2014] presents security requirements, bearing in mind the various IoT areas. Security mechanisms are needed to ensure data protection and user authentication in areas such as the healthcare area, to grant confidentiality and access control to data. This data is sensitive and must be kept private. Availability, integrity, authenticity or confidentiality of the data, bearing in mind the data sensitivity, must be assured in the industrial areas. Also, data reliability must be ensured to get correct results. The transport systems is another IoT area. Here, a network must be protected against data corruption. This need is created because of the false information about traffic or road conditions. Also, in areas such as public safety and military system, this need must be assured, that is why data integrity is needed.

In work [Sicari et al., 2015], the writers introduce works bearing in mind three security requirements: authentication, confidentiality and access control to data or resources. Also, taking into account the sharing environment present in IoT, the authors referred that authorisation, authentication, and non-repudiation, besides access control, are necessary to ensure secure communications. Finally, they defended that IoT privacy is paramount because one of the demands of the users is the protection of their personal information.

In [Airehrour et al., 2016], the authors address security requirements such as availability and authenticity of the network or confidentiality, integrity, and non-repudiation of the data. The network availability must be granted to ensure the survivability of all network services even on attack situation. This availability implies to permit available the network services in all layers to all network nodes. The authenticity is related to nodes. A node must realise the identification, demonstrating their identity, in the network, to prevent illegal nodes access. Confidentiality is assured using routing and data encryption. Integrity is related to the modification of data when it is in transit. This data must remain the same in the message exchange process. Finally, the non-repudiation property is associated with the knowledge of an entity when it is sent or received information. This knowledge cannot be denied.

[Nguyen et al., 2015] present, once more, security requirements such as confidentiality, integrity, authentication, authorisation, and freshness. An attacker should not gain knowledge of the exchanged messages, a message modification should be detected, and their origin should be checked. The authorisation is related to data access by entities. The freshness takes into account the replay attacks mitigation, denying the earlier messages reply. The same requirements are discussed in [Granjal et al., 2015], bearing in mind the flows of data or the WSN environments. For these environments, availability and resilience are essential. In this case, resilience is related to the business needs and the level of redundancy to satisfy these needs. With this in mind, a network, at its regular operation can provide and maintain a service level even in the presence of problems. Others requirements, such as privacy, anonymity, liability, and trust, according to the Authors, are relevant for the social acceptance of IoT applications.

[Dragomir et al., 2016] approach requirements on network security, identity management, privacy, trust, and resilience. The majority of these requirements have already been presented. The trust category, taking into account the authors, embraces data trust or entity trust. One is related to data collected by potentially unreliable devices, and the other is connected to the expected behaviour of entities, services or users. Anonymity is encompassed in privacy and is related to personal identification, so the data or action can not identify a body as a source. At identity management, revocation consists of to remove entity permission to operate. In contrast, authorisation permits to allow authenticated entities to perform operations. Finally, origin authentication is related to entity authentication and data authentication. The first one ensures the origin of the data and the second prevents the invalid data injection in the network.

The next table displays the analysis performed concerning the surveys presented, and each number corresponds to one identifier. Precisely, each work is identified by:

1. [Rault et al., 2014]
2. [Sicari et al., 2015]
3. [Airehrour et al., 2016]
4. [Nguyen et al., 2015]
5. [Granjal et al., 2015]
6. [Dragomir et al., 2016]

Table 2.1: IoT security requirements, given the introduced surveys

Requirements\Surveys	6	5	4	3	2	1
Access Control					✓	✓
Anonymity	✓	✓				
Authentication	✓	✓	✓		✓	
Authenticity				✓		✓
Authorization	✓		✓		✓	
Availability	✓	✓		✓		✓
Confidentiality	✓	✓	✓	✓	✓	✓
Freshness	✓		✓			
Integrity	✓	✓	✓	✓		✓
Non-Repudiation		✓		✓	✓	
Privacy	✓	✓				
Reliability						✓
Resilience	✓	✓				
Revocation	✓					
Trust	✓	✓				

The primary requirements were identified, bearing in mind the requisites presented in the previous works. This information is collected at table 2.1, and it is notorious that confidentiality, integrity, and availability are the most important. These three requirements form the CIA acronym [Dragomir et al., 2016], but authentication, authorisation, and non-repudiation are also essential and can not be ignored.

2.1.2 IoT Protocols

IoT is composed of many protocols bearing in mind the interoperability of any device or technology in global communication. These protocols were created and standardised by entities like IEEE [Engineers and Electronics, 2018] or IETF [Force, 2018], normally open standards, or by industry alliances such as Near Field Communications (NFC) Forum [Forum, 2018], ZigBee Alliance [Alliance, 2018c], Thread Group [Group, 2018b] or Lora Alliance [Alliance, 2018a] [Dragomir et al., 2016]. The majority of these protocols take into account the nodes restrictions, like energy consumption or security restrictions. In [Dragomir et al., 2016] was presented a survey of the most used communication protocols. Also, the authors present their security key capabilities, besides some security requirements. In [Granjal et al., 2015] is presented protocols and security mechanisms for the IoT, bearing in mind IEEE and IETF protocols. In this survey, apart from this, the authors also presented open challenges and strategies for future research, aiming the communications security in IoT.

In this sub-section, a brief presentation and description of the most used protocols in IoT is performed, take into account the previous works. Also, into table 2.2 is presented open standard protocols with the associated IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) stack layers. Tables 2.3 and 2.4 present a summary about the security requirements met by Open and Industrial standards.

Open-standards

- 6LoWPAN - 6LoWPAN [Montenegro et al., 2007] is an adaptation layer protocol that was standardised by IETF. This layer enables the transmission of IPv6 packets over IEEE Standard for Low-Rate Wireless Networks (IEEE 802.15.4) networks. The goal of this protocol is to perform the interoperability of the Internet and the constrained IoT sensing devices, enabling IPv6 end-to-end communications with other IPv6 hosts. Consequently, it is possible to connect various technologies, and it allows packets to travel through different networks. This protocol also implements mechanisms for packet fragmentation or reassembly, since it provides this interoperability. This Adaptation layer is placed between the data link and the network layers. This protocol does not have any security mechanism defined. The security of communications is performed with mechanisms used in other protocols presented in upper or lower layers.

- 6TiSCH - IPv6 over the time slotted channel hopping mode of IEEE 802.15.4e (6TiSCH) [Vilajosana et al., 2017] is a new IETF protocol under development. This protocol is expected to become the new standard for low-power wireless industrial monitoring applications [Dujovne et al., 2014]. It is a sub-layer allowing a scheduling entity to manage, in the network, the Time-Slotted Channel Hopping (TSCH) schedule. This schedule manages all communications in a TSCH network. For example, it can indicate to a node, in a specific time-slot, to receive or to transmit data or even to go to sleep. The protocol is based on IPv6 multi-link subnet spanning over high-speed IEEE 802.15.4e [Airehrour et al., 2016]. Details about the treatment of the packets or issues, like classification or forwarding of packets, will be included in the protocol. Also, other areas will be addressed, such as security, link management or neighbour discovery.

- CoAP - Constrained Application Protocol (CoAP) [Shelby et al., 2014] is an application layer protocol, standardised by IETF, and it is used with constrained nodes and constrained networks. It is a specialised web transfer protocol designed for Machine-to-Machine (M2M) applications such as smart energy or building automation. It provides a request/response interaction model between applications endpoints. Also, the Uniform Resource Identifier (URI) addresses are used to identify resources presented on constrained devices. Messages are exchanged asynchronously between two endpoints. Then, they can be marked with the purposed of ensuring reliability because CoAP gives a lightweight reliability mechanism [Granjal et al., 2015]. The protocol is defined only for User Datagram Protocol (UDP) communications over 6LoWPAN and to ensure Security is adopted Datagram Transport Layer Security (DTLS) at the Transport layer. With this adoption, the CoAP ensures confidentiality, authentication, integrity, non-repudiation, and protection against replay attacks with messages freshness. For this, CoAP defines four security modes: NoSec, PreSharedKey, RawPublicKey and Certificates [Dragomir et al., 2016]. In the first mode, the DTLS is disabled. In the second, each device is pre-programmed with symmetric cryptographic keys, one key for a destination or a group of destination devices. In the third, each device uses an asymmetric key pair and has a list of nodes for communication. Finally, besides the utilisation of an asymmetric key, the widget uses an X.509 certificate. The DTLS is enabled as of the second mode.

- DTLS - DTLS [Rescorla and Modadugu, 2012] is another protocol standardised by IETF, based on Transport Layer Security (TLS) and it aims for the security of the UDP-based communications. It is usually used with other protocols, such as CoAP. Packet reordering, packet loss or packet fragmentation, besides counter replay or Denial of Service (DoS) attacks are handled by the protocol. For these responsibilities, DTLS is considered one the most promising protocols for the IoT [Banerjee et al., 2017]. The DTLS is composed of two layers. One encrypts the Application layer payloads and fragments. Then, they are encapsulated into structured packets called records. This layer is called the Record Protocol. Besides the previous roles, the tier can provide message authentication. The second layer is called the Handshake Protocol. It allows a client and a server to negotiate security settings, to perform mutual authentication or to establish a secure channel for the exchange of encrypted records.

- IEEE 802.15.4 - IEEE 802.15.4 [IEEE, 2015] was designed to support the trade-off between energy-efficiency, range and data rate communications in constrained devices [Granjal et al., 2015]. This protocol operates at the physical layer and Medium Access Control (MAC) sub-layer of data link layer. At physical layer is performed the management of the physical radio transceiver, the channel selection and the energy and signal control. On the other hand, at MAC layer is made the management of the data service, the access to the physical channel, the validation of frames or the security. Also, the protocol supports different technologies, such as peer-to-peer or cluster networks. Also, it defines four types of data formats to be transmitted called frames. These can be data, beacon or even MAC frames. Because of this, the protocol lays the basis for other IoT protocols at upper layers [Granjal et al., 2015]. These protocols can define security services based on the Advanced Encryption Standard (AES) block cypher with a minimal impact on the resources of the device, such as the energy [Dragomir et al., 2016]. Another curious aspect of the protocol is, in the beacon-enabled mode, allowing the energy saving by implementing duty cycling. With this cycling, all nodes can, for time to time, go to sleep [Rault et al., 2014]. The security services are provided at the MAC layer, in particular, security modes providing different security guarantees. These modes use the AES algorithm. The protocol can ensure confidentiality, data authenticity and integrity by combining the security modes. Also, semantic security and protection against message replay attacks can be guaranteed in all security modes by the counter and key control fields of the protocol. These fields must be set by the sender, at auxiliary security header. Another curious characteristic presented in the protocol is the access control mechanisms, specifically the Access Control Lists (ACL). It consists of a table with 255 entries, and each one permits security communications with a particular destination. Finally, time-synchronised channel-hopping communication can be used for synchronised transmissions.

- IEEE 802.15.6 - The IEEE Standard for Local and metropolitan area networks - Part 15.6: Wireless Body Area Networks (IEEE 802.15.6) [IEEE, 2012a] is a new protocol optimised for low power devices, bearing in mind the health area [IEEE, 2012b], where is defined the physical and MAC layers. These low power devices work at proximity or inside the human body for serving medical and other applications purposes. With this in mind, this protocol is standardised for short-range and is necessary QoS, low power, data rate or non-interference to meet the variety of body area network applications. For this, the protocol uses Industrial Scientific Medical (ISM) bands approved by medical or regulatory authorities. Best-effort or event reports make part of the user priorities, and the protocol makes use of a minimum and a maximum contention window to differentiate these priorities. Also, the standard supports three levels of security. Authentication and encryption are provided at the third level [Rault et al., 2014].

- IPSec - The Internet Protocol Security (IPSec) [Kent and Seo, 2005] is another protocol specified by IETF. At the internet/network layer, the protocol offers integrity, confidentiality, and authentication. It can protect IP and transport layer header fields of a packet, creating secure channels. That is why it is suitable for non-trustworthy networks. However, the protocol is more expensive because a packet size can be increased due to encryption. However, it is possible to disable some security requirements in the IPSec bearing in mind the application requirements. The protocol can operate in two modes: in transport mode or tunnel mode. In the first, a packet IP header is left intact, and only some fields are protected such as IP addresses. With tunnel mode, IPSec performs the tunnelling of the entire packet. At this way, all IP header fields are protected. This mode is used for creating virtual private networks over the Internet. The IPSec, to ensure the security requirements, uses two protocols: Authentication Header (AH) and Encapsulating Security Payload (ESP). With AH the IPSec provides authentication, integrity and replay protection. With ESP, authentication, confidentiality, integrity and replay protection are provided. Confidentiality can be disabled to mitigate energy consumption [Dragomir et al., 2016].

- MQTT - The Message Queuing Telemetry Transport (MQTT) [Standard, 2014] is an International Organization for Standardization (ISO) [Standardization, 2018] standard, under development by the Organization for the Advancement of Structured Information Standards (OASIS) [Standards, 2018]. It is used in various IoT areas, such as health or social media and constrained environments. The protocol is based on the publish-subscribe model, using a broker that routes messages to clients, and provides "at most one", "at least once" and "exactly once" semantics or QoS levels. At first semantic, messages loss can occur and at last semantic is assured the arrival of the messages exactly once, so, without duplicates. The last semantic uses more bandwidth and energy. The first can be used with ambient sensor data. The MQTT runs over Transmission Control Protocol (TCP) but can run over other protocols. These protocols must provide ordered, lossless and bi-directional connections [Standard, 2014]. Authentication can be performed using a variable header from an MQTT message. This variable can contain a username and a password. Another version of the protocol, called SMQTT [Singh et al., 2015], addresses the security problems. Also, it is possible to use MQTT with TLS in order to ensure communications security [Dragomir et al., 2016].

- MQTT-SN - This protocol is an extension of MQTT. The MQTT For Sensor Networks (MQTT-SN) [Stanford-Clark and Truong, 2013] is used at wireless sensors, bearing in mind actuators on no-TCP networks [Rault et al., 2014]. Therefore, the protocol is adapted to low bandwidth, high link failures or sort message length. Also, MQTT-SN is optimised to be used on restricted, low-cost and battery-operated devices. That is why the protocol logic is handled on the broker and the gateways. Also, the standard can support sleeping clients and many gatewain the network. These abilities allow the minimisation of energy consumption.

- RPL - The standard IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [Winter, 2012] is a distance vector protocol compatible with IPv6. It was standardised by IETF, and it is focused on meeting the resource-constrained devices requirements [Airehrour et al., 2016]. This protocol operates at the network layer, providing an adaptable framework to applications requirements. The standard development was performed by the Routing Over Low-power and Lossy Networks (ROLL) working group of the IETF, and it is the current approach to carry out routing on 6LoWPAN environments [Granjal et al., 2015]. This protocol creates a Destination Oriented Direct Acyclic Graph (DODAG), which can be focused on energy efficiency. This DODAG consists of sensor nodes and a sink with the role of to gather information originated at other nodes. In this graph, each node has a DODAG id, a version number, a RPL instance id, and finally, a rank. Each rank is computed by each node, bearing in mind the parent rank and an objective function. The rank is helpful to manage control overhead, to prevent loop information or to create optimal network topologies [Airehrour et al., 2016]. That is why the rank affects the generated topology. It is possible to have multiple RPL instances running at the same time, on the same network, each one with its optimisation objectives. Topologies such as multipoint-to-point, point-to-multipoint or point-to-point are possible with the protocol. Also, the RPL supports many control messages types, bearing in mind the establishment of routes or the mitigation of packet replay attacks. These control messages are encapsulated at Internet Control Message Protocol (ICMP)v6 packets. The protocol uses three security modes: unsecured, pre-installed and authenticated. The first one has the purpose of minimising overhead because security can be provided by the link layer. The others add a security section at RPL header and count for the identification of the type of security used. With the combination of these security modes, the protocol can provide data integrity and replay protection with optional confidentiality and delay protection [Dragomir et al., 2016]. Also, data authenticity and key management can be supported [Granjal et al., 2015].

- Wifi - The 802.11 group of standards define the wifi communications [IEEE, 2016] [Intel, 2017]. These communications, many times, operate in wireless environments where the presence of interference is a constant reality. This interference leads to a degradation of performance. Usually, these standards work among 2.4GHz and 5GHz, in various channels, depending on the geographical regions. If these channels overlap, again, the presence of interference is notorious, bringing to low throughput or even connectivity loss. That is why some devices can select, in a dynamic way, other channels. Wireless Equivalent Privacy (WEP) or Wifi Protected Access (WPA) are examples of security protocols that can be used with wifi. However, WEP is obsoleted. Nowadays, WPA version 2 is the most suitable. This standard supports two authentication methods: personal or enterprise. The first one uses pre-shared keys, and the second requires a Remote Authentication Dial-In User Service (RADIUS) authentication server. WPA version 2 can provide encryption and integrity using the Temporal Key Integrity Protocol (TKIP) or Counter Mode CBC-MAC Protocol (CCMP). However, compared with other technologies, Wifi interfaces consume more energy. That is why wifi is not suitable for limited remote energy consumption devices [Dragomir et al., 2016].

- XMPP - This protocol is another popular standard, created by IETF, for instant messages. Extensible Messaging and Presence Protocol (XMPP) [Saint-Andre, 2011] is built over TCP. It uses the Extensible Markup Language (XML) as a data model, enabling the exchange of data between two entities. This protocol allows channel encryption, authentication, and error handling. Also, it makes available communication primitives for messaging, network availability and request-response interaction [Saint-Andre, 2011]. In a complementary way, the protocol also implements authorisation. Security communications are performed using Secure Sockets Layer (SSL) connections or START-TLS. In START-TLS, a normal and not secure connection is created, and then it changes to a secure one, after a handshake. This process allows a service with resources constraints to perform a secure connection. The XMPP is one of the most secure protocols for IoT device communication [Dragomir et al., 2016], but the XML application creates some overhead. Because of this situation, authors in [Dragomir et al., 2016] recommended the use of a lightweight protocol for internal networks, such as MQTT, and XMPP for public networks communications.

Table 2.2: Stack layers with IoT protocols

6LoWPAN stack	
Layer name	Protocols
Application	**CoAP**; HTTP; WebSocket; **MQTT**; **MQTT-SN**; CORE;
	DTLS; UDP; TCP; TLS;
Network\Routing	**RPL**; IPv6; IPv4; ROLL;
Adaptation	**6LoWPAN**; **6TiSCH**;
MAC	**IEEE 802.15.4**; **IEEE 802.15.6**; IEEE 802.15.4e;
Physical	**IEEE 802.15.4**; **IEEE 802.15.6**;

The previous table 2.2 presents the open standards, approached previously. It is possible to see the position of each protocol at each layer. The table was created, bearing in mind the references [Olsson, 2014] and [Granjal et al., 2015].

Industry protocols

- AMQP - Advanced Message Queuing Protocol (AMQP) [Vinoski, 2006] is an open messaging protocol and includes a protocol model besides a network protocol. Network protocol describes how the messages are composed, whereas the protocol model describes entities that are performing a communication. AMQP was created by a group of industry players, such as Red Hat [Red Hat, 2018] or Cisco Systems [Cisco Systems, 2018], named nowadays as OASIS [OASIS, 2012]. The publish-subscribe model or point-to-point communication is used to transmit messages. The protocol was created on top of TCP. It supports the at-most-once, at-least-once, and the exactly-once semantics. On top of the transport layer, the standard establishes another layer called messaging [Dragomir et al., 2016]. It can be implemented TLS over TCP at the protocol, but it is not suitable for IoT devices with limited resources. AMQP provides authentication and secure communications across TLS and Simple Authentication and Security Layer (SASL).

- BLE - Bluetooth Low Energy (BLE) [Gomez et al., 2012] was developed for short-range control and monitoring applications. This protocol was created by the Bluetooth Special Interest Group (SIG) [SIG, 2018], bearing in mind low-power solutions. BLE is used in areas such as healthcare, consumer electronics or smart energy and security. A controller and a host compose the BLE protocol stack. The controller implements the physical and pink layers, whereas the host implements the upper layers [Gomez et al., 2012]. These two communicates across a standardised host controller interface, named the Host Controller Interface (HCI). The standard operates in 2.4 GHz ISM at the physical layer. Here two types of Radio Frequency (RF) channels can be used: advertising channels or data channels. The first is used for device discovery or broadcast transmissions. The second is employed on bidirectional communications between connected devices. BLE specifies two mechanisms of security: LE security mode 1 and LE security mode 2 [Dragomir et al., 2016]. At first mode, security is applied at the link layer. This security supports encryption and data signing, using AES-128 block cypher or the Cipher Block Chaining-Message Authentication Code (CCM) mode with different levels of authentication. Message authentication, named Message Integrity Check (MIC), is used for data signing. It is computed over the link-layer payload. At the second mode, security is applied at the attribute protocol layer and only supports data signing. Also, a counter can be incremented for each message signed. This mechanism ensures protection against replay attacks, and it is applied only over unencrypted connections. Other modes and keys can be used by the protocol, at different layers to ensure security.

- ISA100.11a - ISA100.11a [ISA, 2009] is another protocol based on the IEEE 802.15.4 standard. International Society of Automation (ISA) [ISA, 2018] developed it, and its design includes security, reliability, support for many applications and multiple protocols or the use of open standards [Nixon and Round Rock, 2012]. Also, ISA100.11a has the network and transport layers based on the 6LoWPAN standard. ISA100.11a is a standard for control and monitoring applications in Industry. The protocol allows nodes to go into sleep mode because it uses a deterministic MAC scheduling with variable slot length [Rault et al., 2014]. Also, each node sends the estimated battery life and the associated energy capacity to the system manager responsible for allocating the communication links. The standard, besides the low power consumption focuses on network robustness bearing in mind the presence of interference.

- LoRaWAN - Long Range Wide-Area Network (LoRaWAN) [Alliance, 2018b] is the most adopted and most successful technologies in the Low-Power Wide-Area Networks (LPWAN) space [Adelantado et al., 2017]. It has a large capacity and long communication range, besides low energy consumption, and low cost. The protocol development is performed by the non-profit association LoRa Alliance. LoRaWAN networks are organised in a start-of-star topology where the gateways receive the end-devices messages. Then, these messages are sent to a central server [Dragomir et al., 2016]. This standard has two security layers. One for network and another for application. The first one ensures device authentication, and the other guarantee data confidentiality and integrity. A new end-to-end device added in the grid must be personalised and activated. This activation can be performed through Over The Air Activation (OTAA) or Activation by Personalization (ABP). The first one is used when a device is deployed or reset. The second includes personalisation plus activation. The customisation consists of providing information, such as a global and unique identifier or an AES-128 key, to a device. The AES-128 key is used for encrypting and verifying application data. Another relevant key is the network session key for a specific device. This key is used, by end-device and server, to compute and verify the MIC, besides other usages.

- NFC - NFC [14443, 2018] [18092, 2015] is composed of short-range communication technologies. This protocol is developed by NFC Forum association. An NFC device performs communications through electromagnetic fields, in an active or in passive communication. In an active communication, each device creates each own electromagnetic fields, and in the other way, a device transmits data by modulating the active device field. This standard is used to read or write data stored on tags. The authenticated data can be supplied by protocol, through the Signature Record Type Definition (RTD) 2.0 technical specification. This specification uses certificates, such as X.509 or M2M certificates, to sign information. Also, other cryptosystems are supported [Dragomir et al., 2016].

- Thread - Thread [Zimmermann et al., 2017] [Group, 2015] is another open standard protocol for low-power and wireless IPv6 communications, bearing in mind home applications. It has been created by Thread Group, composed of Industry players such as Samsung [Samsung, 2018] or Google's Nest Labs [Google, 2018]. Thread stack is formed by IEEE 802.15.4 physical and MAC layers, by 6LoWPAN, Distance Vector Routing (DVR), UDP and DTLS [Dragomir et al., 2016]. With 6LoWPAN, IPv6 headers are compressed to minimise the transmitted packets. IEEE 802.15.4 MAC layer is used, bearing in mind congesting control and basic message handling. UDP is utilised for messaging between devices. Thread MAC layer offers message confidentiality and integrity protection. The network layer provides reliable end-to-end communication, so Thread ensures data confidentiality, integrity, and authentication. Finally, the creation of connections or new network topologies can be performed by a node autonomously [Zimmermann et al., 2017].

- WirelessHART - WirelessHART [Kim et al., 2008] [Group, 2018a] ensures robust and reliable communications. Also, a simple configuration, a flexible installation or straightforward access to instrument data is offered [Kim et al., 2008]. The protocol provides a multi-layered approach for authentication, integrity, and encryption. The physical layer is based on the IEEE 802.15.4 specification. A Time Division Multiple Access (TDMA) is used at the MAC layer, and the nodes can go to sleep [Rault et al., 2014]. Another protocol characteristics are high reliability and power efficiency. The standard stipulates a central mesh network, controlled by the network manager. This manager gathers information about each surrounding nodes. With this information is created a global network graph where is defined as the graph routing protocol. Energy savings could be achieved, taking into account the Authors in [Rault et al., 2014] using the node battery-level information.

- Z-Wave - Z-Wave standard [Labs, 2018] is a low-power wireless communication protocol created by Sigma Designs [Laboratories, 2018], to allow home devices to interact together. The protocol specification and the software development kit are not open. On the physical layer, Z-Wave works in the ISM radio frequency band [Dragomir et al., 2016]. The topology used by the protocol is the mesh network with a controller device and 232 nodes. The controller has a unique home ID, which corresponds to a network identifier. This identifier is written by the manufacturer on the chip. Therefore it can not be changed by software. Data origin authentication and data integrity are ensured by protocol, besides freshness and data confidentiality [Dragomir et al., 2016].
- ZigBee - ZigBee [Alliance, 2012] is a low-cost, very low-power consumption, two-way wireless communication specification protocol, based on IEEE 802.15.4. It was created by the ZigBee Alliance, bearing in mind sensor networks applications. His security is developed on top of the IEEE 802.15.4 security services. More specifically, it uses the same AES-128 cypher and CCM mode to secure communications either at network or application layers [Dragomir et al., 2016]. The protocol thus provides a complete stack to ensure interoperability among different devices and to allow compatibility with IEEE 802.15.4 hardware. Also, ZigBee stipulates security levels, compared to the IEEE 802.15.4 levels, to achieve the same protections. However, the security level can not be changed, and it is familiar to the hole network. The standard specifies two types of keys bearing in mind data communication. A network key protects devices from others outside the network, and it is shared between all network members. A link key offers end-to-end security among two devices, that is why these types of keys are shared between devices pairs. Other keys are used by the standard to establish link keys or to secure messages containing other keys [Dragomir et al., 2016].

Tables 2.3 and 2.4 present the majority of the security requirements, fulfilled by the presented protocols. Recalling the IoT security requirements, presented at table 2.1, the ones that are more important were confidential, integrity and availability, besides authentication, authorisation or non-repudiation. The referenced tables prove the high importance of the referenced needs. The majority of the presented standards ensure authentication, confidentiality and integration.

2.2 IoT Attacks

Wireless Sensors Network are composed by resource-constrained nodes regarding memory, energy and processing capacity. These networks cover many IoT areas such as area monitoring, health care supervision, environment sensing or industry monitoring. WSN dominates M2M connectivity technology. This control occurs in the majority of these areas due to the use of low-cost devices. However, these networks handle sensitive data, and in many cases, they are in hostile environments. Also, the sensible data addressed by actuators and sensors can be distributed and processed with other networks or systems [Islam et al., 2012]. For these reasons, security is vital regarding the privacy of information. Internal or external attackers can execute many attacks with the purpose of to break the confidentiality or integrity of this information. The CoAP is one of the most used protocols in M2M applications such as smart energy and building automation

Table 2.3: Open standard protocols and security requirements

Protocols\Security requirements	Access Control	Authentication	Authenticity	Confidentiality	Freshness	Integrity	Non-Repudiation	Privacy
6LoWPAN [Montenegro et al., 2007]								
6TiSCH [Vilajosana et al., 2017]								
CoAP [Shelby et al., 2014]		✓		✓		✓	✓	
DTLS [Rescorla and Modadugu, 2012]		✓		✓		✓	✓	
IEEE 802.15.4 [IEEE, 2015]	✓	✓	✓	✓	✓	✓		
IEEE 802.15.6 [IEEE, 2012a]		✓		✓				
IPSec [Kent and Seo, 2005]		✓		✓		✓		
MQTT [Standard, 2014]		✓						
MQTT-S [Stanford-Clark and Truong, 2013]								
RPL [Winter, 2012]			✓	✓	✓	✓		
Wifi [IEEE, 2016]		✓		✓		✓		
XMPP [Saint-Andre, 2011]		✓		✓				

Table 2.4: Industry protocols and security requirements

Protocols\Security requirements	Access Control	Authentication	Authenticity	Confidentiality	Freshness	Integrity	Non-Repudiation	Privacy
AMQP [Vinoski, 2006]		✓						
BLE [Gomez et al., 2012]		✓		✓	✓	✓		✓
ISA100.11a [ISA, 2009]								
LoRaWAN [Adelantado et al., 2017]		✓		✓		✓		
NFC [14443, 2018] [18092, 2015]		✓						
Thread [Zimmermann et al., 2017] [Group, 2015]		✓		✓		✓	✓	
WirelessHART [Kim et al., 2008]		✓		✓		✓		
Z-Wave [Labs, 2018]		✓		✓	✓	✓		
ZigBee [Alliance, 2012]								

[Shelby et al., 2014], and it is positioned at the application layer. Our focus is related to attacks carried out on this protocol to ensure confidentiality, integrity, and authentication of communications.

2.2.1 Types of attackers

An attacker can be passive or active regarding his actions [Islam et al., 2012]. He can choose to passively steal data from a network eavesdropping communications, thus eliminating his presence in the network. In this situation, these types of attacks are challenging to detect. If he chooses to be active, the communication channel is influenced by his actions. Actions such as the modification or insertion of packets into the network are possible. Also, an adversary may be located in the same WSN domain and even the attacked device, or be external, found on the Internet or in another WSN domain [Granjal and Pedroso, 2018].

2.2.2 Types of attacks

An attack is an attempt to gain unauthorised access to a service, a resource or information [Diaz and Sanchez, 2016]. The purpose of this access could be an attempt to compromise the integrity, availability or confidentiality of a system. An attack can be:

- Passive – when an attacking player gathers or steals information by intercepting or monitoring packets exchanged between the nodes of a WSN. These actions are related to privacy, and the eavesdropping attack is one example of these attacks. Here the player attempts to learn or make use of the network information, not affecting the resources;

- Active – when an adversary performs actions such as the injection of incorrect data into the network. The impersonation or modification of resources and data-streams, the destruction of sensor nodes or the overloading of the network are other examples. These attacks mainly produce two effects: the traffic increase, due to the introduction of packets or noise, and the stream decrease due to the elimination and loss of network packets. Contrary to passive attacks, in this case, the attacker attempts to alter the network resources or affect their operation.

Bearing in mind the previous types of attacks, in particular, the active attacks, the authors in [Diaz and Sanchez, 2016] presented four categories related to the combination of the effects showed. An attack can introduce new packets, noise or the two in a network or even can modify the firmware of a node. Some examples of attacks regarding these categories were presented by the authors. They are summarised in table 2.5.

Table 2.5: Examples of attacks related with actions.

Attacks \Actions	Packets	Noise	Packets and noise	Firmware
Application attack				✔
Black hole		✔		
Collision attack		✔		
DoS		✔		
Energy drain	✔			
Flooding attack	✔			
Hello flood attack	✔			
Homing attack		✔		
Interrogation attack	✔			
Jamming attack		✔		
Looping in the network			✔	
Misdirection attack	✔			
Node replication			✔	
Overwhelm attack				✔
Resource extraction		✔		
Selective forwarding		✔		
Spoofed attack			✔	
Sybil attack			✔	

Attacks can be categorised by being outside or inside of the network, regarding the work performed in [Islam et al., 2012], beyond the previous categories. At an outside attack, the attacker is not a participant of the network. Whereas at an inside attack, the nodes are attacked by running malicious software on them. More concretely, inside attacks are initiated by authorised entities or nodes which are allowed to access network resources, but they use these resources in a way not approved. These type of attacks are stronger than external attacks because encryption and authentication are unable to detect it [Granjal and Pedroso, 2018]. This situation occurs because a captured node encrypts or decrypt transmitted messages between nodes of the network with valid and legally keys. Again, authors presented various attacks, bearing in mind smart home environments but applicable in any WSN area. At the following table is summarise some attacks, submitted by the authors, taking into account the categories presented:

Table 2.6: Attacks description regarding the attackers roles.

Attacks \Categories	Outside	Inside
DoS	✔	✔
Eavesdropping	✔	-
Node compromised	✔	✔
Physical attack	✔	✔
Sinkhole attack	-	✔
Wormhole attack	-	✔

A DoS attack can be outside or inside the network. As outside example, an attacker can use a laptop to transmit signals to interfere with the network channels used. If a node compromised performs many requests to other nodes in the network, we are in the presence of an inside attack. The aim can be the purpose of shutdown these nodes. These types of attacks can be detected and mitigated by analysing the transmission rate of received and sent packets [Granjal and Pedroso, 2018].

An outside attack must be carried on to compromise a node, physically reprogramming the nodes or remotely inserting malicious code into them. If successful, then this attack leads to others inside attacks.

A malicious node can attract network packets towards it by publishing false routes. When the packages pass through it, the node has access to all traffic and can make selective forwarding of packets. We are in the presence of a Sinkhole attack.

An attacking player can redirect the received packets to a network point, different from the actual location, thus creating confusion in routing or aggregation decisions made by the network nodes. The adversary reacts as a forwarding node and leads the network nodes to waste their energy. This attack is a wormhole attack.

Physical attacks are carried out by adversaries, that have physical access to network sensors. With this access, attacks can perform various actions such as inserting malicious code or retrieving secret information from the nodes.

Taking into account the work [Diaz and Sanchez, 2016], attacks that introduce packets in the network have the purpose of making the network nodes process them. In interrogation attack, an adversary sends Request to Send (RTS) messages to obtain Clear to Send (CTS) responses from a target neighbouring node. In the energy drain attack, an attacker can use compromised nodes to inject fabricated packets into the network. This situation causes false alarms that waste response effort and drain nodes energy. The purpose of the hello flood attack is to spend the power from a node or exhausts its resources using HELLO packets. These packets are employed in protocols for node discovery. The attacks that introduce noise have the aim of packets loss and disrupt the proper function of the network.

Our interests are on external attacks to guarantee confidentiality, integrity, authentication, and non-repudiation of the communications. The primary objective is to create and use the optimal security profile in a given context.

2.2.3 CoAP attacks

CoAP protocol supports low-energy wireless and M2M communications between constrained nodes. This protocol gives support to external internet devices, and it is compatible with the Hypertext Transfer Protocol (HTTP) [Granjal and Pedroso, 2018]. Because of this compatibility, security limitations of HTTP 1.1 [Fielding et al., 1999] are pertinent to CoAP [Shelby et al., 2014]. Attacks against caching and proxy model, such as men-in-the-middle are some examples to consider. Others attacks can distort the usage rules of the protocol, such as the sending of acknowledgements to a device when no CoAP request existed. Another example is the transmission of a request to a CoAP server using invalid options such as ACCEPT, Content-Format or Max-Age. Anomalies can occur on a client or server CoAP or both.

CoAP messages types

CoAP uses various types of messages regarding too many purposes, and some types of packages may be related to some CoAP attacks. Regarding RFC [Shelby et al., 2014], the protocol employs the following types of messages:

- Confirmable and Non-confirmable – the first one is related to the need for an acknowledgement or a reset message to confirm delivery. The second type is the opposite;
- Acknowledgement and Reset – An acknowledge message confirms the arrival of a confirmable message. However, it does not verify the success or failure of the operation transmitted, whereas a reset message confirms the processing or not regarding the sent message. When the processing fails, it is regarding the lack of context;
- Piggybacked responses – in some cases, the acknowledgement also transports a given response regarding a given request and in this way is not necessarily a new message;
- Separate responses – an independent answer is sent in a different exchange of communications when a confirmable request is acknowledged with an empty message. This situation occurs because a server, in some cases, may need more time to perform a request;
- Empty messages – these types of messages only contain a 4-byte header with the code 0.00, and they are not a request or a response.

However, the protocol defines only four types of messages such as Confirmable, Non-Confirmable, Acknowledgment and Reset. The responses can be carried on these types as well as in piggybacked in acknowledgement messages. The requests are just carried in Confirmable and Non-confirmable packages.

CoAP options

The protocol has various options regarding the resources identification, requests forwarding, or data freshness which are used by requests and responses. These alternatives can be used to perform attacks in CoAP networks, and that is why they are remembered taking into account the CoAP RFC [Shelby et al., 2014]:

- A target resource of a request is specified by the URI options (URI-Host, URI-Port, URI-Path and URI-Query). These options are related to the internet host, transport-layer port number, segment of the absolute path and the specification of arguments parameterising a resource;
- The Proxy-URI is utilised to prepare a request to a forward-proxy. This forward-proxy can forward the request on to another proxy or directly to the server. If the proxy recognises his servers names including aliases or IP addresses, request loops are avoided;
- The Content-Format Option indicates the representation of messages payloads. This option consists of a numeric content-format identifier defined in CoAP Content-Formats registry;

- The Accept option can suggest which Content-Format is accepted by the client. Here, a server returns, if available, the preferred Content-Format indicated;
- Another option is Max-Age. This alternative specifies the maximum time regarding the cache of a message. After this max time, this package is not fresh. 60 seconds is the default value in a response when Max-Age is not defined;
- The ETag or Entity-tag option is used as a resource-local identifier to distinguish the various representations of the same resource over time. The server creates this preference, and it can be used as a response option or as a request option;
- A relative URI consists of an absolute path, a query or both. The Location-Path and the Location-Query options indicate this URI. The combination of these options indicate the location of the created resources, and it is included in responses created from POST requests;
- With the conditional request options, a client asks a server to perform a request only if certain conditions are met. If a condition is satisfied, the request method is performed by the server in the usual way. The If-Match and the If-None-Match compose these conditional request options. The first one is used with the Etag alternative and is useful for resources update requests or for protecting against accidental overwrites. The second is relevant for resource creation requests;
- The size1 options is an integer number of bytes and provide size information related to the request resource representation.

Examples of attacks in the CoAP

In [Mattsson et al., 2018], attacks were presented, regarding the CoAP protocol, and the authors assumed that CoAP was protected with a secure protocol such as DTLS. The attacks presented were request delay, response delay and mismatch and finally the request fragment rearrangement. In [Shelby et al., 2014], was presented others security problems regarding proxying and caching, risk of amplification, IP address spoofing attacks or cross-protocol attacks.

The amplification attack Amplification attack is related to the exchanged of messages between the server and client CoAP. A server replies typically to a request, sending a packet with a response packet. This response can be larger than the solicitation, and an attacker may use this factor to implicate nodes in DoS attacks in networks with a limited amount of traffic. Since CoAP uses the UDP protocol, and it cannot verify the source address from a request packet, an adversary can place the IP address of the victim at the source address. Then, he can perform a DoS attack. We are in the presence of internal or external aggression, and the authentication can mitigate this attack if significant amplification factors not be provided in responses to not authenticated requests.

IP address spoofing attacks Many attacks regarding IP address spoofing can be performed to CoAP messages due to the lack of the handshake in UDP protocol. An adversary can spoof:

- A reset message in response to a confirmable or non-confirmable message with the consequence of an endpoint become deaf;

- A acknowledge in response to a confirmable message. This situation prevents the sender from retransmitting the confirmable message;
- An entire response with a forged payload or options;
- A multicast request for a target node thus creating network congestion or collapse or a DoS attack on the victim;
- An observe message.

If a randomised token is used in the request, the detection and mitigation of response spoofing attacks are possible. Also, if the CoAP uses confirmable message semantics, other types of spoofing can be detected.

Cross-Protocol attacks Taking into account the amplification attack, where a node sends packets to a fake source address, the same can be done to perform cross-protocol attacks. First, the attacker sends a packet to the CoAP endpoint with an IP address as the fake source address. Then, the endpoint replays to this source address. The machine at this address receives the UDP packet. However, this package is interpreted according to the rules of a different protocol. These attacks aim to circumvent firewalls rules that prevent direct communications from the attacker to the victim. If the endpoints do not authorise activities just focused on trusting the source IP address of a packet, these attacks became mitigated.

The block attack This attack consists of blockading a request or responses from clients and servers. An adversary, located between the two, can perform these actions. Also, this attack can be conducted by an attacker obstructing the lower layer radio protocol. If the players are sensors, this attack is a DoS, but if the parties involved are actuators, the client loses information regarding the status of the server. The client does not know if a door is unlocked or locked. This situation occurs when an actuator is a lock to a door. The client cannot distinguish the attack from offline servers, connectivity problems or unexpected behaviour from middle players such as firewalls or proxies. In this case, the confirmation is essential, and the user must be notified about the reception response or warned in case of not receive the response.

The request delay attack This offensive action is related with the Block attack, but an adversary can also delay the delivery of a request or response packet. If the CoAP uses security, using DTLS or Object Security for Constrained RESTful Environments (OSCORE) [Selander et al., 2019], other messages can be delivered before the delayed messages. This situation is possible if we take into account the replay window. The replay attacks are mitigated bearing in mind these windows and the use of sequences numbers by CoAP with DTLS. An attacker can control this window blocking messages. First, a request is delayed, then later, after a performed delivery of this message, the block of response is done. In this case, a client is not aware of the delayed request but is mindful of the missing answer. The server will process the order because it has no way to know about the request delay. This attack is a severe problem regarding actuators performing actions. If a client does not see the lock opening or do not receive the response message, the user will walk away to call for help.

Regarding the use of a non-zero replay window, an adversary can let the client interact with the actuator after the storage of the first request to unlock the door. The client will resend the same claim with a different or equal sequence number. When the client locks the door, an adversary can resend the first stored request and unlocks the door.

The second situation presented can be mitigated by using a replay window with a size equal to zero, but the first attack cannot. In this case, the use and exchange of timestamps between the client and server can mitigate the first situation.

The response delay and mismatch attack This attack is carried out by an adversary by delaying the response delivery until the client sends a request with the same token. In this case, the old response is accepted by the client as valid to the order. If a reliable and ordered transport is used with CoAP, the delivery of messages cannot be done before the delayed signal. Otherwise, other packages can be delivered inside the replay window. For example, a client sends a unlock request to open a door. Then a confirmation response is blocked by an attacker. Later, the client sends a lock request to close the door. Once again, the adversary jams the order and then transmits the old response from the actuator. Here, the client interprets this response as a confirmation of a locked door, but the proof is related to the first request, an unlocked door. The adversary can enter through the door because the client allowed the door unlocked. The same attack can be effectuated in sensors. A client can send a GET to see the door status, concretely if the door is locked or not. Then, the response status is blocked by an adversary and the client send a new PUT request with the unlock command. Once more, the attacker blocks the request status. Then, the client sends a GET request to know the door status. This request also is blocked. Then, the adversary sends the first GET blocked response with the status lock, but the door has the unlock state.

The relay attack This attack is performed in setups where actuators are triggered automatically, without user interaction and based on proximity. Taking a car as a client and the key as a server, the client, communicates regularly with the server. When this one responds, the client unlocks the doors and engine. An attacker may relay the CoAP messages out-of-band, and the actuator believes that the client is nearby and performs actions based on this assumption. This attack is severe because it leads to unlock and to drive away with the car. A response over a short-range radio is not a proof of proximity, that is why no actions must be done based on this proximity. The measure of the Round-Trip-Time (RTT) or the use of short radio range like NFC are some mechanisms presented by authors to mitigate this attack.

The request fragment rearrangement attack The combination of the request delay and the block attack create the request fragment rearrangement attack. This offensive action can be used against blockwise transfers to perform illegal operations on the server. Regarding the [Bormann and Shelby, 2016] specification, these blockwise are pairs of "Block" options for transferring various blocks of information such as firmware updates. These pairs extend the basic CoAP. Also, it can be used to perform responses to not authorised tasks to be mistaken for responses to authorised activities. The authors presented two scenarios for this attack. In the first scenario, blocks from two operations are combined to make the server to execute an action not intended by the authorised client. Here, the client sends a POST with "incarcerate" content and indicating that it will come more blocks. Then, the server replays with a continue block. The client sends the last block with the content "Valjean", but the adversary jammed this block. All data blocks transmitted by the client are blocked until the client sends another operation. Here, after the client sends the first block of the second operation with the content "Promote", the attacker blocks the server confirmation. Then, the adversary sends the last block of the first operation with the content “Valjean” to the server, and this executes the second operation with the last block of the first operation.

The second scenario presents a similar situation but using a withheld first block. Here, an action is initiated by the client, but the adversary blocks the first block. Then the client, after a time, starts a new operation sending a new block. The attacker prevents the server replay, and then sends the first block of the first action to the server. Afterwards, the adversary blocks the server replay and send this replay to the client. This one responds to the server bearing in mind the second operation, but the server returns and acts taking into account the first action. This attack is carried out even if the request/response pairs are encrypted or authenticated. The authors said that needs to tag the fragments to the server not mix up them.

2.3 Energy and Security

In this section, works bearing in mind energy consumption, the trade-off between security and energy consumption are presented. Some approach just the energy cost, whereas others just security. Finally, a few combine energy and security. The most interesting are the ones that performed the approach of the energy-security trade-off, not in a validation way, but in an adaptation way, taking into account the node resources or the network status.

2.3.1 IoT Energy-saving Mechanisms

Energy consumption, in IoT, is one of the primary focus. The devices restrictions, bearing in mind portable ones, are influencing factors to minimise the consumption. Also, the minimisation of economic costs, in industry, are relevant factors to take into account. Other technologies, bearing this in mind, have been studied and tested. Technologies such as small cells or millimetre wave communications were used in [Ge et al., 2014]. In this work, the authors analysed the wireless traffic backhaul and took into account the throughput and energy consumption. Taken into account these metrics, they built two scenarios, one centralised and another distributed. A macro cell and small cells constituted the first one. Here, all small cells sent wireless backhaul traffic to the macrocell, and then, this traffic was forwarded to the core network by fibre. In the distributed scenario, only small cells were taken into account. The traffic from adjacent small base stations was cooperatively forwarded to a specific small base station, and then it was sent again to the core network. In two solutions, all small base stations were uniformly distributed in a given area. The authors performed experiments, changing the radius of small cells and using various wireless frequencies. They concluded that the distributed solution had higher energy efficiency than the centralised solution.

The authors in [Banerjee et al., 2017] presented a low-variant of the DTLS protocol named Energy-Efficient DTLS (eeDTLS). First, they performed and described the DTLS protocol, pointing out some disadvantages, such as the present overhead for the resource-constrained IoT sensor nodes. Then, an energy analysis was performed, and getting experiments as a basis, the authors concluded that the elliptic curve cryptography algorithms are more expensive than symmetric algorithms. Also, they found that the handshake and the radio frequency transceiver are the primary energy consumers. Bearing this in mind, they performed packet reduction and handshake optimisation. This analysis was performed with

DTLS version 1.3 over BLE, and they choose to maintain the BLE headers intact. This choice, according to the authors, permits the most simple portability to different physical protocols, such as the IEEE 802.15.4 or to different link-layer protocols. The handshake optimisation was performed, eliminating the need to verify the Certificate Authorities (CA) signatures in the certificates and also reducing the certificate messages.

In [Tsikoudis et al., 2016], the authors presented a low-latency and energy-efficient Intrusion Detection System (IDS). With this system, they reduced power consumption keeping security. According to the authors, traditional approaches to low-power system design are not appropriate because they increase packet processing and queuing time. With this negative impact, the detection latency becomes worst and consequently impedes an early reaction. The main reason for this latency is high queuing delays imposed by high core utilisation. The fundamental idea, adopted by the authors, to mitigate this trade-off was, at first, identify the packets that were likely to carry an attack and gave them a higher priority. With this identification, the ones with the big priority would be processed first. Authors explored two alternative approaches to reduce the latency of packets with top priority: time-sharing and space sharing. The first approach uses a priority queue scheduling in each core. The second uses dedicated cores with much lower utilisation to achieved low latency. It is possible, using the modern network interface features, to move packets, with low priority, from interfaces to cores with higher usage. This process is called flow migration technique and was used to implement space sharing. The space sharing, after a comparison between the two approaches, had better power-latency when power consumption was reduced. The prioritisation of the packets was performed, bearing in mind the packet size. So, multimedia packets have a greater length compared with the others. Finally, the authors did experiments in an experimental environment bearing in mind the architecture validation.

2.3.2 Security

In this subsection, some works are presented regarding the security trade-off. However, these use energy consumption as a way to validate the proposal submitted and not to perform adaptation. The ones that handle this adaptation are presented in the next subsection.

Security is another focus in Fifth Generation (5G), including the IoT. Many technologies were studied and analysed in many works, bearing in mind the significance of this requirement. The usage of the physical layer with some of the most promising technologies, such as Heterogeneous Networks, Millimetre Wave (mmWave) or Multiple-Input and Multiple-Output (MIMO), is defended by the authors in [Yang et al., 2015]. Physical layer security provides secure wireless transmissions. Also it offers two significant advantages in comparison with Cryptography: these security techniques do not depend on the computational complexity, and they have high scalability. With the first advantage, the level of security will not be compromised by not authorised powerful smart devices present in the network. With scalability, devices join in or leave the network, in a dynamically way. These devices are connected to nodes with various specifications, and that is why the scalability is essential. Finally, authors identified opportunities and challenges that security designers must address, bearing in mind the usage of the physical layer with these technologies.

Nowadays, a WSN is a dynamic network composed of various and diverse restricted devices [Maerien et al., 2015]. These devices represent nodes, and they can own more than one owner. Also, a WSN can have multiple applications running at the same time, with one or various owners. As well, nodes owners can share capabilities of nodes, taken in mind the return of investment or services of these nodes. In consideration of these facts, trust establishment, access control, and security policy enforcement must be addressed. There are no appropriated security middle-ware solutions for these scenarios, because, the presented solutions are focused on a single owner, a single application or a static network paradigm. Authors in [Maerien et al., 2015], bearing this in mind, presented a WSN middle-ware that enables secure multi-party interactions on top of resource-constrained sensor nodes. Also, concepts, architecture, implementation, and integration of the components of this middle-ware were presented. The executions were evaluated with care to communication overhead, memory or processing overhead.

In [Gai et al., 2016] was proposed a higher-level framework for implementing secure Mobile Cloud Computing (MCC), based in IDS for protecting mobile applications, in a heterogeneous 5G network. According to the authors, this security can be achieved by adopting IDS techniques for applying mobile cloud-based solutions in the 5G networks. Also, they presented different IDS systems and methodologies, bearing in mind the challenges of each technique. Then, a comparison was made. Besides, they introduced MCC terminology in 5G, and the advantages and disadvantages were addressed, in addition to some issues. The Authors support IDS as an efficient approach for protecting 5G wireless communications. In Cloud Computing, techniques such as Parallel Programming, Virtualisation or Mass Distributed Storage were presented. In the IDS system, were discussed methodologies such as Signature-based detection or Stateful Protocol Analysis Detection.

A new encryption method was presented in [Elhoseny et al., 2016] to secure data transmissions in WSN composed by dynamic clusters nodes. In this method, authors utilised elliptic curve cryptography (ECC) for the generation of binary strings. Encryption keys of 176 bits were created using these strings and network properties. Encryption and decryption were performed utilising the exclusive or, the substitution and the permutation operations. Encryption was conducted in the sensor nodes and only decrypted in the base station. The role of cluster head nodes was the data collection from sensor nodes, without carrying out the decryption, because homomorphic encryption was integrated. Also, this type of encryption avoids attacks and saves energy consumption because it was not necessary to recover data from the cluster member nodes. This new encryption method was based on another work that used a genetic algorithm. This genetic algorithm had the role of balancing energy consumption and extending network life. Authors conducted experiments and demonstrated the improvement of network lifetime, the reduction of energy consumption, and finally, the capacity of the system to resist various attacks.

2.3.3 Security-Energy trade-off

A self-adaptation mechanism was proposed and implemented in [Taddeo et al., 2010], [Taddeo et al., 2011b] and [Taddeo et al., 2011a] for WSN. With this mechanism, a node can be adapted in run-time bearing in mind its power constraints. This adaptation was performed in an automated way and consisted of choosing the best cryptographic algorithms according to applications requirements and system energy consumption optimisation. Initially, the most robust security algorithm was selected and then was performed downgrades or upgrades, changing other configurations. This reason is security adaptation, but system workload also could be adapted to killing or preempting tasks. The cryptographic algorithms were ordered based on energy consumption and routines were distinguished between critical and non-critical by a priority level. The critical ones were essential to the system operation, and that is why they could not be adapted. The various configurations were determined by policies. Actions, specified by policies, were enforced to meet adaptations goals. These aims consisted of maximising the number of running tasks or increasing the security level of each routine, but each configuration had to respect a consumed energy range. The system was composed by a monitor, a controller, and an adapter. The first one was used to monitor system parameters and to initiate system events in predefined situations. The controller was used to receive these events and to build strategies to meet system and applications goals. The third had the role of enforcing the strategies or adaptation decisions, thus changing security changes, killing or suspending applications. Authors presented security considerations of the self-adaptation mechanism. With adaptation, weakest cryptographic algorithms may be chosen to bear in mind energy consumption. This point could expose the data to attacks, but security degradation is only conducted if the system energy constraints cannot be fulfilled. Another issue is related to multiple requests. These can consume battery node, forcing the system to adopt adaptation.

In [Taddeo et al., 2011b], the authors extended the work presented previously. They kept the same approach, but because of the limited configurations of parameters and the definition of most straightforward policies, applied to all tasks, they performed changes. New measures were created bearing in mind the characteristics of routines, especially energy consumption and resource usage. Also, task period was introduced for system workload gradual adaptation, bearing in mind the execution period of periodic tasks. In [Taddeo et al., 2011a] was presented a more complete and dynamic framework take into account [Taddeo et al., 2010]. Here was altered a sample period to periodic tasks, was inserted a dynamic energy budget, a dynamic monitoring period or the creation of adaptation decisions policies for each task. In this case, the system workload was adapted by modifying the periodicity or the suspension of tasks. The energy was estimated by each job and was performed a separation into dynamic and static energy. The static was associated with essential nodes functionalities. On the other hand, the dynamic could be optimised because it was associated with the radio module, the sensors, encryption or decryption.

In [Muradore and Quaglia, 2015], the authors proposed an energy-efficient security-aware wireless control architecture. Using encryption to protect packets is weighted to battery-power devices. With selective encryption, energy is saved and allows attack detection. Adapting the number of encrypted packets according to the presence of an attack allows a more significant energy consumption when needed. Also, energy consumption depends on the packet transmission rate, so they proposed the adaptation of this rate when the instantaneous control performance is better than desired. Authors had focused on deception attacks. This attacks, according to authors, are complicated to detect because they can be a natural disturbance. A deception attack affects data integrity of packets because the payload is modified. The architecture proposed can identify the begin and the end of an attack, and can react against it. This detection is performed based on the comparison between encrypted and no encrypted anchor packets, sent at the same constant frequency. Encrypting all packets of a flow, during an attack, except some anchor packets allows detecting when an attack is over. When the network is not being attacked, the encryption performed is minimal. So, an attack is detected by comparing the statistical properties of encrypted and not encrypted sequences. If an attacker does not change packets, the statistical difference is minimal. This architecture, taken into account authors, can be embedded in smart devices following others works. The carried out Simulation performed the validation of the architecture.

The security-energy trade-off can be managed in a dynamically way, taking into account the works presented previously. Having a cryptographic algorithm list, after performing an ordering of these algorithms, by energy consumption, by computational power or security level, allow choosing the best algorithm in a given context. This context can be the battery status of a given node, an attack occurrence or a set of applications requirements.

In the first three previous works, the approach adopted was the following: at first, start with maximum security and then perform the adaptation. This adaptation was carried taken into account a cryptographic algorithm list and task prioritisation. This prioritisation allowed task management in the system. The task management was carried out by killing, preempting, changing period or enforcing policies to tasks. The priority tasks usually were not modified. This management also involved more parameters such as a dynamic or static energy budget, so the Authors identified what could be changed. Finally, these changes were enforced. In the last presented work, authors performed a different approach: at first, the encryption was minimal, then, in the presence of an attack, full encryption was used. After the end of an attack, the performed encryption returned to be minimal bearing in mind the energy consumption minimisation. An attack was identified using an ingenious way. Two sequences of packets are sent at the same frequency. One of them is not encrypted and has one half temporal window of the encrypted. If the statistical difference between these two sequences is very different, an attack occurred or are occurring. Then, these not encrypted anchor packets could detect the end of an attack. In this approach, the authors also identified the packet transmission rate as a modifiable parameter.

With these approaches, taking into account the works presented, the security-energy trade-off can be managed in different ways. This management is essential, bearing in mind WSN devices since these widgets have limited resources.

2.4 Discussion

In the previous section, proposals regarding energy consumption, security and security-energy trade-off were presented. In a nutshell, distributed solutions can have better energy efficiency, and the use of symmetric cryptography has lesser energy consumption. The primary source of energy consumption is radio communication and the data processing of packets. That is why the prioritisation of packets or packets reduction was fulfilled. Security can be used at the physical layer with a lesser impact regarding energy consumption, providing scalability or independency regarding computational complexity. Homomorphic encryption and elliptic curve encryption can be used to achieve security with reasonable energy consumption.

A self-adaptation mechanism was presented regarding the security-energy trade-off. Also, some extensions of the same arrangement were presented. [Taddeo et al., 2010] chose the best encryption algorithm according to application requirements and system energy consumption optimisation. Also, authors prioritised tasks, removing or preempting them. The energy was estimated for each job and was performed a separation into dynamic and static electricity. The first one was regarding the radio module, sensors and encryption or decryption tasks. The second was related to essential nodes functionalities. Finally, energy-saving and attacks detection are possible using selective encryption.

The security works presented do not have into account performing security at the application layer, particularly at CoAP protocol. They address security in the lower tiers. The CoAP is used with constrained nodes and constrained networks because it has suitable energy consumption. There is an absence of works bearing in mind security achievement at the application layer, especially at CoAP.

The lack of works bearing in mind energy-security trade-off, especially adaptation ones, is a reality. In the previous section, they begin with the most robust security algorithm. Only then it was performed adaptation. In my opinion, the initial security configuration must be chosen, taking into account application security requirements and not the most secure algorithm. Another point is regarding the use of weak security algorithms when the energy constraints of the system cannot be fulfilled. Again, if an application needs strong security, this security must be accomplished regardless of the energy constraints event if a node perish.

On the other hand, an application may not require security, and it can be disabled. The works presented prioritised tasks, and they performed a separation into dynamic and static energy. They associated the sensors, radio module and encryption or decryption tasks to a variable power. However, if we think that each application has its exchange rate, the previous reason could not be necessary. Selective encryption was used to detect attacks or to perform energy-saving. If an application needs security, all packets will be encrypted. If security is not necessary, they will not be encrypted. What is essential, from a different angle, is the security strength. An application can need robust, regular or weak security. Another point of view is regarding the detection of an attack. If an IDS is used, a given node does not need to carry out this task.

2.5 Summary

This chapter presented an overview regarding the IoT, in particular regarding WSN. It started with identifying the network requirements and security requirements. Then, the most critical needs were identified, mainly the security ones. Also, a survey regarding the protocols used in IoT was performed. This survey allowed the reader to keep track of open or industry standards and perform the connection with the identified requirements. Then, a search regarding attacks performed on IoT was done.

With this background, we proceeded to the identification of works regarding energy-trade-off handling. This identification was handled in three stages. First, it was carried on the tagging and analysis of submissions regarding just to energy-saving mechanisms. Then, we proceeded to proposals regarding just security and finally, with the merge of energy consumption and security. The chapter became completed with a discussion. In this discussion was identified the absence of works regarding security achievements in the application layer, concretely at CoAP, besides the handle of energy trade-off. This handle is especially regarding adaptative and not to validate ones. The next chapter presents requirements and a top-level architecture in consideration to previous aspects, identified in the discussion section.

This page is intentionally left blank.

Chapter 3

System architecture and methodologies

Given the IoT overview presented at subsection 2.1 of the previous chapter, and taking into account proposal [Granjal et al., 2013], one architecture and mechanisms are proposed to manage the energy-security trade-off. This management is to be performed in a dynamic form, taking into account IoT and CoAP environments. The central objective of this thesis is to create and implement a framework taking into consideration these mechanisms. The aim is to adjust security and energy consumption. This adaptation takes in mind various variables such as the level of the battery nodes, network status or the requirements of applications. The premier technology focus is low-power and short-range mechanisms such as IEEE 802.15.4, 6LoWPAN, DTLS and CoAP. Then, an architecture is proposed in the context of which the energy management mechanisms, as well as, the different strategies regarding security, will be implemented and evaluated.

3.1 Objectives

Performing a short overview at CoAP protocol, this standard is designed to carry out M2M communications, providing a request-response interaction model, where the messages are exchanged between two endpoints. The resources are identified or discovered by a URI. The standard was designed for UDP communications over the 6LoWPAN and DTLS is used to ensure security. This security can be disabled using NoSec security mode at CoAP. In turn, the DTLS is based on TLS, that is why the protocol can handle packet loss, reordering or packet fragmentation. Also, the standard can tackle some attacks such as counter replay or DoS attacks. It is constituted by the record protocol layer, responsible for the encryption of the application layer payloads, and by the handshake protocol layer, responsible for the security negotiation between two entities.

In proposal [Granjal et al., 2013], the authors presented a way to integrate security at the application layer, concretely on CoAP. This security was based on new CoAP security options and security profiles created to take into account the requirements of the applications. Thereby, one application can request only data integrity or confidentiality plus integrity. Also, disabling DTLS security on CoAP allowed to have more payload space in the application layer. This security application layer proposal emerged, bearing in mind DTLS limitations. Authors identified the lack of adaptation security to the application requirements, bearing in mind the type and contents of an application message. This need emerges because when the CoAP uses security through the DTLS, this security is performed on all packets, in a given security session, for all CoAP communication. This mechanism requires the use of static security configurations in an application. Another issue derived from the static security configuration is the incompatibility of this security with the employment of CoAP intermediaries. In the proposal, the authors quantified the energetic and computational impact of the CoAP safeness and the DTLS security. Also, packet payload space usage was analysed, taking into account the minimisation of fragmentations at the 6LoWPAN adaptation layer. Authors demonstrated good results of the changes and most of the time, CoAP security was better than DTLS security.

The primary objective is to create a framework that allows security adaptation at runtime, taking into account a given context and using the same low-power and short-range technologies previous approached in this section. The main idea is to use standard security of CoAP with DTLS and extend the previous proposal. In other words, the framework will allow security self-adaptation, according to various conditions, external and internal, to device employed in the context of the IoT application at hand. This security can be the standard approach used by CoAP or the method used in the previous work. This change is performed according to a cryptographic algorithm list and a given context. This context can be a network status, more concretely if a network is being attacked or not, the security and functional requirements required by a given application or the battery status of a device. So, this context will impose the security mechanism in a given moment.

The implementation of the security mechanisms aforementioned will be experimentally evaluated and validated according to the mentioned technologies and the context. Specifically speaking, in a dynamic way, these mechanisms should be able to:

- adapt the security mechanisms and related configurations according to requirements of applications and energy present in the device;
- change security according to characteristics or available resources on the equipment;
- adapt security, for instance, algorithms employed and its run-time configurations, in the presence of external attacks.

The related configurations apply to security protocols and associated setups. These configurations are associated with keys size, keys refresh frequency or even with the selection of the cryptographic algorithms. Finally, applications will identify or request functional and security requirements using appropriate profiles. These profiles will be used to initiate communication protocols and security mechanisms. Thereby this consists of to adjust security level to an application needs. Briefly, the new framework will be able to use and switch between a standard security approach and a new approach. Secondly, this switch will be performed taking into account the presence of external attacks, the available resources of a given device, a given cryptographic algorithm list, and lastly, requirements of a given application. The final objective is ensuring security, in an adaptable way and extending the battery life of a device while guaranteeing appropriate security, according to the requirements of the application.

3.2 Requirements

In this section, the functional and security requirements of the system are presented. The first ones are related, especially with the behaviour of the applications and devices, whereas the security needs are concerned with the whole system.

3.2.1 Functional requirements

In this subsection, the identification and the presentation of the operational requirements are performed. These requirements are related to nodes and 5G applications used at these network nodes.

Applications

The system can have various applications running at the same time, and these can communicate with one or more than one implementation. Given this situation, an implementation should be able to:

- Perform the choice of the initial security level. Individually, an application can choose the level and what security demands it needs. For example, an implementation A can require encryption and another may not need protection at all. Also, an application A might need high encryption, and implementation B might require standard integrity. The requirements of a given application will ensure the initial security level.
- Conduct communication with a specific application. An application is running at a particular node. If this communication is performed for the first time, the application identity will be stored for future connections. If two implementations communicate securely, using symmetric cryptography and the used key is known just by the two, it is possible to save this relation at a node. Then, each application can access its information.
- Create the message signature. This step is performed to ensure the message origin. If an application communicates with another, the receiver must be sure about the source of the message received. This feature provides the communication distinction about the various implementations, as well as integrity or authentication.
- Add-On or remove applications entities which running at other devices. As discussed previously, each application has access just to his information regarding pairwise communication with different implementations.
- Realise the delivery of its entity. Also, the demand for entities can be performed. A list with all implementations and the pairwise keys can be uploaded to nodes to respect this need. Another approaches to fulfil the requirement is to use external services, such as public key infrastructure or even the Diffie-Hellman key exchange method [Chung and Roedig, 2007] [Kumar and Singh, 2016].
- Perform the exchange of messages. Each application has its exchange rate and types of messages.

As previously said, an application entity corresponds to public keys, certificates or the pairwise key used with symmetric cryptography.

Node

A node, in this system, should be able to:

- Perform security adaptation. This adaptation will be made by taking into account the actual context. As previously presented, this context can be the network status, the requirements of the application or the battery level of a node.

- Measure the battery. This measure will be performed, taking into account the exchange rate of an application. If the implementation has an exchange rate of six messages per minute, the measurement of the battery will be performed with the same ratio.

- Fulfil needs of the applications. As previous presented, each application has its requirements. The node will interpret and enforce these requirements. Each set of needs will result in various security profiles. Also, the node will notify the applications regarding the implementations of the demands.

- Receive the notification regarding the occurrence of attacks. An IDS can perform this notification. The node will receive notifications from an external device where the IDS is running.

- Perform the permanent establishment of a secure connection with the neighbours. This connection is separated from applications communications. For example, a DTLS connection can be established between the nodes using certificates or keys.

- Carry out the communication with other neighbours nodes. This communication is performed with the previous permanent secure connection. The aim of the correspondence is the exchange of information. This information embraces the security level, the cryptographic algorithm in use or the occurrence of an attack.

3.2.2 Security requirements

The system, bearing in mind the needs presented previously will:

- Provide end-to-end security between applications and nodes. This requirement is performed if the secure communications between the implementations are conducted. The same is true regarding the secure connection between the nodes.

- Enforce message confidentiality, integrity, authentication and non-repudiation. These properties are related to the requirements of the applications. If their needs are fulfiled, the same applies to this requirement.

- Ensure protection against external attacks. The majority of these attacks are mitigated with encryption and hashing. Encryption assures confidentiality, whereas hashing ensures integrity. If the two are used, we can have authentication and non-repudiation as long as the used key is known just by the two parties involved.

- Provide support to the use of low-power communications technologies. These technologies are the 6LoWPAN, the RPL, the DTLS, and the CoAP, and they were presented previously. This support is provided due to the restrictions of the WSN devices.

- Ensure DTLS security and CoAP message security. The main idea is ensured the two types of protection. Then, an application will choose between the two approaches. As previously presented, the first one provides static security, whereas the second offers adaptive protection.

- Support the use of various cryptographic algorithms. The same applies to the related configurations according to standards and the initial proposal presented earlier. At WSN, many security algorithms are used, and one of the most used is AES. Other algorithms are used, such as Triple Data Encryption Standard (3DES) or Secure Hash Algorithm 2 (SHA2).
- Brace interoperability, bearing in mind the attacks warnings and the performing communications with the other nodes and the base station.

3.3 High-level Architecture

In this section, a high-level architecture is presented to carry out the requirements presented previously. It is presented in image 3.1 and represents the whole system.

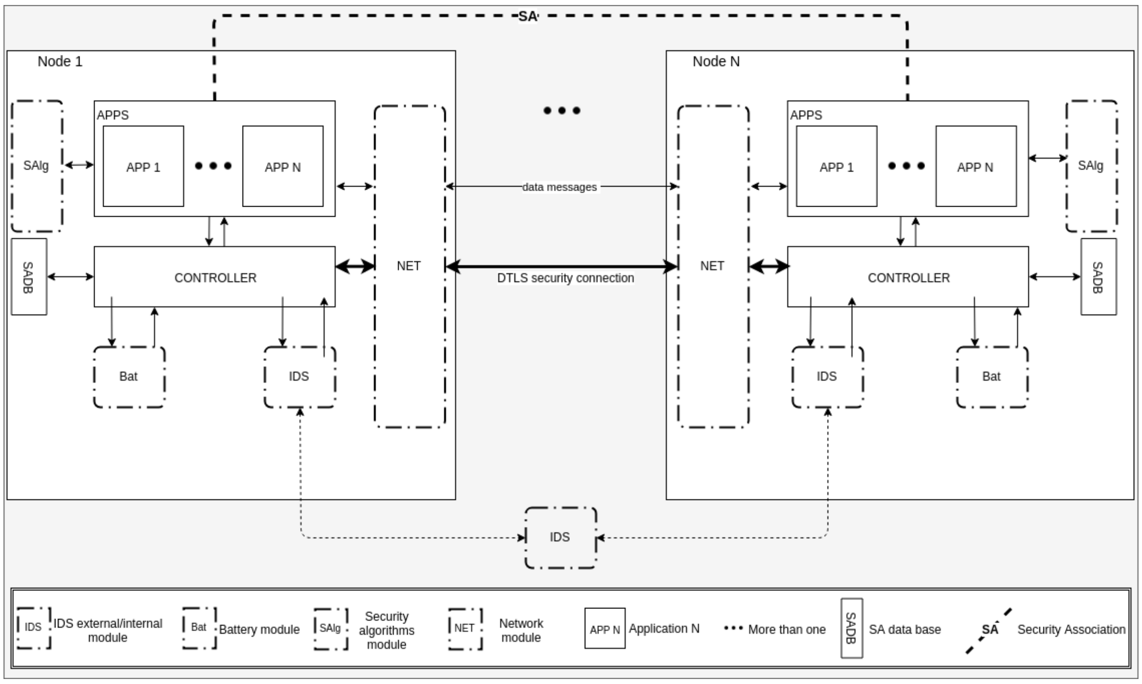

Figure 3.1: High level Architecture

As described at objectives section 3.1, applications will identify or request functional and security requirements using appropriate profiles. Given the high-level architecture presented, an application sends its requirements to the controller. After to attempt to apply the given specifications, the controller informs the implementation regarding the result of this operation. Figure 3.2 shows this described process.

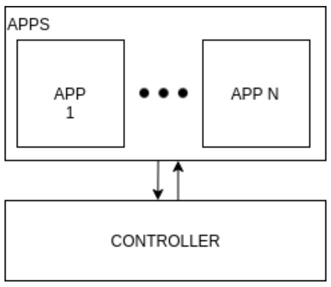

Figure 3.2: The interaction between the controller and applications

Image 3.3 presents the modules used by the controller. The battery and IDS modules will be used to inform the controller about the battery level of the device and the network status. This status may be an attack occurrence. These modules will not be developed in the context of this Thesis as the primary focus is the security self-adaptation component.

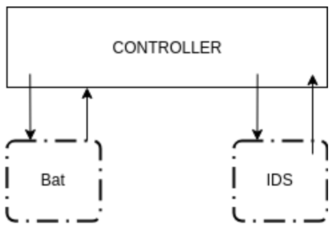

Figure 3.3: The interaction between the controller and external modules

Each application has a Security Association (SA) with other applications to carry out security communication. This association is represented by the connection with the SA abbreviation and is necessary to respect the previous requirements. Figure 3.4 presents the pieces of architecture involved. When an application creates a security association, the controller is informed, and then, this association must remain at the device until its removal, by the same implementation. The Security Association Database (SADB) module guarantees this role. This SADB has all security associations of a given device, in a given moment. This table will have at each line information about these associations such as the ID application of the sender and receiver, the security method, for instance, DTLS security or application security, the cryptographic algorithm in use, the cryptographic key, the signature algorithm or the signing key.

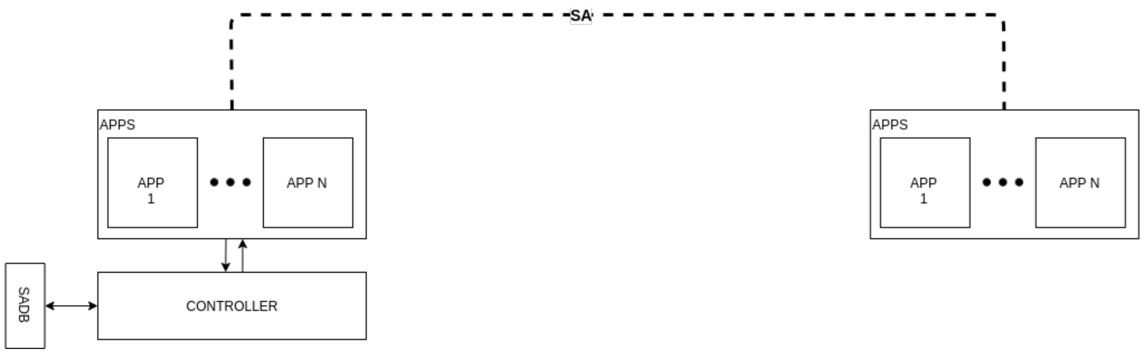

Figure 3.4: The secure associations regarding applications

The network module is used by applications and the controller. This module has the low-power communications technologies need to perform communications between the parties involved. These technologies were presented previously. The security algorithms component has all the necessary algorithms to ensure the requirements of applications. These requirements were given prior so that this module will have encryption and hashing algorithms. Again, these components will not be developed in the context of the dissertation.

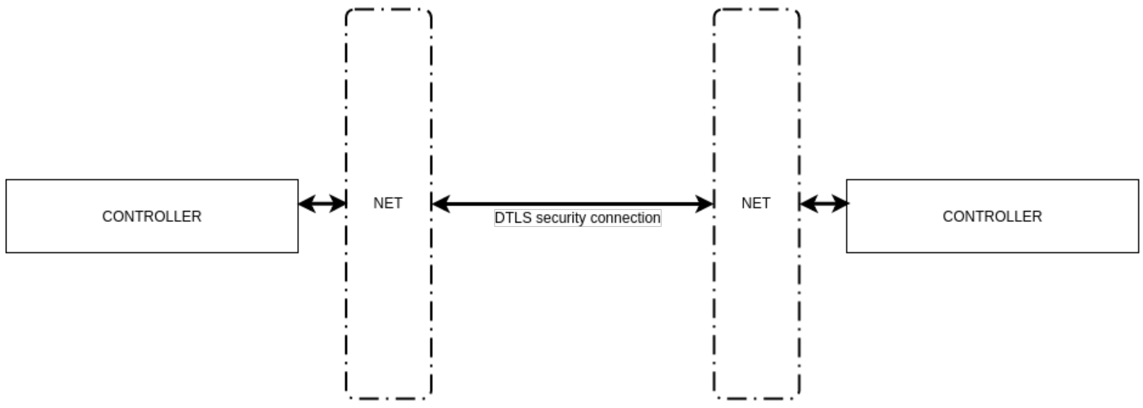

Figure 3.5: The communication between controllers

Finally, the controller is the primary objective of this work because it is the most vital component of the architecture. All security adaptative logic depends on its development, and that is why we started with it. This module is composed of decision algorithms that will be developed and implemented. They will manage security according to the energy of the device and the previous aspects addressed. In a nutshell, the controller will handle the requirements of the applications, the network status, the battery level, and it will connect, in a permanent way, to others controller nodes. This connection is presented in image 3.5 and will be used DTLS security with pre-shared keys. The aim of this connection is informing the others controllers about the changed context. This context, in this situation, can be the change of the security level with the proper characteristics. The primary objective, beyond the previous ones, is to perform the security adaptation. This adaptation might be an application switching from DTLS security to application security or changing the cryptographic algorithms in use. The focus is to provide security, bearing in mind the energy consumption minimisation.

3.4 Implementation and Validation

In this section is presented how is performed the implementation and validation of the previous architecture. Concretely, in this section, is shown the decisions regarding the hardware and operating system to be used. Also is exposed a discussion regarding the architecture components and the metrics to validate all architecture.

3.4.1 Hardware selection

The implementation will be performed at IoT-LAB platform [IoT-LAB, 2018]. This platform allows the use of real hardware nodes. These nodes are accessed by remote mode, and it is possible to assemble and deploy the firmware using the same manner. The platform is a scientific testbed [Fambon et al., 2014] where it is possible to conduct experiments with a set or sets of network nodes. This testbed is spread in various cities in France. Besides the usage of the network nodes, the platform allows the supervision of them. This supervision covers the energy consumption, radio monitoring and radio sniffing using profiles. However, we do not have more control, besides the accessible tools. In the past, support regarding the use of the nodes with batteries was given. Presently, this support is not provided because the nodes are plugged to the power grid.

Table 3.1: The main nodes provided by IoT-LAB

Properties\Nodes	WSN 430	M3	A8
External memory	1 MB external (STM25P80)	128Mbits external (N25Q128A13E1240F)	Co-microcontroller M3
MCU	16-bit Ultra-Low-Power, 48KB Flash, 10KB RAM	ARM Cortex M3, 32-bits, 72 Mhz, 64KB RAM	High-performance ARM Cortex-A8 microprocessor, 600 Mhz, 256 MB
Power	3.7V battery (830 mAh)	3.7V LiPo battery (650 mAh)	3.7V LiPo battery (650 mAh)
Radio	860 MHz\2.4 GHz (TI CC1101\TI CC2420)	802.15.4 PHY standard, 2.4 Ghz (AT86RF231)	Co-microcontroller M3

The platform provides, mainly three types of nodes. Table 3.1 summarises these nodes. The WSN430 is losing support, that is why the platform managers favour the M3 node utilisation. Also, other boards can be used.

The nodes selected to carry out the implementation are M3. This choice was made because of these types of nodes have more memory. Also, they present support to low-power and short-range technologies, such as the IEEE 802.15.4. Regarding the reducing WSN430 nodes support, there is a higher number of M3 nodes. Consequently, more experiments can be conducted by users. Comparing the M3 node with the A8 network node, we see that the last one is more powerful than the first. Also, the M3 node can be used on A8. The A8 node was not chosen because it does not represent a WSN device with the respective constraints.

3.4.2 Operating System selection

The Contiki Operating System (OS) [Contiki OS, 2018], the RIOT [RIOT OS, 2018], the TinyOS [TinyOS, 2018] or FreeRTOS [FreeRTOS OS, 2017] are examples of OS used in WSN. Given the previous decision related to M3 nodes, these nodes can use four operating systems. These are the FreeRTOS, the Contiki, the RIOT and the OpenWSN [OpenWSN OS, 2018]. At this moment, The RIOT system runs on more devices. The Contiki OS runs in three types of nodes and the FreeRTOS as well the OpenWSN run in two.

Contiki OS was selected regarding the selection of the operating system. This operating system was chosen because it provides IPv4 and IPv6 support with the uIP TCP/IP and uIPv6 stacks. According to IoT-LAB, this uIPv6 stack was contributed by Cisco [Cisco Systems, 2018]. Also, the support to low-power and short-range technologies is provided, concretely to CoAP, RPL and 6LoWPAN. The usage of multi-threading and event-driven programming is made using protothreads. The programming language used in the operating system is the C language [Ritchie, 1993]. According to IoT-LAB, the Contiki has a particular focus on low-power wireless Internet of Things devices. Finally, the Contiki OS is one of the operating systems with more tutorials at the presented platform.

The main tutorials related to Contiki and the low-power and short-range technologies are "Private IPv6/6LoWPAN/RPL network with M3 nodes" and "CoAP server with public IPv6/RPL/6LoWPAN network". At this moment, the used version of the Contiki is the 3.x. The platform managers are integrating the Contiki-NG, a fork of the previous OS.

3.4.3 Components

As previously presented, the architecture is composed of various modules. This subsection provides what is needed for these elements, what may be used, and what may be modified. The component that will be developed is the Controller, and this module will use the others.

The network module

As previously presented, Contiki has support to low-power and short-range technologies described. These technologies are located at the directory /core/net. The previous tutorials use this directory to perform communication between a client and a server. As said, the module can be used to perform the communication between applications and the controllers, taking into account the proposed standards.

The IDS module

This module might be internal or external. Internal means that it will be at the same node, whereas external indicates that it will be running at an external device. There is a considerable diversity regarding IDS tools. A possibility is using the Snort [Snort, 2019]. The master student had contact with this tool regarding attended courses. That is why the implement is a possibility, and it is open-source. Also, this tool can send messages regarding the occurrence of attacks.

The battery module

As previously said, the M3 nodes do not use batteries, even if they support them. That is because they are plugged to the power grid. It might be possible to have a small set of nodes with batteries after speaking with the platform administrators. Regarding this module, we can use the energest module or the powertrace [Dunkels et al., 2011] in Contiki. The first one is located at the directory /core/sys/ and the second is in directory /apps/powertrace/. The powertrace reports the estimated power and resource utilisation of a network node.

The Security Association Database module

This module will have all security associations regarding each application. Each application will have its association list file with each specific format. Initially, when an application initiates and if its file exists, it will be loaded. Otherwise, it will be created as appropriate. This component does not exist, so its creation is necessary. Each file can be created with the application name beside a unique identifier. With these properties, an application cannot gain access to other data besides its.

The Security Algorithms module

This component will have all the algorithms needed to satisfied the requirements of applications. Bearing in mind these needs, presented previously, the module will have encryption and hashing algorithms. All of these practices are implemented by software to ensure portability, and so they can run on more devices. These algorithms set are composed by the AES, the 3DES, the Secure Hash Algorithm 1 (SHA1) and the SHA2. These set of practices will be integrated at the Contiki OS and not developed.

AES is one of the most suitable algorithms used in WSN, especially by hardware, and it was acknowledged at subsection 2.1.2. It uses key sizes of 128, 192 and 256 bits [Dener, 2014], whereas the 3DES uses a key length of 168, 112 and 56 bits. The block size used by 3DES is 64 bits, whereas the AES uses 128 bits. The variation of Data Encryption Standard (DES) performs 48 rounds, whereas the AES makes rounds depending on the key size. The last one can operate at various operation modes.

Regarding hashing algorithms, the SHA1 creates a hash with 160 bits. The remaining algorithms produce hashes with 256, 384 and 512 bits. The work [Nunoo-Mensah et al., 2015] presents a comparison and analysis of these algorithms regarding the WSN.

3.4.4 Metrics

Measurements will be obtained bearing in mind the predefined security and functional profiles of the applications, as well as, the defined attacks against applications and devices. The main objective is to allow security adaptation at run-time, taking into account a given context, using low-power and short-range technologies. This objective implies the following metrics:

- The energy consumption of the previous security algorithms must be performed. With this energy cost, they can be ranked and then they can be used in a given context. Then, it will be possible to use each of them at a particular moment, bearing in mind the energy cost and the security level provided by each of them.
- The system must resist attacks. As presented, the system will ensure protection against external attacks. These attacks can be simulated, injecting them in the network.
- The communication between applications and devices must be performed securely, using the security algorithms presented previously. The starting point to achieve this metric may be the adaptation of the second tutorial introduced. Then, the IoT-LAB provides the possibility of creating profiles to capture and analyse radio communications during the performed experiments.
- The energy cost of a given node must be measured. This measure must be performed with static security and adaptable security. The aim is to verify the gains or not regarding the two ways of achieving security.
- The system must change the security algorithm in use, taking into account the security needs of applications and context. As previously presented, the requirements of an application will create a security profile. However, if the implementation chose static security, this profile will not change. Otherwise, it can adapt to a better one taking into account the actual context.

- The computational effort of the mentioned security algorithms must be quantified. A complete analysis regarding these algorithms can be performed at M3 nodes. Some works were accomplished at different architectures but not in these nodes. The memory and the CPU utilisation can be evaluated for each algorithm.

3.5 Summary

This chapter presented the objectives of the work to be developed. Regarding these objectives, functional and security requirements were discovered and presented. Also, these points are related to the given discussion of the previous chapter. Then, an architecture was designed, exhibited and debated, besides the measures to be implemented.

In the next chapter, a set of experiments, performed at IoT-LAB, is presented beside a proposal and formalisation of the controller module.

This page is intentionally left blank.

Chapter 4

Controller Proposal

In this chapter, a controller proposal and one formalisation are presented. Also, a security analysis, regarding the energy consumption of the introduced security algorithms, is made. This analysis was conducted as prior work to be used in the next chapter. Also, a description of the performed experiments is provided, besides an automation process to carry out them.

4.1 Security algorithms analysis

In this section, the security algorithms analysed are presented in an orderly manner by energy cost. Experiments, performed at IoT-LAB, were prepared and conducted to create this list. These were done as prior work to have a real value regarding each energy cost of each algorithm to use them with the controller.

4.1.1 Security algorithms

The security algorithms analysed were presented previously. It was researched the implementations of these algorithms. The code used for AES was from [GitHub-kokke, 2019], whereas to 3DES, the code used was from [Hall, 2013]. To SHA1, the used code was from [Packetizer, 2019], and to SHA2 the code [Aaron Gifford, 2012] was utilised. All implementations of the algorithms are in language C [Ritchie, 1993].

The previous implementations, in particular, the encryption ones, allow the usage of various key sizes. Regarding AES, the algorithm was used with 128, 192 and 256 bits key length. The 3DES was utilised with 168, 112 and 56 bits key size. All three used keys were different to ensure 168 bits, and when they were equal, the algorithm assured 56 bits of security. The AES was used just in Electronic Codebook (ECB) mode as using other ways would increase the experiments combinations. This mode is the most basic form of block cypher encryption, but other methods are possible to use. Finally, the separation between encryption and decryption actions was performed.

This set of algorithms will compose the security profiles to be used by each application. The aim is to calculate the energy cost of each profile. Then, an analytic validation may be made to validate the controller proposal.

4.1.2 Stipulated variables

For all experiments, the following parameters were fixed and used:

- The size of the used word. This string size was set with a length of 16 Bytes. The encryption algorithms encrypted this string, whereas the hashing ones calculated its digest.
- The time of each experiment. This time was fixed at 10 minutes, and it was established, bearing in mind the previous referred works.
- The number of runs. Each experiment was repeated 31 times to take scientific results.
- The same monitor profile. This profile was created to gather data collection regarding energy consumption. The sampling period chosen to perform this data collection was of 1.12 milliseconds. Upper values can be selected, but using them means less precision.
- The same node. All experiments ran on the same node, at the Lyon site. The used M3 device was M3-12.

4.1.3 A simple application

The security algorithms, previously presented, were used in a simple Conkiti implementation. The implementation, localised at the directory contiki/examples/hello-world/ was modified to carry out the integration of these algorithms. At first, the application was changed 4.1 to print out a message every two seconds. Then, the previous algorithms were used with this string.

Listing 4.1: A Contiki simple application

```
#include "contiki.h"
#include <stdio.h> /* For printf() */
#define TIMEOUT 2

PROCESS(hello_world_process, "Hello world process");
AUTOSTART_PROCESSES(&hello_world_process);
PROCESS_THREAD(hello_world_process, ev, data)
{
  PROCESS_BEGIN();
  /*variables are declared static to ensure their values are kept between kernal calls*/
  static struct etimer timer;
  const static uint8_t msg[] = "hello world 1234";//"PINg Hello: 1234";

  /*set the etimer module to generate an event in two second*/
  etimer_set(&timer, CLOCK_CONF_SECOND * TIMEOUT);
  while(1)
  {
    /*wait here for an event to happen*/
    PROCESS_WAIT_EVENT();

    /*if the event is the timer event as expected....*/
    if (ev == PROCESS_EVENT_TIMER)
    {
      printf("%s\n",msg);

```

```
        /*reset the timer to generate another event*/
        etimer_reset(&timer);
      }
    }
    PROCESS_END();
}
```

A firmware was created based on this application for each configuration. Regarding the number of settings, sixteen firmwares were built. Twelve relatively to AES and 3DES, taking into account the size of the keys, whereas the remaining ones were to the Secure Hash Algorithm (SHA) algorithms. Also, a firmware was created to the initial implementation 4.1.

Each version of the previous application was compiled in a Linux machine with the following properties:

- CPU: Intel(R) Core(TM) i5-6400 CPU @ 2.70GHz
- Operating System: Linux SparkyMultiDeb 4.19.0-1-amd64 #1 SMP Debian 4.19.12-1 (2018-12-22) x86_64 GNU/Linux
- Memory RAM: 8GB - 2 DIMs of 4GB with speed of 2133 MT/s and manufactured by Kingston
- Disk: TOSHIBA-TL100 with 223,6 GiB
- GNU Compiler Collection (GCC) Compiler: version 7.3.1 20180622 [ARM/embedded-7-branch revision 261907]

Then, each firmware version was upload to IoT-LAB, to M3-12 device presented previously, to perform experiments.

4.1.4 An automation process to perform experiments

In previous subsections, the compiled firmwares were presented. Each one of them was organised into folders. Then, a bash script was created to carry out the thirty-one runs. This script is exhibited in the appendix chapter, at appendix E 6.3. At first, it enters at each configuration folder. Then, it takes the firmware and the needed configurations to launch an experiment. Each experiment is launched 31 times, and for each time, the script waits to the end of it. Finally, the bash script gets the energy file created by each run of each experiment and puts it at the configuration directory.

4.1.5 Results

As presented previously, the used monitor profile was employed with 1.12 milliseconds. This sample period is the one that provides more precision regarding the data collection. The data collected was the current, the voltage and the power, besides the timestamps. The image 4.2 presents an example of the files gathered. This power, as well as the other variables, was the instantaneous power at each moment collected.

The total energy consumption of each experiment was calculated with Python functions presented in appendix D, at 6.2. These functions were developed, bearing in mind the code shown at 6.1, from GitHub [Florian Kauer, 2018].

Listing 4.2: An example of a gathered file with energy data

```
protocol: 5
domain: 172545
start—time: 1560369566
sender—id: m3_12
app—name: control_node_measures
schema: 0 _experiment_metadata subject:string key:string value: string
schema: 1 control_node_measures_consumption timestamp_s:uint32 timestamp_us:uint32 power:
      double voltage:double current:double
content: text

12.742764 1 1 1560369578  720631  0.126046  3.256250  0.038688
12.743252 1 2 1560369578  721729  0.128244  3.257500  0.039343
12.743374 1 3 1560369578  722858  0.125924  3.253750  0.038683
12.743618 1 4 1560369578  723957  0.128366  3.253750  0.039450
12.743710 1 5 1560369578  725086  0.125679  3.255000  0.038620
12.743832 1 6 1560369578  726184  0.128122  3.256250  0.039362
12.743954 1 7 1560369578  727313  0.125924  3.253750  0.038669
12.744046 1 8 1560369578  728412  0.128122  3.256250  0.039353
12.744137 1 9 1560369578  729541  0.125924  3.252500  0.038747
 ...
```

From each experimental time, 30 seconds were taken. This time was removed from the beginning. This decision was made because, in the beginning, a node starts the services and other components. Figure 4.1 presents the instantaneous power of a experiment performed. We can see the idea described in the first time elements displayed in the plot.

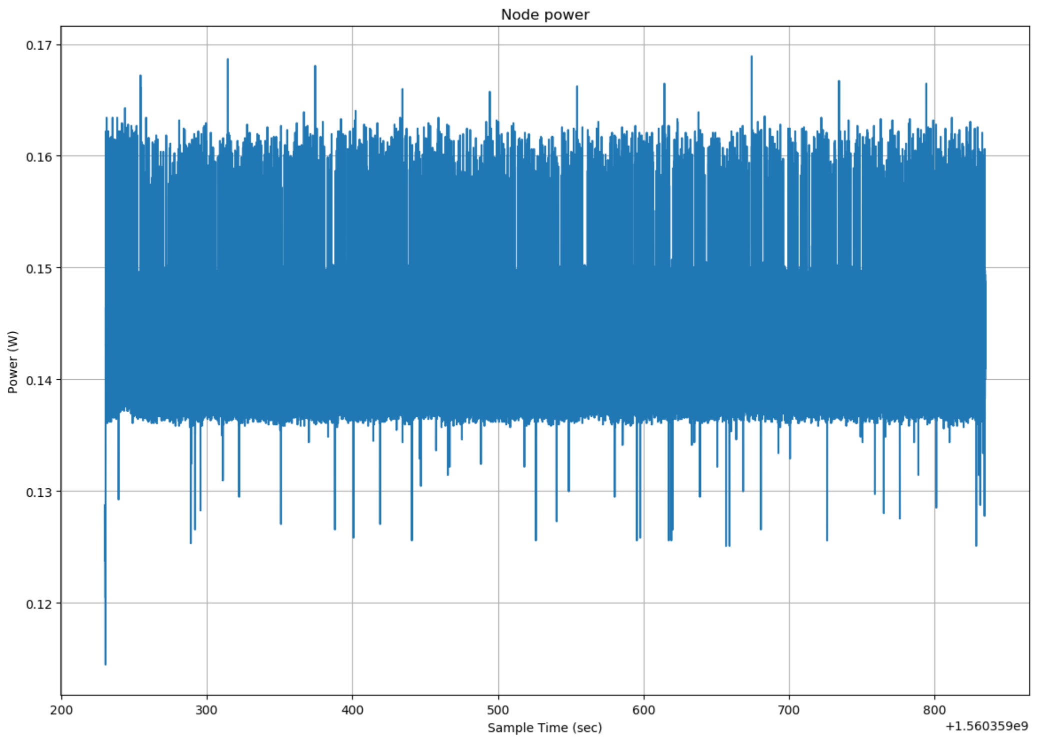

Figure 4.1: The values of the instantaneous power from an experiment

However, an analysis of the data gathered from the experiments was performed with and without the cutting explained previously. As said, the data analysis was made using the previous Python code. Table 6.2 presents the results regarding the performed cutting, whereas table 6.1 shows the achievements with respect to the opposite. These two tables are presented in the appendix sections, concretely at appendix F.

Table 4.1: The values of the energy cost regarding operations

Configurations\Measurements	Total energy cost (J)	Operation energy cost (J)	CoAP request (J)
AES 128 ENCRIPT	1.251	0.004	0.007
AES 128 DECRYPT	0.761	0.003	
AES 192 ENCRYPT	0.954	0.003	0.006
AES 192 DECRYPT	0.786	0.003	
AES 256 ENCRYPT	1.055	0.004	0.006
AES 256 DECRYPT	0.736	0.003	
3DES 56 encrypt	0.275	0.001	0.004
3DES 56 decrypt	0.813	0.003	
3DES 112 encrypt	0.464	0.002	0.004
3DES 112 decrypt	0.774	0.003	
3DES 168 encrypt	0.235	0.001	0.004
3des 168 decrypt	0.790	0.003	
SHA1	0.670	0.002	0.005
SHA2 256	1.106	0.004	0.008
SHA2 384	1.118	0.004	0.008
SHA2 512	2.345	0.008	0.016

Table 4.1 presents the energy cost of each security algorithm, and it was created using table 6.2 with sample cutting, presented at appendix D. This table was created performing the subtraction of each configuration with the initial setup. Then, the energy cost of each operation was calculated. This energy cost is presented by column "Operation energy cost". The total operations performed in each experiment were 285. This value was achieved, dividing 570 seconds by 2 seconds. The first value is regarding the experiment time, whereas the 2 seconds is related to timer timeout. The last column is regarding a CoAP request. One demand and one response compose this request. So, the final value was calculated performing the sum of the energy cost regarding the encryption and decryption operation. In this case, for hashing operations, the energy cost was multiplied by two.

4.2 Controller proposal

In this section, a proposal to the creation of the controller module is presented. An analysis regarding input, internal and output variables was performed. These variables are required to implement security adaptation at real-time.

4.2.1 Input variables

These variables will inform the controller regarding the requirements of the applications, the battery level and the network status as presented by the previous architecture. The needs of the implementations more relevant to the controller are the ones related to security.

As previously introduced, each application might require different security requirements, such as integrity or authentication. Also, they can opt not to use security. Another aspect related to these needs is the level of protection. Again, an application can choose to use standard or high-security requirements. These two aspects result in a set of security algorithms. This set is the initial security profile for each application. Besides these relevant aspects, an application might choose not to change this initial profile. Also, each application, in case of the change the initial set of algorithms, can provide weight to the selected security. This weight can tell if the selected security is essential or not.

The controller must have into account the battery level of a node. If an application provided a weight saying that security is not essential, the algorithm must take actions to minimise energy consumption.

The controller must apply the same behaviour when an IDS informs the occurrence of attacks. In this case, the algorithm will adapt security to high levels and will not look up to the energy cost. When an attack terminates, the controller can downgrade security.

4.2.2 Internal variables

The internal variables to be handled by the algorithm are mainly security algorithms presented. As previously said, the requirements of the applications will result in initial security profiles. Also, these security needs are confidentiality, integrity, authentication and non-repudiation, and they were presented previously. With this information, initial security profiles can be created and used by implementations.

Table 4.2: The robustness level of security algorithms

Algorithms	**Security type**
SHA2-512 SHA2-384 SHA2-256 SHA1	Hashing
AES-256 AES-192 AES-128 3DES	Encryption

In table 4.3, the mentioned profiles are presented. As we can see, the security profiles correspond to the combinations between standard or high security. When an application chooses P1, it represents that this implementation does not need protection. In contrast to P1, if an application wants the P9, it means that this implementation needs high security. The security configuration, to ensure these profiles, is presented in grid 4.4.

Table 4.3: The security level combination of security profiles

Profiles\Security needs	Confidentiality	Integrity/Authentication
P9	high	high
P8	high	standard
P7	standard	high
P6	standard	standard
P5	high	-
P4	standard	-
P3	-	high
P2	-	standard
P1	-	-

Table 4.4 was created, bearing in mind grid 4.2. This table presents all the security algorithms ordered by descending order regarding the given security. As shown, integrity and authentication were added in the same column. With this merge, the combinations were reduced. As previously mentioned, authentication can be performed encrypting the created digest by the application of the hash algorithm. The only requirement is that the encryption key must be known only by the sender and the receiver. Also, all the algorithms to respect these profiles must be in memory to allow security change.

Table 4.4: The set of security algorithms to fulfil static security profiles

Profiles	Security configuration
P9	AES-256, SHA2-512
P8	AES-256, SHA2-256
P7	AES-128, SHA2-512
P6	AES-128, SHA2-256
P5	AES-256
P4	AES-128
P3	SHA2-512
P2	SHA2-256
P1	-

With table 4.1, presented in the previous subsection, the security profiles might be ordered by energy cost. Each security profile is composed of the security algorithms analysed already. If the sum is performed regarding the operation of each algorithm, used in a given security profile, we have the energy cost of this profile. This reason was made, and table 4.5 was created.

In other words, table 4.5 presents the energy cost of a CoAP request when an application uses a given security profile. The values displayed in this table were rounded to three decimal digits. The presented tables in appendices sections were not rounded. The energy cost shown is regarding just to security operations and not to other costs.

The security profiles presented are not perfect regarding security. As mentioned, the focus is regarding external attacks and not internal attacks. That is why it was presented algorithms to perform encryption and hashing. Another limitation is regarding the use

Table 4.5: Security profiles ordered by energy consumption

Security profiles	Energy cost (J)
7	0.024
9	0.023
3	0.016
6	0.015
8	0.014
2	0.008
4	0.007
5	0.006
1	–

of symmetric encryption. If the same key is used several times, it is possible to identify patterns regarding the data transmitted. With the possibility of to not use security in an application, it can be conducted all types of attacks. Also, if an attacker has access to a given node, he might get access to the used keys.

4.2.3 Output variables

These types of variables are the final result of the usage of the controller. The security algorithms in use, in a given moment, compose this set. Also, this combination is the most suitable, given a context at the time.

4.3 Controller formalisation

In this section, the formalisation of the controller is presented. This formalisation embraces input, output and internal representations regarding the variables identified. Then, an algorithm is presented, where is covered the relations between the variables.

4.3.1 Input variables

Application security requirements

An application can choose various security requirements regarding the security of communications. This set of needs is composed of Confidentiality (C), Integrity (I) or Authentication (A). Confidentiality is achieved by performing the encryption of messages. On the other hand, integrity is fulfilled, performing the hash of a message. Another critical need is Non-repudiation (NR) and is achieved by creating the hash of a given message and then encrypting this hash. The same applies to perform authentication. These properties are possible to reach because the key used to perform encryption is only meet by the sender and receiver. After the selection of the security requirements, a level (L) can be specified. This level can assume one of two possible values, concretely, value 1 or 2 regarding the standard or strong security. C, I, A and NR can have the numbers 0 or 1, regarding True or False.

$$Sec = [C, I, A, NR];$$
$$L \in [1, 2].$$

Another critical variable is called Val and represents the weight given by an application regarding adaptive security. This variable assumes values between 0 and 1. The value 0 represents that security is not essential, whereas the value 1 means otherwise. In a complementary way, an application chooses:

$$decision = \begin{cases} Val \in [0, ..., 1] & \text{regarding to security;} \\ (1 - Val) & \text{regarding to energy.} \end{cases}$$

In an automated manner. Then, a comprehensive formula is created:

$$(1 - Val) * energy + Val * security \tag{4.1}$$

The previous formula shows the controller logic regarding the security adaptation to be performed. When security is necessary, the energy consumption is less critical. Contrarily, when energy is more important, the formula portion regarding protection is smaller. This controller intelligence is the starting point to make the change of a given security profile.

Battery level

The battery level of a node can have values between 0 and 1. These values represent 0 or 100% regarding the battery load. This level can be partitioned into two sub-levels, concretely battery level 1 (BL1) and battery level 2 (BL2):

$$BL = \begin{cases} BL1 \in [0, \ldots, 0.5[& ; \\ BL2 \in [0.5, \ldots, 1] & . \end{cases}$$

Network status

An array with size n represents a set of network attacks equals to 0 or higher than 0. Each attack is associated with a severity degree. This degree is related to the application profile, and for this reason, the same attack can have different severity degrees regarding the profile of the application. Finally, each attack has associated a type. This type is related to attack mitigation. Attacks mitigated by encryption have type E, whereas the ones handled by hash have type H. Attacks with type A are the ones that are covered by authentication. In a nutshell, a set of attacks is represented by:

$$AT = [At1, \ldots, Atn], \text{ with a severity degree}$$
$$SAt = [SAt1, \ldots, SAtn], \text{and a type}$$
$$T = [E, H, H, A, \ldots].$$

The size of each array is equal for all. Regarding the type of the attacks, E == 1, H == 2 and A == 3. The severity level of each attack takes the following values: severe == 3, critical == 2 and normal == 1. The vector of attacks without the severity degree and the type has binary values.

4.3.2 Internal variables

Encryption algorithms

In consideration of the chosen encryption algorithms, the Enc array represents these algorithms. Each algorithm has associated a security level (SecL1) and an energy consumption level (Econ1). The Enc array has the values regarding the AES-128, AES-192, AES-256, and 3DES, whereas the SecL1 and Econ1 have the security level and the energy consumption level of each algorithm. The values of these levels arrays vary between 0 and 1, and the sum of each level collection is equal to 1. Once more, 0 means insecure or less energy consumption, whereas 1 means more secure and more energy consumption. Briefly, the following formulation can be applied:

$$Enc = [AES128, AES192, AES256, 3DES],$$

where each array position takes values 0 or 1;

$$SecL1 = [y1, \ldots, y4], \text{ and } Econ1 = [x1, \ldots, x4],$$

where all the arrays have the same length.

In a nutshell, an encryption algorithm is characterised by:

$$Enc * SecL1 * Econ1. \tag{4.2}$$

Hash algorithms

A new array called Hash is used to represent the hash algorithms SHA1, SHA2-256, SHA2-384, and SHA2-512. Like to encryption algorithms, for each hash algorithm is associated with a security level (SecL2) and an energy consumption level (Econ2). Once again, the values of these levels arrays vary between 0 and 1. The sum of each collection is equal to 1. The value 0 represents insecure, or less energy consumption and 1 means more secure and more energy consumption. The following formulation can be applied:

$$Hash = [SHA1, SHA2-256, SHA2-384, SHA2-512],$$

where each array position takes values 0 or 1;

$$SecL2 = [i1, \ldots, i4], \text{ and } Econ2 = [j1, \ldots, j4],$$

where all the arrays have the same length.

So, a hash algorithm is characterised by:

$$Hash * SecL2 * Econ2. \tag{4.3}$$

Static security profiles

To initiate the controller, opening static security profiles were established, taking into consideration the application security requirements, and the security configuration to fulfil these requirements. These security profiles were presented in subsection 4.2.2. A matrix represents these static security profiles with nine lines and three columns. The columns describe the security requirements with security configurations (algorithms), and the lines describe all combinations regarding the security levels presented earlier. Integrity and authentication were gathered together. The only difference between these two properties is related to encryption of the digests. Also, with this merger, the combination of states was reduced. One formulation to describe this table may be the following:

$$SPinit = [SP[i] = (Sec * L) \rightarrow (Enc * SecL1 * Econ1 + Hash * SecL2 * Econ2)],$$

where i=1 and less or equal to 9.

4.3.3 Output variables

The controller output is the creation of a Security Profile (SP) related to a given context specified with the input variables. This security profile is composed of the set of algorithms regarding the performed encryption and or hash. So, for each SP, we have a different combination regarding these algorithms along the time. Each SP has a total energy consumption level created by the sum of all energy consumption levels of the algorithms used.

4.3.4 Relations between variables and an algorithm

The primary objective is the creation and the use of a SP suitable in a given context. We aim to provide appropriate security, and at the same time, extend the battery load of a node. With this objective in mind, the first logic idea is to relate the battery level, in a given time, with an energy consumption of a SP. The following formulation can be used:

$$NtSP = BL(t)/SP, \tag{4.4}$$

where NtSP means the Number of times a SP can be used.

This formulation makes sense, because a security profile is composed of encryption and hash algorithms, and each one has associated an energy cost beyond a security level. With this formalisation, it is not taken into consideration other energy costs such as the communication ones. A security profile, at first is selected bearing in mind the application security requirements. Then, if the application chosen adaptation regarding these requirements, the next security profile must be picked regarding the maximisation of the number of times a security profile can be used. Also, with the maximisation of this number, the energy consumption is minimised, because we are selecting the security profile with the lower energy consumption. Taking into account this reason and the previous variable description, a simple pseudo-code was conceived and is presented in 1.

As previously mentioned, in the beginning, a security profile is selected, taking into account the application security requirements. At line 2, the function named selection_init_security_profile returns the security profile regarding these requirements. This profile is the initial SP, and it will change or remain the same over the time, depending on the Val value as previously presented and the occurrence of an attack. That is why the SP is stored at variable SPfirst, at line 3. Then, in line 4, we have the security level of this SP. The WHILE cycle represents the core of the application running. Here, we have the initial SP and his security level, selected in consideration of the application security requirements. This SP is applied at line 6, and then the central process is blocked on line 7. At this line, we are waiting for the occurrence of an event. In line 9 and 11, an IF statement is presented. It has the purpose to identify an event between an attack or a timeout. Other events are not relevant, taking into account the algorithm context. The occurrence of an event defines the attack_status value, where 0 corresponds to a timeout event and 1 to an attack event. Then, the controller_function is called with the necessary parameters. This function returns the best suitable security profile to be applied at a given moment. This new security profile will be enforced, and it will be maintained, or a new one will be created when the occurrence of an event happens.

Algorithm 1 Selection of the optimal energy-aware security profile

```
1: procedure CONTROLLER_APPLICATION_FUNCTION($Sec, L, Val$)
2:     $SP \leftarrow$ SELECTION_INIT_SECURITY_PROFILE($Sec * L$)
3:     $SPfirst \leftarrow SP$
4:     $security\_level\_SPfirst \leftarrow$ GET_SECURITY_LEVEL_SP($SPfirst$)
5:     while 1 do
6:         Apply $SP$
7:         Process WAIT for event
8:         $BL \leftarrow$ READ_BATTERY_LEVEL()
9:         if $event = attack\_event$ then
10:            $attack\_status \leftarrow 1$
11:        else
12:            $attacks\_status \leftarrow 0$
13:        $SP \leftarrow$ CONTROLLER($Val, attack\_status, SP, BL, SPfirst, security\_level\_SPfirst$)
14:
15:
16:
17: procedure CONTROLLER($Val, attack\_status, SP, BL, SPfirst, security\_level\_SPfirst$)
18:     $NtSP \leftarrow$ GET_NUMBER_TIMES_SP($SP, BL$)
19:     $NtSPfirst \leftarrow$ GET_NUMBER_TIMES_SP($SPfirst, BL$)
20:     if $attack\_status \neq 1$ then                ▷ behaviour when attacks do not occur.
21:         if $Val > 0.5$ then                        ▷ the SP downgrade.
22:             if $security\_level\_SP > security\_level\_SPfirst$ then
23:                 $SP \leftarrow SPfirst$
24:         else              ▷ The security profile will be downgrade below the initial SP.
25:             $NtSP\_previous \leftarrow NtSP$
26:             $SP \leftarrow SP - (1 - Val)$
27:             $NtSP \leftarrow$ GET_NUMBER_TIMES_SP($SP, BL$)
28:             if $SP < minimum\_number\_SP$ then
29:                 $SP \leftarrow minimum\_number\_SP$
30:             else
31:                 if $NtSP < NtSP\_previous$ then
32:                     $SP \leftarrow SP + (1 - Val)$   ▷ If the previous security profile has a lesser energy consumption, we stay with him
33:     else                                           ▷ behaviour when attacks occur.
34:         $SAt \leftarrow$ GET_ATTACK_PROPERTIES()
35:         if $security\_level\_SP < security\_level\_SPfirst$ then
36:             $SP \leftarrow SPfirst$
37:         else
38:             $security\_level\_SP\_previous \leftarrow security\_level\_SP$
39:             $SP \leftarrow SP + (Val * SAt)$           ▷ The update of the security profile.
40:             $security\_level\_SP \leftarrow$ GET_SECURITY_LEVEL_SP($SP$)
41:             if $SP > maximum\_number\_SP$ then
42:                 $SP \leftarrow maximum\_number\_SP$
43:             else
44:                 if $security\_level\_SP < security\_level\_SP\_previous$ then
45:                     $SP \leftarrow SP - (Val * SAt)$   ▷ If the previous security profile has a higher security level, we stay with him.
46:     return $SP$
```

The function controller is the algorithm core. The parameters of this function are Val, attack_status, SP, BL, SPfirst and security_level_SPfirst. Val represents the security weight, whereas the attack_status says if an attack occurs or not. The SP is the security profile composed by encryption and hash algorithms, and BL is the battery level at a given moment. The SPfirst is the initial security profile, and finally, the last parameter provides the security level of the initial profile. The first action to be performed at this function is the calculation of the number of times a given SP (NtSP) can be used. That is why at line 18 and 19, the function get_number_times_SP is called with the parameters SP and battery level. At this point, we have the actual SP and the first SP NtSP. To have two distinct behaviours when an attack occurs or not, the first IF statement at lines 20 and 33 was created, to split this logic. At the second part we have the algorithm logic when an attack occurs and at first part the opposite.

To conserve the node energy, when an attack does not occur, we must take into consideration the weight given to security. When security is fundamental, in an application context, the algorithm will decrease the SP, but it will not reduce below the first security profile. Here, we are taking into account only the security level of each SP. When Val is equal or less than 50%, the algorithm will decrease below the initial SP. At this situation, we are taking into account the energy consumption of each SP besides the security level, since the algorithm logic does not jump security profiles. The decrease below the initial SP is performed, taking into consideration the $SP-(1-Val)$ formula. This reason is expressing between lines 24 and 32.

When an attack occurs, a different logic is presented between lines 33 and 45. In this situation, we will not take into consideration the energy consumption of each security profile directly, but only the security level. If we have an SP with less security level than the first SP chosen in the beginning, the new SP will be the first security profile (line 40). Otherwise, we must improve security, upgrading the actual SP. This upgrade takes into account the weight given to security, besides the severity of the attack. It is wise to take into consideration these two parameters because we can safeguard resources and use them in a recommended way. If I can mitigate an attack with level two using an SP number n, I will not use an SP n+1 that consumes more energy than the previous profile and has a high-security level. All this reason is expressed with the $SP+(Val*Sat)$ present at line 39.

Regarding a given context, the algorithm presented upgrades or downgrades a security profile. It takes into consideration the security requirements chosen by an application besides the importance of security, the network status, and the energy consumption of each security profile. Also, the battery level is taken into account, even if in a direct or indirect form.

4.4 Summary

In this chapter, a set of experiments were performed to analyse the energy cost of the security algorithms involved. Then, a controller proposal was presented besides its formalisation. At this point was introduced and discussed the all involved variables. Then, an algorithm was created and exhibited to handle these variables.

The next chapter presents an analytical validation of the proposed algorithm and architecture, using the energy consumption results, collected experimentally. Also, more specifications were carried out, taking into consideration an energy cost simulation. This energy simulation begins from the real measurements performed in this chapter.

This page is intentionally left blank.

Chapter 5

Analytical validation and results

In this chapter is validated the proposed controller, presented previously. The analytical validation of the algorithm is performed, using various scenarios and the Python language [Van Rossum et al., 2007]. These scenarios have several attacks simulated with distinct severities. Then, an additional analysis is performed defining applications and specific attacks with context and severities. Also, for each implementation, an initial security profile is provided. The chapter is concluded with the simulation of the previous arrangements concretely, the energy cost, the number of messages and the node lifetime. Then, a summary is presented.

5.1 Analytical validation

The algorithm, presented in chapter 4, in section 4.2, is analysed and validated in this chapter. The pseudo-code was mapped into Python [Van Rossum et al., 2007] code and it is presented in image 5.1. Then, some scenarios were simulated to see the behaviour of the presented algorithm. Simulated scenarios just with the occurrence of attacks and with severity levels of one, two and three were used. Also, the appearance of random attacks with different severity levels was taken into account. With this reason, the primary purpose is to verify the behaviour of the algorithm related to severity levels that an attack can have.

Listing 5.1: The controller algorithm in Python language

```
def controller_algorithm(attacks_status):
    attacks = attacks_status
    all_sp_values = []

  for i in range(len(val)): #see each val value/application
    sp = first_sp[i]
    sp_values = []
    sp_values.append(sp)
    for j in range(1,len(attacks)): #SIMULATE EVENTS DURING THE TIME... the first event is to stipulate first_sp
      if attacks[j] == 0: ############## NOT ATTACK
        if val[i] > 0.5: #if the val > 0.5 i will see only the security table
          # i will compare the actual sp and the first regarding to security and
          # downgrade sp in the case the actual sp is more secure
          if findIndex(round(sp),1) < findIndex(first_sp[i],1):
            sp = first_sp[i]
        else: #val has values <= to 0.5 -> implies that i will downgrade security and i will see only the energy table
          if findIndex(round(sp),0) < findIndex(first_sp[i],0) : #if the actual sp > first_sp regarding to energy consumption
            sp = first_sp[i]
          else: # otherwise
            # i will decrease the sp regarding to the formula 1-val: if val == 0.75 i will have 1-val == 0.25
            sp = sp - (1-val[i])
            if round(sp) < 1: #the minimal border number of SP's indices...
              sp = 1
            else:
              # if the actual sp consumes more energy than the previous -> get the previous
              if findIndex(round(sp),0) < findIndex(round(sp + (1-val[i])),0):
                sp = sp + (1-val[i])
      else: ############# ATTACK
        # i will see only the security table for all val values
        # return to the init SP in an occurrence of an attack
        if findIndex(round(sp),1) > findIndex(first_sp[i],1) :
          sp = first_sp[i]
        else:
          # i will upgrade the sp regarding to the attack severity and val
          #this is for new tests ...
          if type(attacks[j]) is str:
            severity = get_severity(i, attacks[j])
            update = val[i]*severity
          else:
            update = val[i]*attacks[j]
          sp = sp + update
          if round(sp) > 9: #the maximum border number of SP's...
            sp = 9
          else:
            # if the actual sp is less secure than the previous -> get the previous
            if findIndex(round(sp),1) > findIndex(round(sp - update),1):
              sp = sp - update
      sp_values.append(int(round(sp)))
    all_sp_values.append(sp_values)
  return all_sp_values
```

For all simulations, the number of security profiles taking into account was nine. These profiles were presented previously, and they are related to the security requirements discovered and analysed. Also, two tables were taken into consideration. The first one was the table with security profiles organised by security level. The second had the same set of algorithms but sorted by energy consumption. These two tables were arranged into decreasing order, and they were presented in chapter 4, specifically at tables 4.3 and 4.5. In a nutshell, the following table 5.1 shows the referenced tables joined.

Regarding table 5.1 presented, the cost provided is the sum of the energy cost of each security algorithm that composes each security profile. When a CoAP request is made, we have two messages, the demand and the response. The previous energy cost, presented of each profile is the cost regarding these two messages. Also, the algorithm behaviour was analysed in case the energy consumption table is equal to the security table. However, the most curious and essential situation is the one where the tables are organised differently. Finally, the first security profile considered the following analysis was the SP equal to 4.

Table 5.1: Security profiles ordered by security and energy consumption

Index	**Security**	**Energy consumption**	**Energy cost (J)**
0	9	7	0.024
1	8	9	0.023
2	7	3	0.016
3	6	6	0.015
4	5	8	0.014
5	4	2	0.008
6	3	4	0.007
7	2	5	0.006
8	1	1	–

5.1.1 Simulation just with attacks

The main idea of this group of attacks is to verify the algorithm behaviour when it occurs attacks. Taking into account the three severity levels that an attack can have, three arrays of offensives were considered and are presented at the next steps, to check the behaviour.

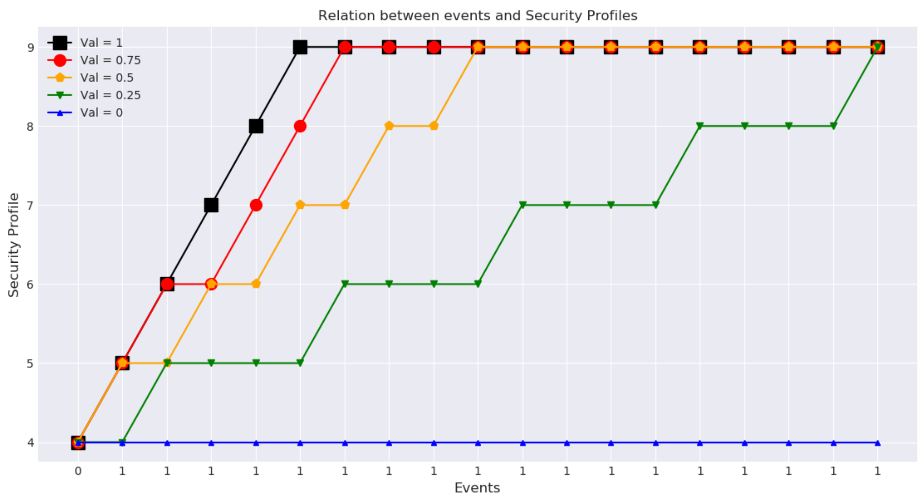

Figure 5.1: Random occurrence of attacks with severity equal to 1

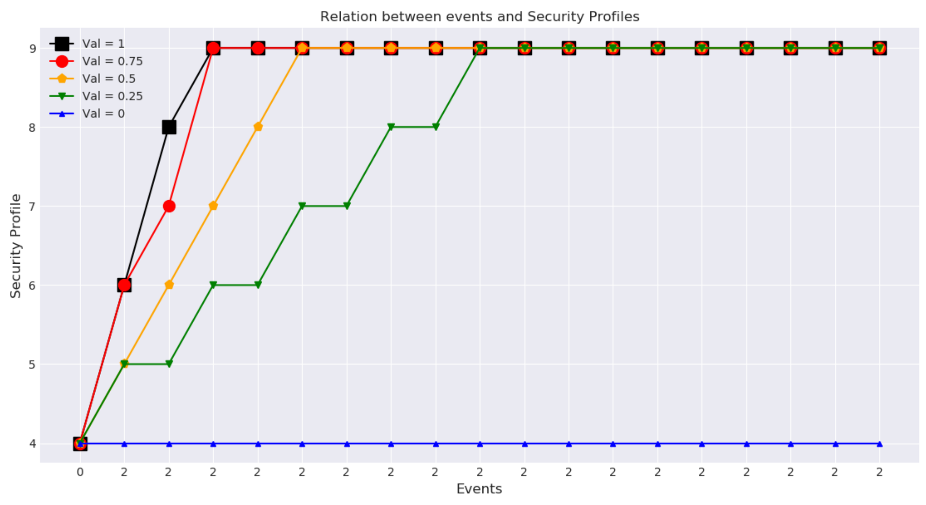

Figure 5.2: Random occurrence of attacks with severity equal to 2

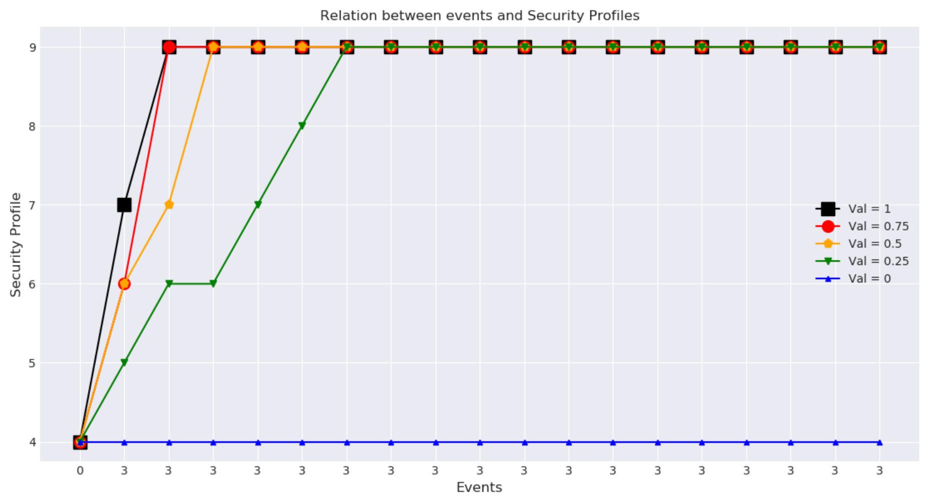

Figure 5.3: Random occurrence of attacks with severity equal to 3

The previous images, 5.1, 5.2, and 5.3, illustrate the algorithm behaves as expected. A security profile is updated, taking into account the severity of an attack. The main conclusion of this behaviour is the sharp increase of the SP due to the attack severity when the gravity is equal to 3. Another interesting point to take into account is the weight given to security into the final result.

5.1.2 Simulation without attacks

This vector of attacks seeks the analysis of the other side of the algorithm behaviour boundary. In the previous charts, we see the behaviour of the algorithm just when an attack occurs. Here, we seek evolution when do not occurs attacks.

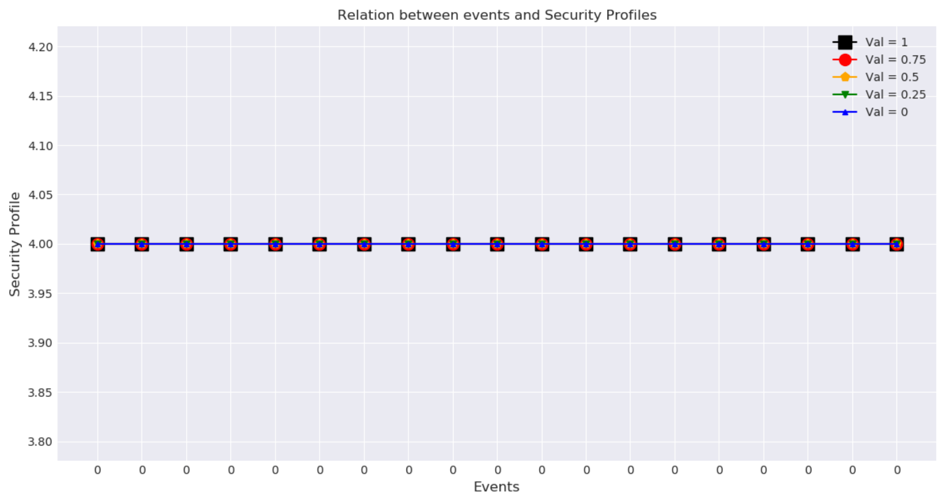

Figure 5.4: The security profile downgrade taking into account different tables

Once more, we have the expected behaviour. The algorithm decreases the security profile taking into consideration Val value. When Val is higher than 0.5, the SP remains the same over the time because the security is critical. In the other side, for inferior values, the security profile downgrades in consideration to $SP-(1-Val)$ formula. The previous image, 5.4, was created with different tables. The SP remains the same over the time because SP number 3 has a higher energy consumption cost than SP number 4. If we use the same tables, the final result is presented in figure 5.5.

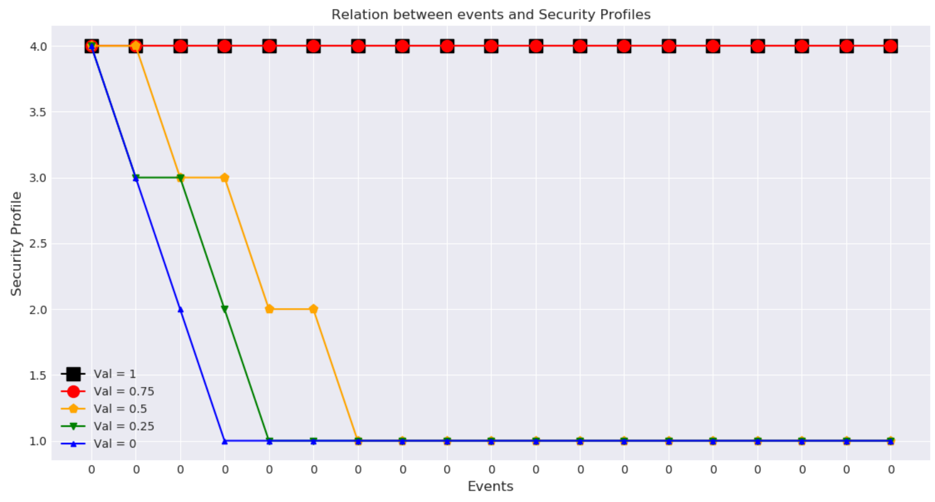

Figure 5.5: The security profile downgrade taking into account equal tables

5.1.3 The analysis of the Val values

The Val represents the weight given to security and is related to the security profile adjustment. It has a higher importance. The next images seek the demonstration of this importance related to the algorithm behaviour.

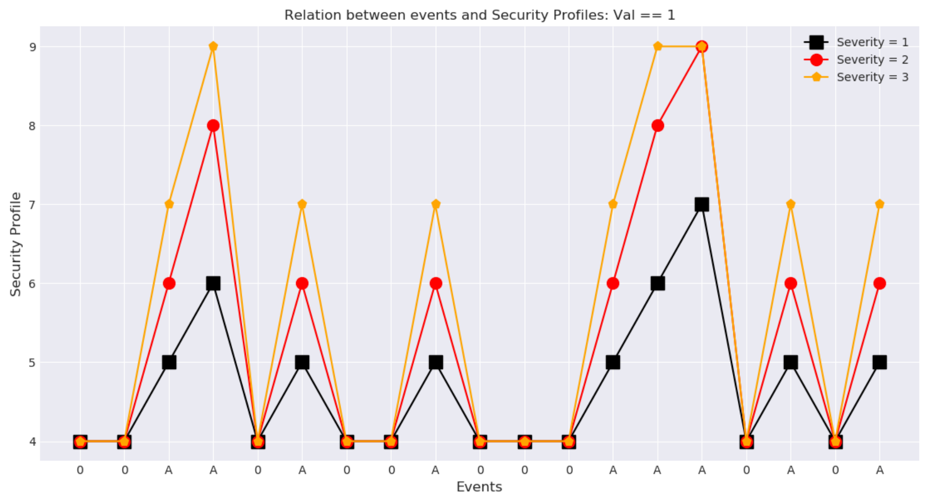

Figure 5.6: The value of each attack severity related to Val equal to 1.

In image 5.6, we see the upgrade of the security profile directly linked with the severity of each attack. It is normal behaviour because Val is equal to 1. For Val equal to 0.75, 0.5, 0.25 or 0, the SP update is performed not directly linked with the severity of the attacks. This reason occurs regarding $SP + (Val * SAt)$ formula.

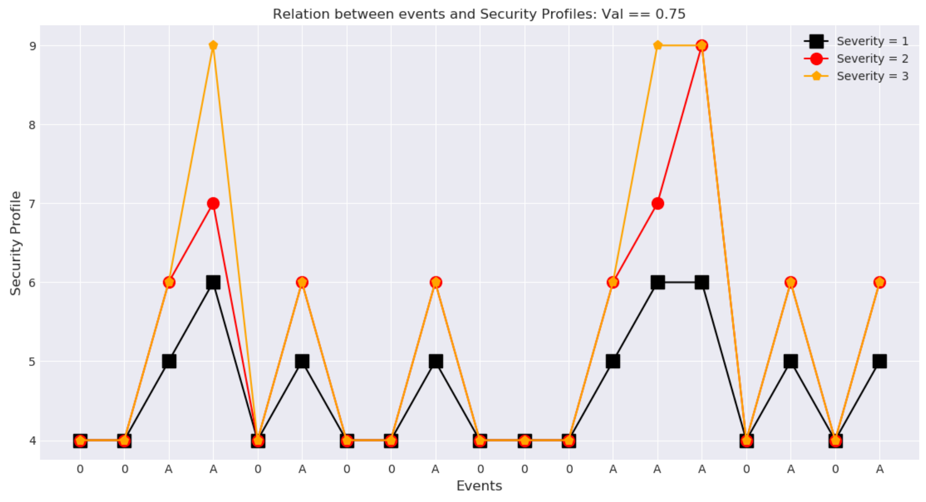

Figure 5.7: The value of each attack severity related to Val equal to 0.75

The calculations were performed with floating values, and then, it was conducted rounding of these values to carry out tables search to achieve a higher granularity.

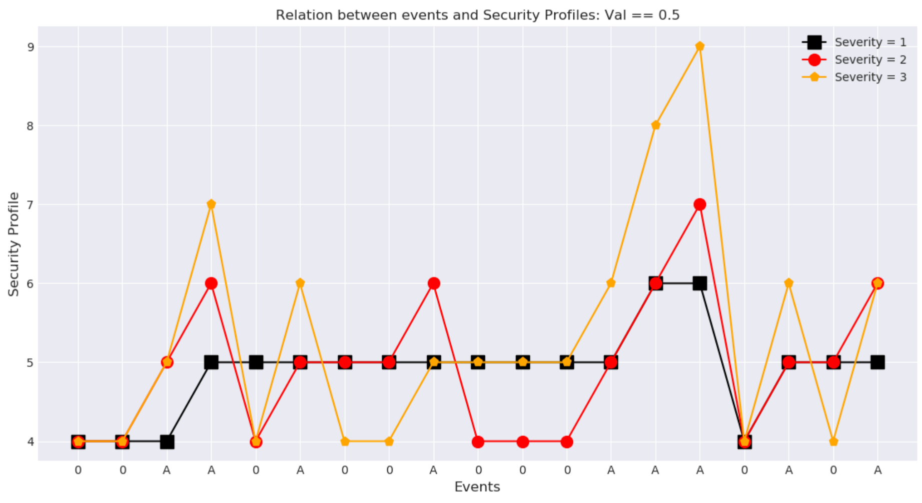

Figure 5.8: The value of each attack severity related to Val equal to 0.5

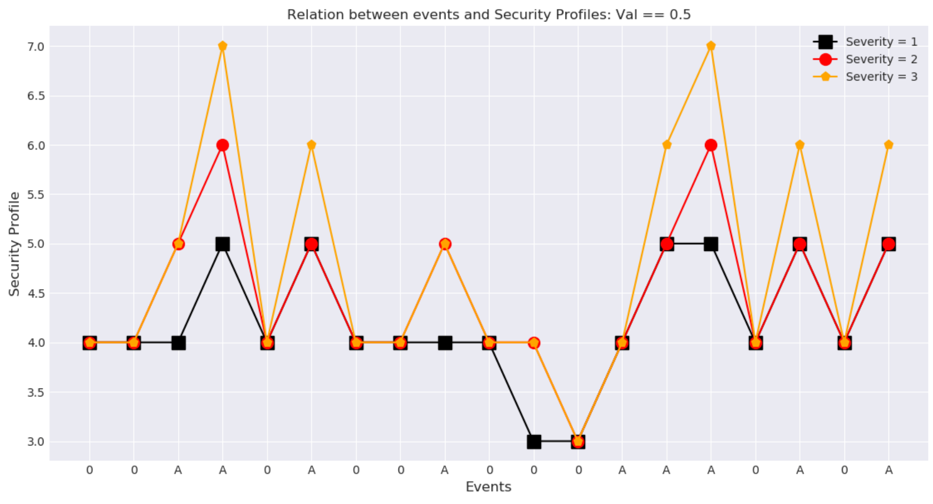

Figure 5.9: The value of each attack severity related to Val equal to 0.5 regarding identical tables.

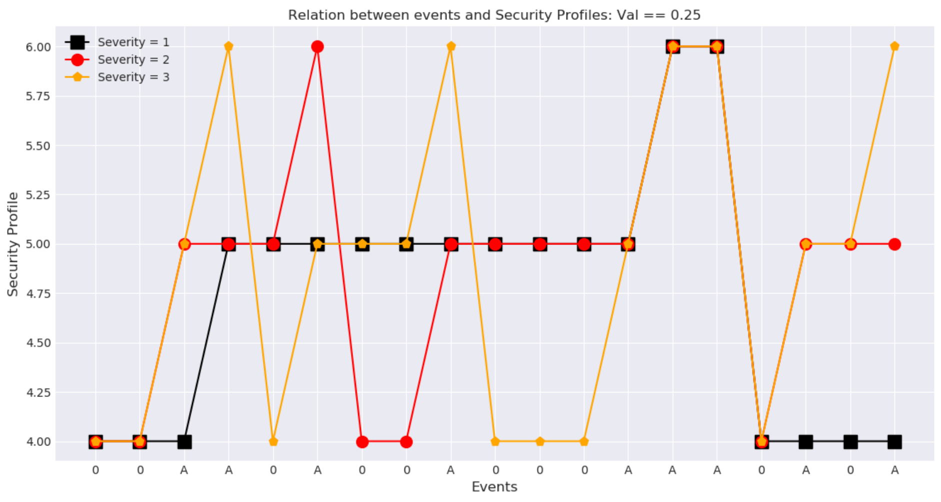

Figure 5.10: The value of each attack severity related to Val equal to 0.25

As said, for values of Val equals or lower than 0.5, the algorithm performs security profile downgrade under the first security profile. This situation is more visible in figures 5.9 and 5.11 when the severity of an attack is equal to 1, Val is equal to 0.5 and 0.25 when we have equals tables. In images 5.8 and 5.10, this behaviour is not visible because we are taking into account the different energy consumption table.

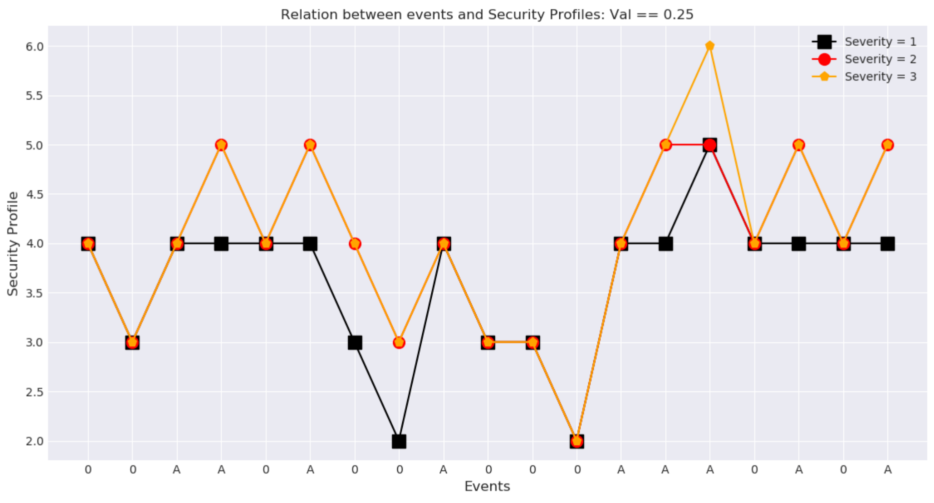

Figure 5.11: The value of each attack severity related to Val equal to 0.25 regarding identical tables

5.1.4 Simulation with random attacks

The following figures have all the values of Val represented for different attacks severity. The aim is to see the algorithm behaviour when, for the same attack severity, various Val values are applied.

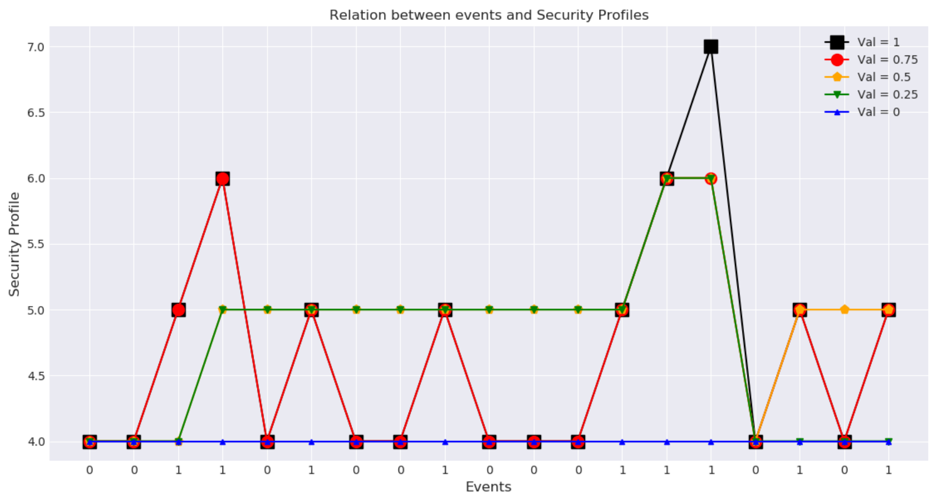

Figure 5.12: Random attacks with severity equal to 1

In figure 5.12, when Val is equal to 0.5, we can see that security profile equals to 4 or 5 are the ones with more utility. This situation occurs because, in the energy table, presented previously, these security profiles are the ones that consume less energy.

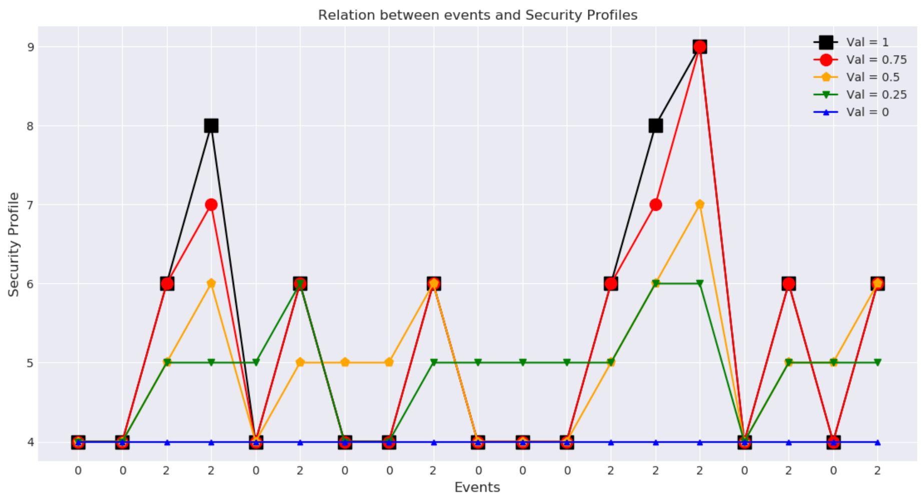

Figure 5.13: Random attacks with severity equal to 2

We can see the same behaviour in figures 5.13 and 5.14. The energy table is responsible for this pattern. The higher the severity of an attack, the stronger the security profile used.

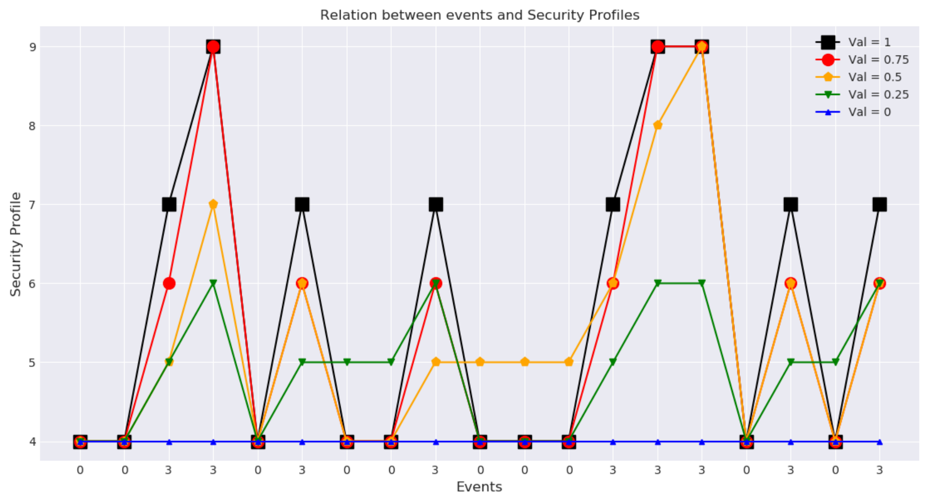

Figure 5.14: Random attacks with severity equal to 3

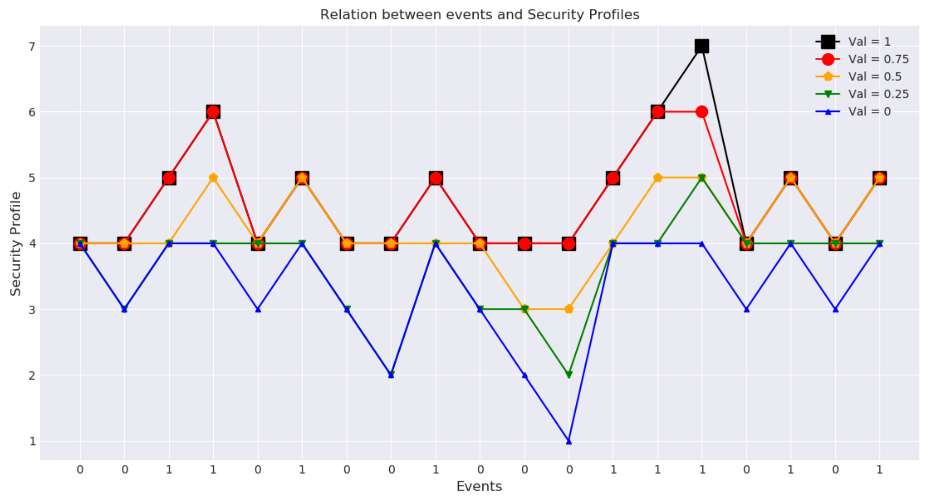

Figure 5.15: Random attacks with severity equal to 1 regarding identical tables

On the occurrence of an attack, the algorithm goes back to the first security profile if an inferior profile is used. An inferior profile is regarding another security profile with a lesser energy cost. This behaviour stands up when the security and energy consumption tables are equals. The figures 5.15, 5.16 and 5.17, exemplify this behaviour.

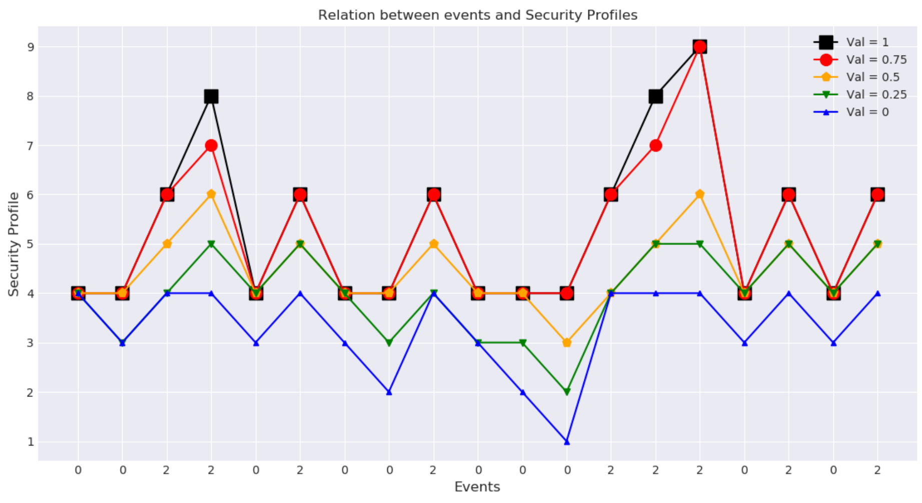

Figure 5.16: Random attacks with severity equal to 2 regarding identical tables

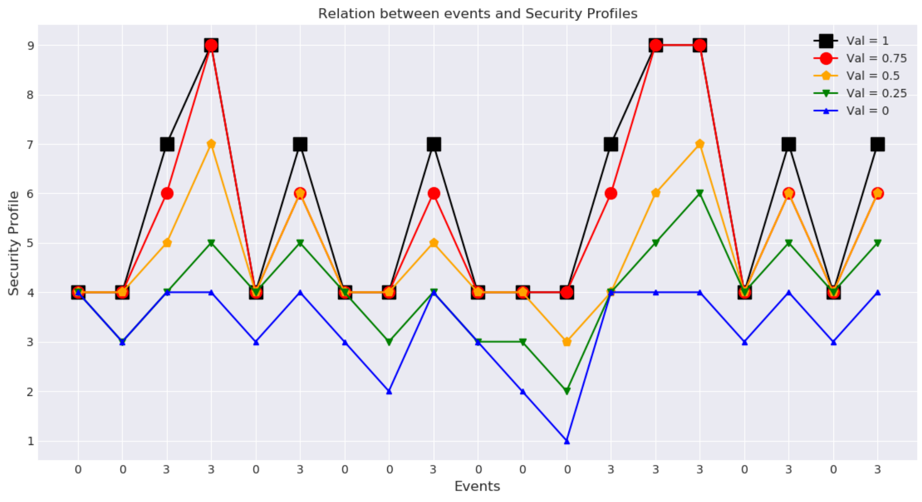

Figure 5.17: Random attacks with severity equal to 3 regarding identical tables

Images 5.18 and 5.19 present the algorithm behaviour when we have a set of random attacks. These attacks can have different severities.

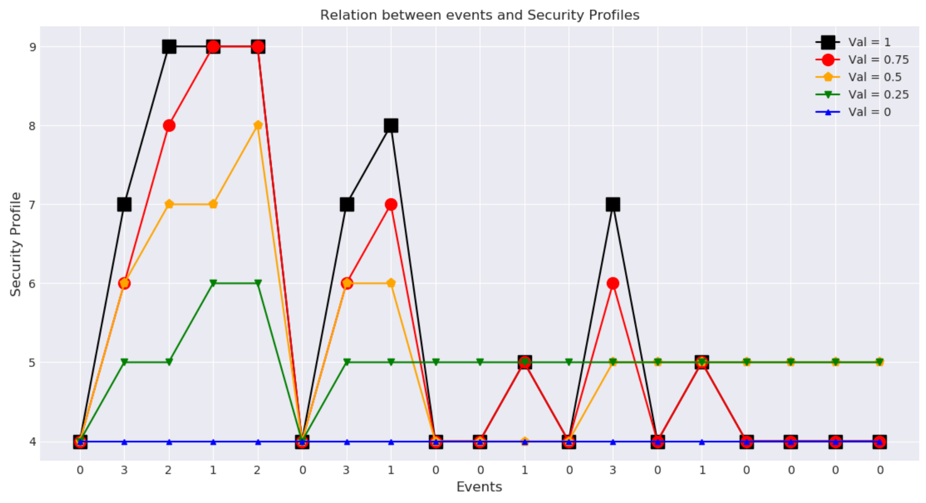

Figure 5.18: Random attacks with different severities regarding different tables

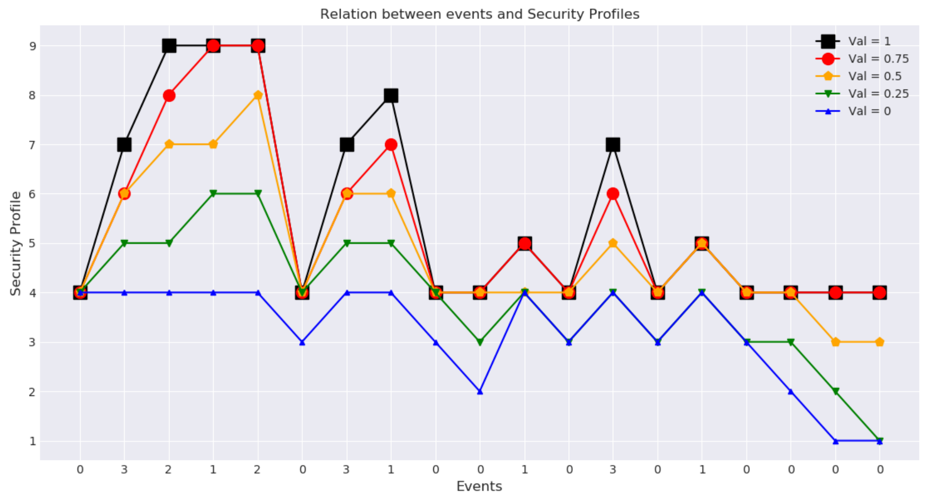

Figure 5.19: Random attacks with different severities regarding identical tables

5.2 Attacks to contemplate

A significant number of attacks can be performed on a WSN, as referenced in section 2.2 from chapter 2. Our focus is related to attacks carried out on the CoAP protocol to use the most suitable security profile, in a given context, thus ensuring confidentiality, integrity, and authentication of communications. The pertinent attacks are the ones mitigated with encryption, hash or authentication. So, we are speaking about external attacks. In this point is presented three applications using CoAP protocol. Then, it is presented external attacks frequent at the various contexts presented by the applications, but these attacks have different severities duo to these contexts. These attacks are the ones that will be considered taking into account the study and the testing of the proposed algorithm.

5.2.1 Applications

Application A

The first application is related to smart homes and is called A. It is a system that allows a user, the owner of the house, to measure and see the house energy consumption. It is possible to know this consumption in an aggregated form or by appliances. Additionally, it is possible to get access to the house temperature and provide commands to turn on or turn off the air conditioning. Moreover, it is possible to open or close shutters and turn off or turn on position lamps. In a smart home, we have several different message exchanges. These rates depend on each equipment used. Regarding this situation, the application A will send one message every 5 minutes. This rate was chosen because it is suitable regarding context presented.

Application B

The second application, called B, is another system related to weather stations. A weather station has a set of sensors. The hygrometer is used to measure the humidity. The pressure, of the local atmosphere, is taken by the barometer. The anemometer is used to confirm the wind direction and speed in a given moment. This data is gathered from the sensors, periodically, and then it is handled, and it is presented, for example, at a web page. This application will send one message every 10 minutes. This value was specified, taking into account a commercial weather station [DESIGN Technologies, 2019]. However, other transmission rates are used [of Miami, 2019].

Application C

The third application is related with actuators and sensors, in industrial control, and is the application C. It is another system to control an oil refinery. The system reads data from sensors such as pressure, temperature, flow or the level of the crude oil refinery process and, analysis this data to detect anomalies or unusual behaviours. Also, it is possible to send operational commands. These commands can be open or close valves, turn on or off compressors or pumps, among others. The transmission rate specified for this application is one message per minute. However, a presentation of the Royal Institute of Technology [Institute of Technology, 2019a] [Institute of Technology, 2019b], regarding Supervisory Control and Data Acquisition (SCADA) systems, presents other transmission rates, concretely a variation between 2 and 15 seconds. The IEEE Standard for SCADA and Automation Systems (IEEE C37.1-2007) was used as a reference [451, 2008].

5.2.2 Attacks

Different attacks can be performed, bearing in mind the previous applications contexts. Attacks such as communications amplification, packets spoofing or messages blocking and delayed can be carried on. However, these attacks have different severities regarding the previous applications context.

The spoofing attack

In a spoofing attack, an attacker can forge messages in response to confirmable and non-confirmable packages. Also the same applies to ACK packets in response to a CON message. Taking into account the previous applications, an attacker can create a spoofed request. This request can be used to turn on the air conditioning, to gather the weather data from the sensors outside the natural collection period or to open a valve from the distillation unit. Once again, these actions performed by this intruder have different severities. To application A, the action will change the optimal temperature, and it will instil a higher energy consumption, so this action has an impact. To application B, it has lesser importance taking into consideration the consequences. However, regarding application C, the effects are significant because we have a critical system presented. This action can result in an unpredictable and dangerous impact, such as the explosion of the conduct distillation unit. The authentication has a more significant role in mitigating these attacks, besides the hashing when a packet is modified.

The amplification attack

In the amplification attack, an aggressor modifies original IP. Also, he can increase network traffic, putting the network in an overburdened state. If the attacker holds a request packet and changes the origin IP address, the server will replay to this IP. This response has a greater size than the request. Taking into account the application A, some appliances may work with delay or even can be blocked with responses from a server. The same applies to applications B and C. In application A this attack is relevant because the owner of the house cannot see the data in real-time or cannot execute commands. This situation may occur in C, and the same situation is more dangerous because we are in the presence of a critical system. If the data gathered from the sensors is not fresh or if a user cannot execute an operational command in time, the system can be damaged. This time, usually, is a short execution time. Regarding application B, the system may become slower. This attack can be mitigated if a server does not accept requests that are not authenticated.

The block attack

The block attack consists of blocking requests or responses between clients and servers. The use of encryption makes selective blocking of packets harder but not impossible because IP address, ports or CoAP messages lengths are available. Concerning actuators, this attack is critical. A request can be performed with a given action. If it does not reach the destination, the transmitter does not know if the operation was performed or not. The same applies to the response. The attack is severed and vital for applications A and C because a client loses the server state. If a closed valve command is sent by application C, the client will not know if the command was executed or not. The same applies when an open shutter command is sent in scenario A.

To application B when a request is performed to get the weather data, and then it is blocked, the final user will not have data updated to see. As previously described, the encryption complicates selective blocking. The users of the systems must be warned when the system do not receive a resource or a confirmation related to the required action.

5.2.3 Severities, Val values and initial security profiles

Concerning the previous attacks and applications presented, we have different severities to the same offence. It is a typical situation because the context of the use is distinct. In application A, we have a system with some importance where the privacy of the user data is essential. In C, we have a critical system. That is why it is vital to carry out the enforce of commands in a short time or perform constant monitoring. Finally, an application B is presented as a data collection system where the correct and the up-to-date data is relevant, taking into account the collection period. The following table 5.2 shows this information in a summarised way:

Table 5.2: Severity of the attacks regarding each application

Applications\Attacks	**Spoofing**	**Amplification**	**Block**
A	3	2	2
B	1	2	1
C	3	3	3

The previous table 5.2 presents the severity of each attack, bearing in mind the context of the applications introduced. All attacks have the maximum hardness on C application because it corresponds to a critical system. The spoofing attack has gravity three on application A because the user data is crucial and private. The other offensives have gravity two because when they occur, the system will not have critical behaviour such in C. The amplification attack has a severity factor of 2 regarding application B because, apart from flooding a sensor, the network as a whole can become also flooding. It is wise to define the initial security profile equals to P8, P2 and P9 to application A, B and C. Also, regarding the characteristics of the applications, the values of Val equal to 0.75, 0.25 and 1 are suitable. The following table 5.3 summarised this information.

Table 5.3: The properties of each application

Applications\Properties	**Initial security profile**	**Val values**
A	P8	0.75
B	P2	0.25
C	P9	1

5.3 Complementary analytical validation with results

The previous analytic validation does not have into account the total energy cost of the use of each security profile, neither the battery lifetime of a given node. It had into account just the energy and security tables, the various values of Val and the occurrence of random attacks with random severities. The main objective was to prove the algorithm adaptation regarding the presence of an attack. So, the previous validation was performed considering a generic application and general strikes. This complementary validation aims to embrace the previous implementations and attacks identified and characterised.

5.3.1 Energy cost, number of messages and lifetime

The energy cost of one day was calculated, considering the energy of the use of each security profile and the message exchange rate of each application. The following table 5.4 presents the final results:

Table 5.4: Energy cost and message exchange rate of each application

Applications\Properties	**SP**	**Message exchange rate**	**Number of requests**	**One day energy cost (J)**
A	P6	5 \minute	288	4.267
B	P2	10 \minute	144	1.117
C	P7	1 \minute	1440	33.856

When the previous applications were identified and characterised, it was stipulated different initial security profiles. However, new initial security profiles were used to demonstrate the controller logic. The same was applied to the Val values. Specifically, the used values of Val were 0.5, 0.25 and 0.75 to application A, B and C. The energy cost of one day was achieved performing the multiplication of the energy cost of one CoAP request by the number of requests. This cost was calculated, taking into consideration the initial security profile used, and the message exchange rate of each application. Another numerical calculation that can be done is the number of days that a given node can operate. This value is presented at the next table 5.5.

Table 5.5: The number of days regarding each application

Applications	**Number of days**
A	2029.064
B	7749.376
C	255.728

Once again, these values were calculated, dividing the total energy of the battery of the M3 nodes by the previous day energy cost calculated. The M3 nodes can use 650 mAh batteries, and in one hour, we have 8658 Joules of energy. This value is achieved using the following formulation:

$$0.65\text{Ah} * 3600\text{Seconds} * 3.7\text{Volts, [Electronics, 2018].} \tag{5.1}$$

The previous values are valid data regarding static security or without the use of the controller, regardless of the presence or not of attacks. So, as to study the controller regarding energy consumption and the battery lifetime, three frequencies of attacks were stipulated. Then, they were simulated by each application. With this in mind, it was calculated the energy cost of one day taking into account each use, and each attacks frequency. Finally, the Val value was altered to verify the consumption variation.

5.3.2 Definition of the attacks frequency

These frequencies of attacks are standard in the three applications. In previous was presented the number of CoAP requests that a given application makes in a single day. To remain, application A makes 288 requests, B exchange 144 and C creates 1440. With these numbers in mind, the frequency of attacks stipulated is 25%, 50% and 75%. The following table 5.6 presents the number of strikes, regarding the total messages:

Table 5.6: The number of attacks regarding each attack frequency

Applications\% of attacks	**25%**	**50%**	**75%**
A	72	144	216
B	36	72	108
C	360	720	1080

After the specification of these frequencies, some results are displayed for each application.

5.3.3 Application A

In figure 5.20 the blue line represents the initial security profile of application A. This figure demonstrates that in attacks frequency equals to 25% the energy cost of one-day utilisation is lesser than the standard security profile. Clearly, with the use of the controller, we probably have an energy saving. The blue bar represents the occurrence of attacks of all types. The green, orange and red bars are related to amplification, block and spoofing attacks. If we remember the severity of each attack, regarding this application, they have 2, 2 and 3 regarding the degree of seriousness. That is why we have a higher energy cost at green and orange bars when the frequency of attacks is equal to 25%.

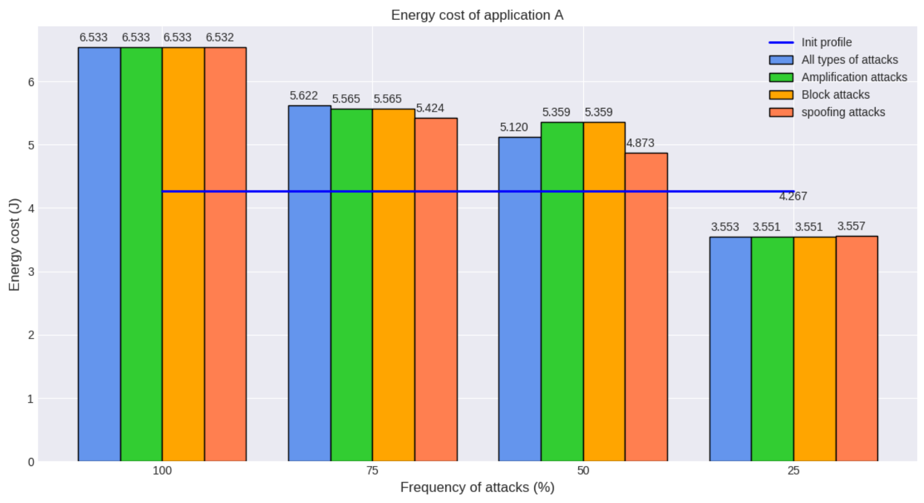

Figure 5.20: Energy cost of one day regarding each frequency and type of attacks

Looking at attacks frequency when this frequency is equal to 50%, we have a surprising behaviour. Let us remind the energy table presented previously. Application A has the initial security profile same to P6. This profile is the fourth most expensive regarding the energy table. With the occurrence of a lesser severity attack and a Val value equal to 0.5, the controller upgrades security more slowly. Contrarily, the controller uses more quickly the security profile P9. This profile has a lesser energy cost than P7. Also, when the top profile is used, in the presence of no attacks, the controller downgrade the security backing to the initial profile.

With this behaviour in mind, figure 5.21 shows the number of days regarding each attack frequency.

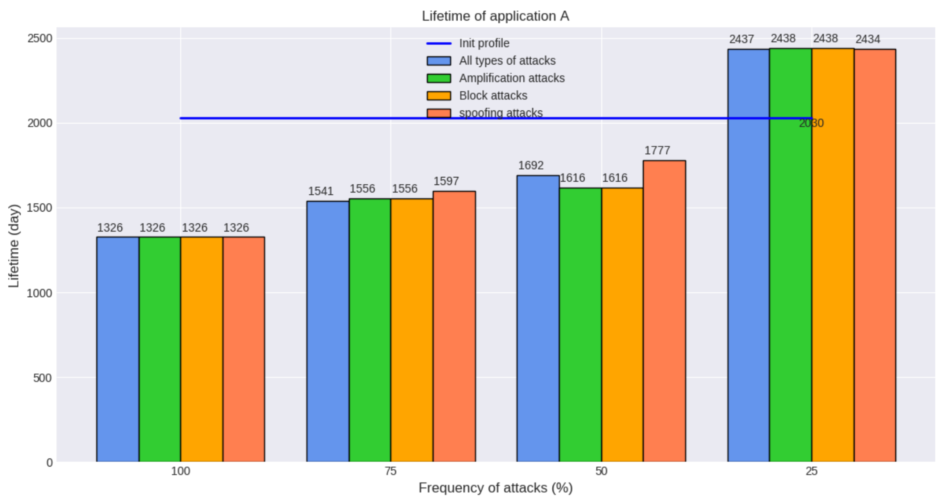

Figure 5.21: The number of days related to the energy cost of a day

Once more, the energy saving is visible in the frequency of the attacks equal to 25%. One interesting point, after the results presented, is seeing what happens when we change the Val value. The following figures describe the controller behaviour, in the context of the application A.

Val value of 1

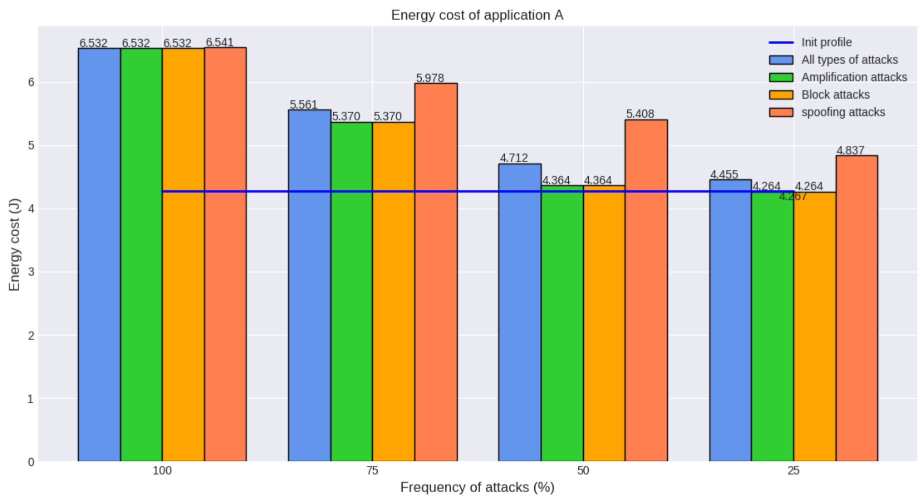

Figure 5.22: Energy cost of one day regarding each frequency and type of attack when Val value is equal to 1

In figure 5.22 and when the Val value is equal to 1, we do not have energy saving. This behaviour is normal because security is critical. This behaviour implies a decreasing number of days, as shown in the next figure 5.23.

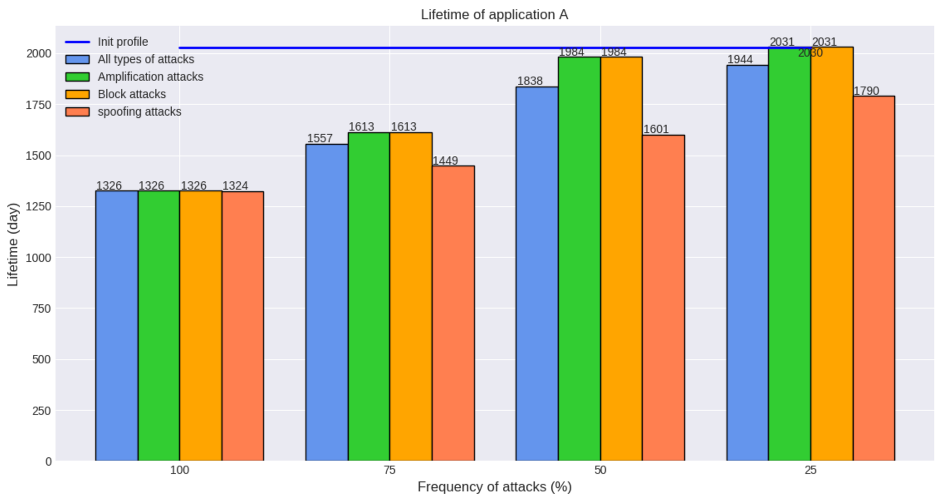

Figure 5.23: The number of days related to the energy cost of a day when Val value is equal to 1

Val value of 0.75

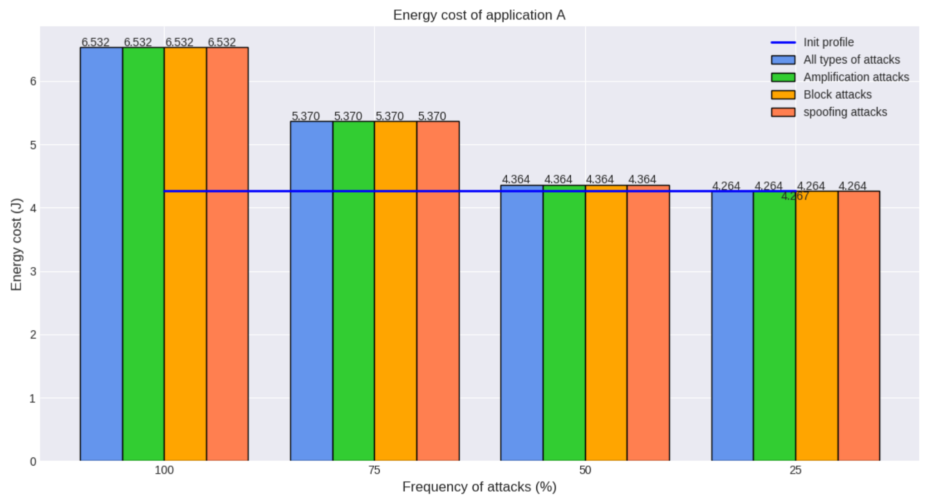

Figure 5.24: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 0.75

Figures 5.24 and 5.25 show the energy consumption of the algorithm when Val value is equal to 0.75. We can see that there is no variation of results regarding the same frequency of attacks. This behaviour is regarding the performed calculus of the security profile adaptation, concretely the rounding when the controller changes the profile. The energy-saving is almost nil.

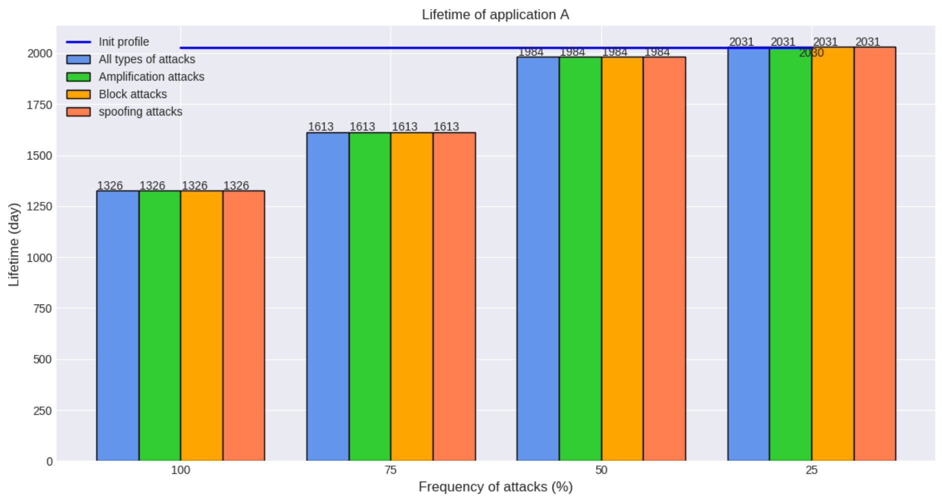

Figure 5.25: The number of days related to the energy cost of a day when Val value is equal to 0.75

Val value of 0.25

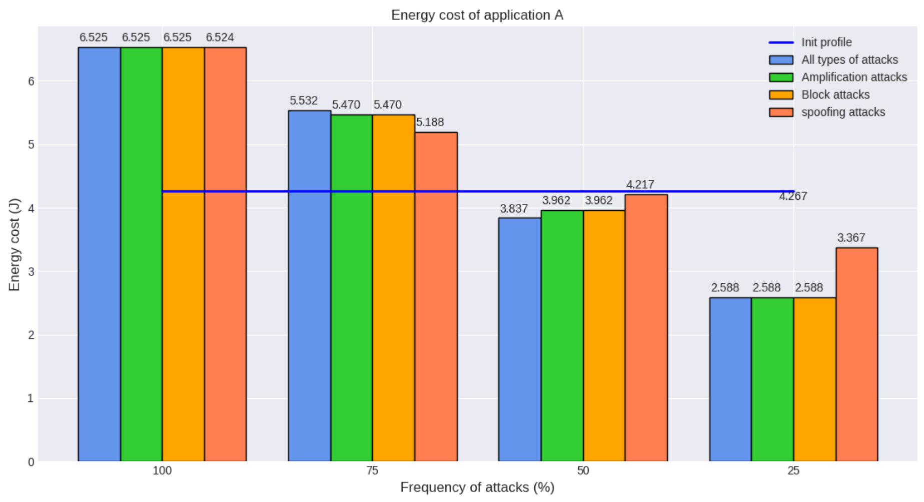

Figure 5.26: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 0.25

With a Val value equal to 0.25 we have an energy-saving behaviour to a frequency of attacks similar or less than 50%.

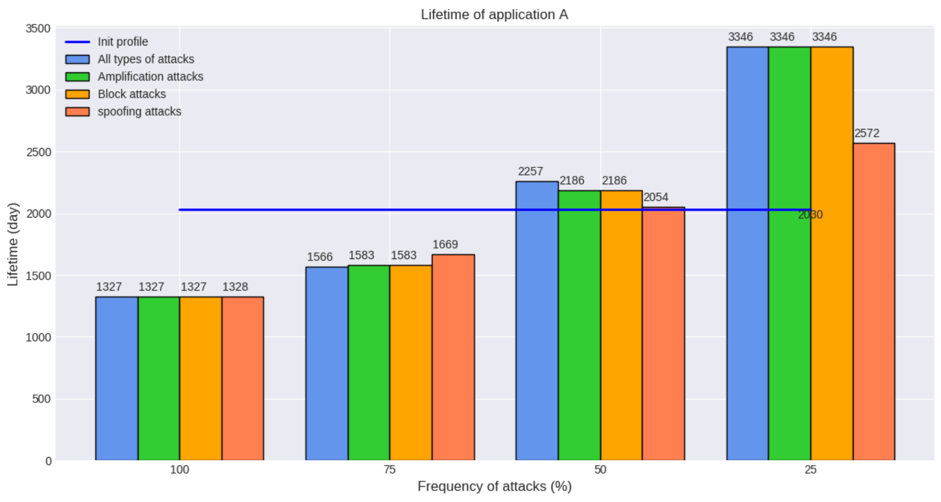

Figure 5.27: The number of days related to the energy cost of a day when Val value is equal to 0.25

5.3.4 Application B

The following properties distinguish the application B: Val value is 0.25, and the initial security profile is P2. This application assigns different severities to the attacks presented previously. The amplification attack has gravity 2, and the block strike has the same severity of the spoofing attack. The value is 1 of these attacks.

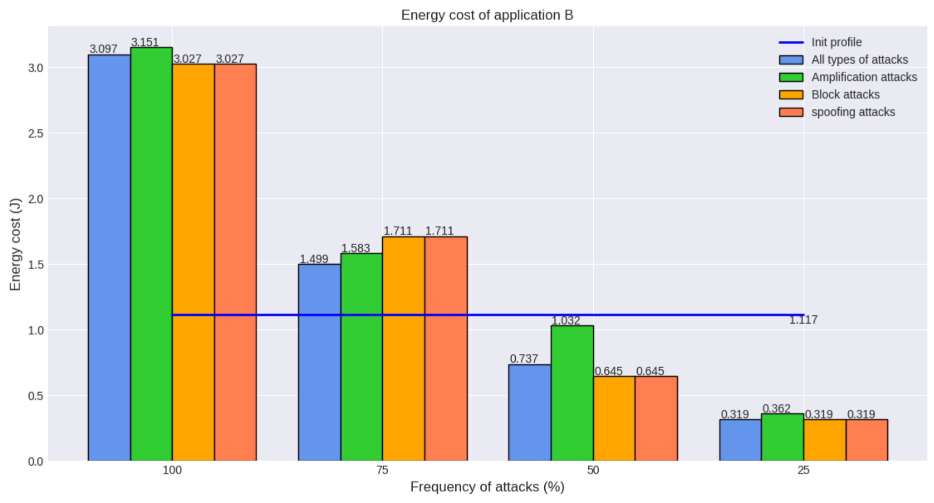

Figure 5.28: Energy cost of one day regarding each frequency and type of attacks

Once again, the blue line represents the energy cost or the number of days regarding initial security profile. The green bar, when the frequency of attacks is similar to or less than 50%, is higher. This behaviour is normal because this bar represents the attack with the most higher severity. Again, a surprising cost is presented when the frequency of the attacks is equal to 75%. This pattern is the same behaviour explained previously in application A when the controller updates security in a slowly way. The following figure presents the number of days regarding the node lifetime.

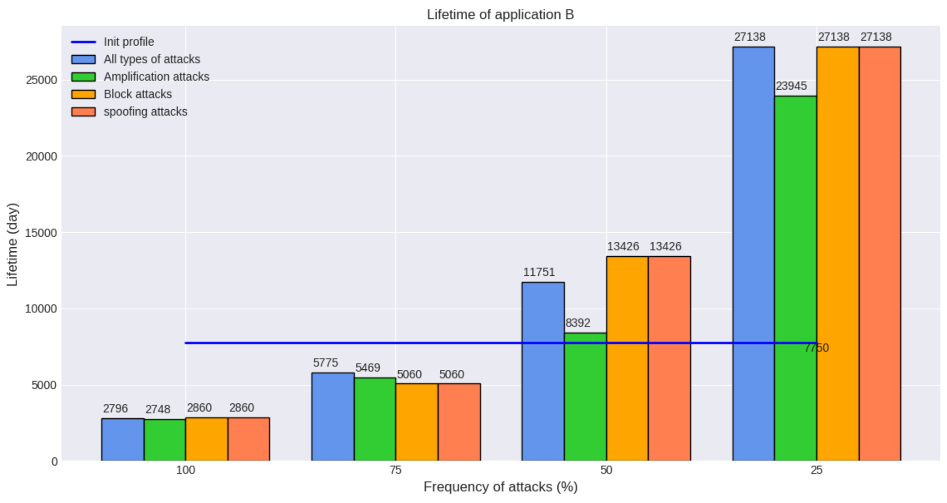

Figure 5.29: The number of days related to the energy cost of a day

The usage of this application increases the lifetime of a given node because the security is not essential. Again, the Val values were changed, such as on application A. The possible values are upper than the standard, that is why the energy cost will increase, and the number of days will decrease.

Val value of 1

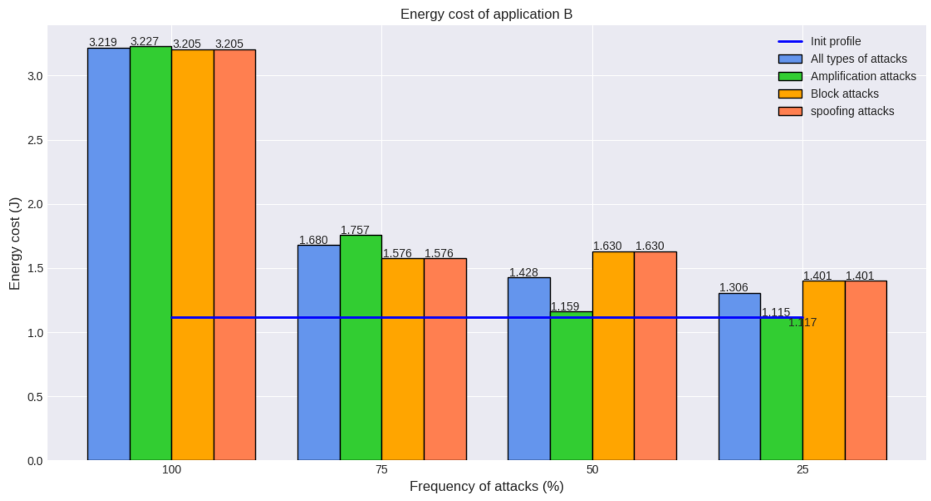

Figure 5.30: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 1

Again, when the Val value increases, the energy cost of one day increase 5.30, and the number of days decreases, as shown in image 5.31. In this situation, we do not have energy saving.

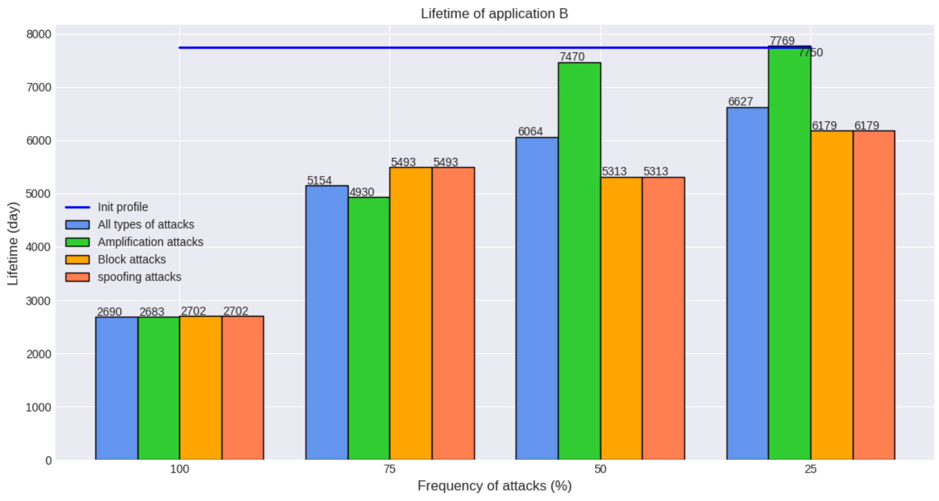

Figure 5.31: The number of days related to the energy cost of a day when Val value is equal to 1

Val value of 0.75

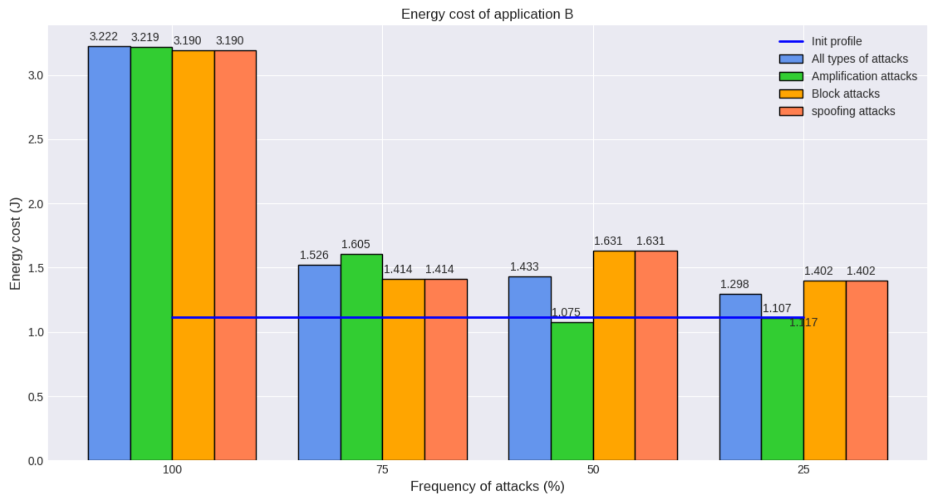

Figure 5.32: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 0.75

Comparing the results when Val value is 0.75 with the previous ones when Val is equal to 1, we have a small decrease regarding the energy cost of one day.

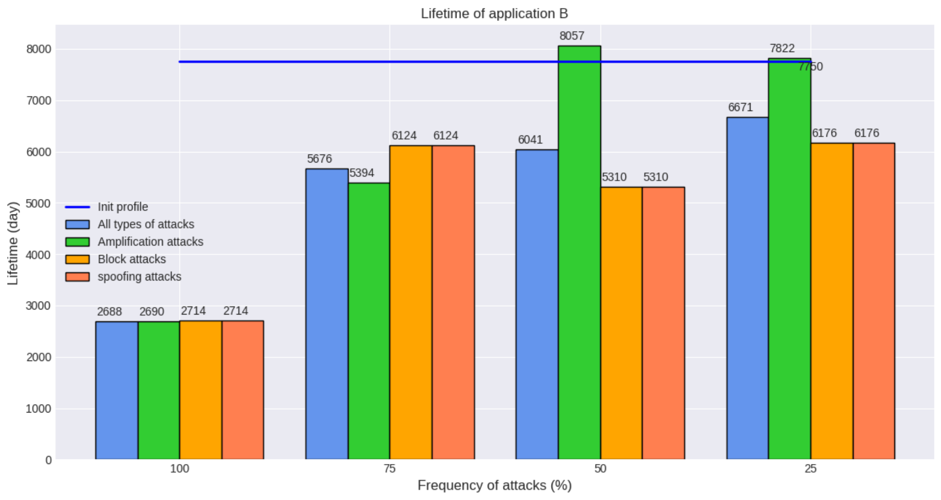

Figure 5.33: The number of days related to the energy cost of a day when Val value is equal to 0.75

Again, a surprising cost is presented, in the previous four figures, when the frequency of the attacks is equal to 50% and 25%. This pattern is the same behaviour explained previously in application A.

Val value of 0.5

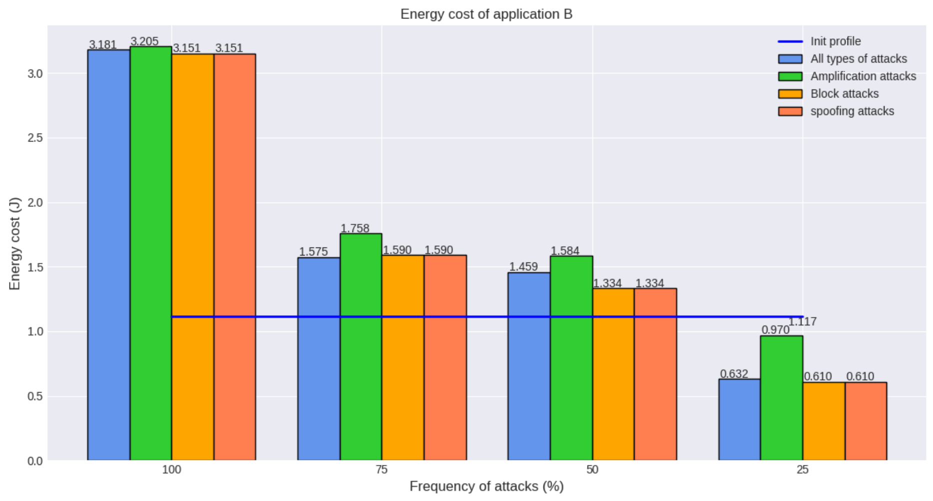

Figure 5.34: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 0.5

When the Val is equalled to 0.5, we can have energy-saving because of the controller behaviour. It is possible to downgrade the security profile below the initial SP.

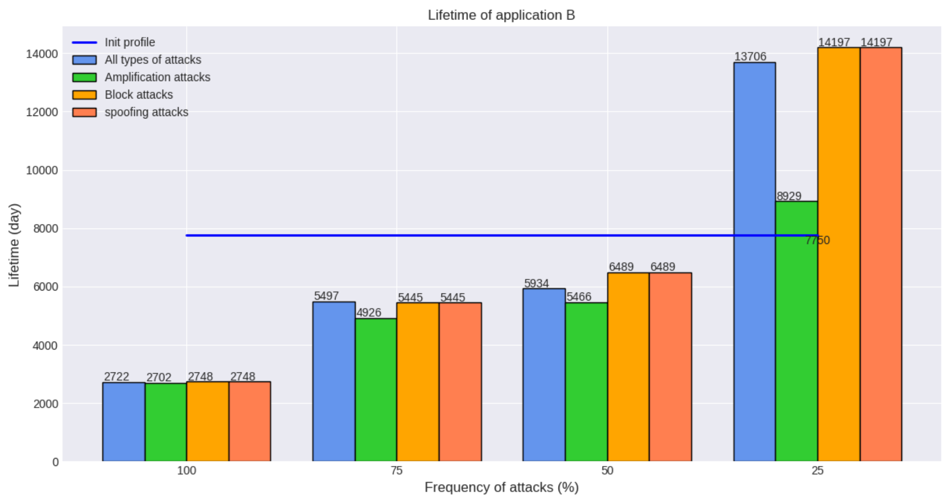

Figure 5.35: The number of days related to the energy cost of a day when Val value is equal to 0.5

5.3.5 Application C

This application uses the initial security profile P7 and Val value equal to 0.75. All attacks have the same severity same to 3. For this application, security is critical.

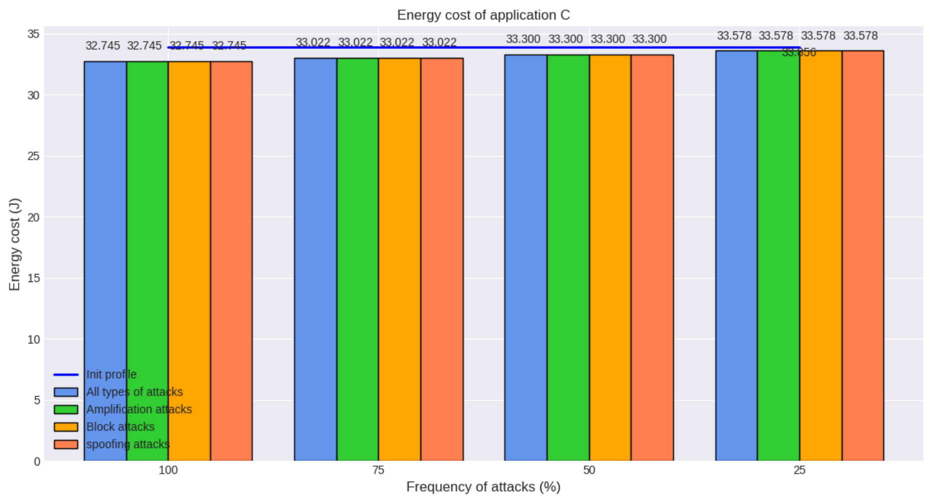

Figure 5.36: Energy cost of one day regarding each frequency and type of attacks

A curious behaviour is presented in image 5.36. We can have a slight gain related to energy saving. This gain is possible because the initial security profile is the P7. This profile has a higher energy cost regarding the energy table. So, the controller will use more often the P9 that has a slightly lower charge but a higher security level. That is why we can have two or ten days above the initial security profile. This number of days is presented in figure 5.37.

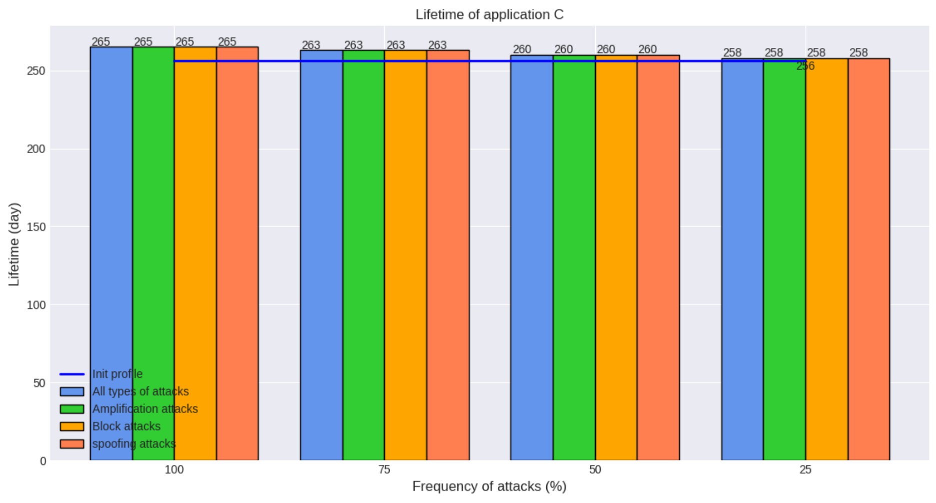

Figure 5.37: The number of days related to the energy cost of a day

The following figures are related to the variation of the Val value like performed in the previous contexts.

Val value of 1

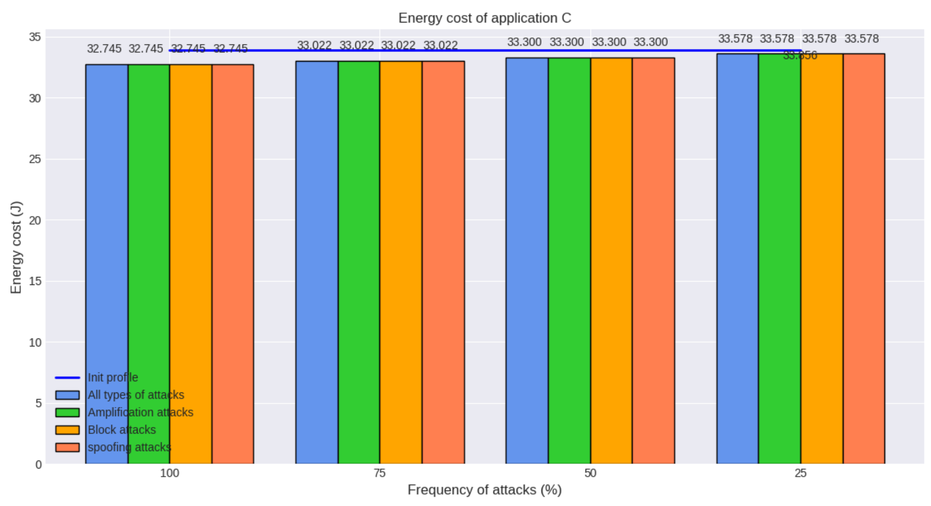

Figure 5.38: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 1

The behaviour presented when Val is 1 is similar when it is 0.75. The explanation regarding this behaviour is related to the formulation used to upgrade the controller and the maximum number of security profiles presented in energy and security tables. Although the value 0.75 is not equal to 1, the controller will update the SP just the same. The P7 plus 2.25 equals to P9 such as P7 plus 3.

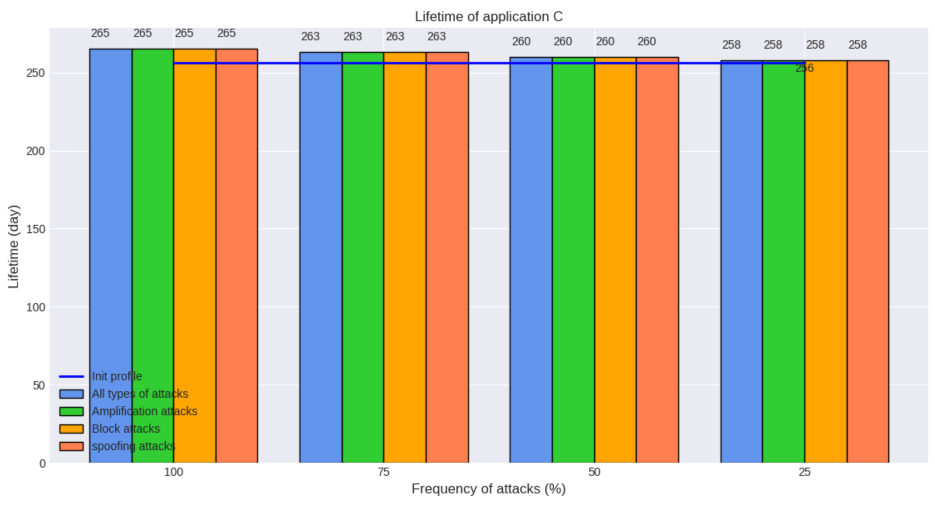

Figure 5.39: The number of days related to the energy cost of a day when Val value is equal to 1

Val value of 0.5

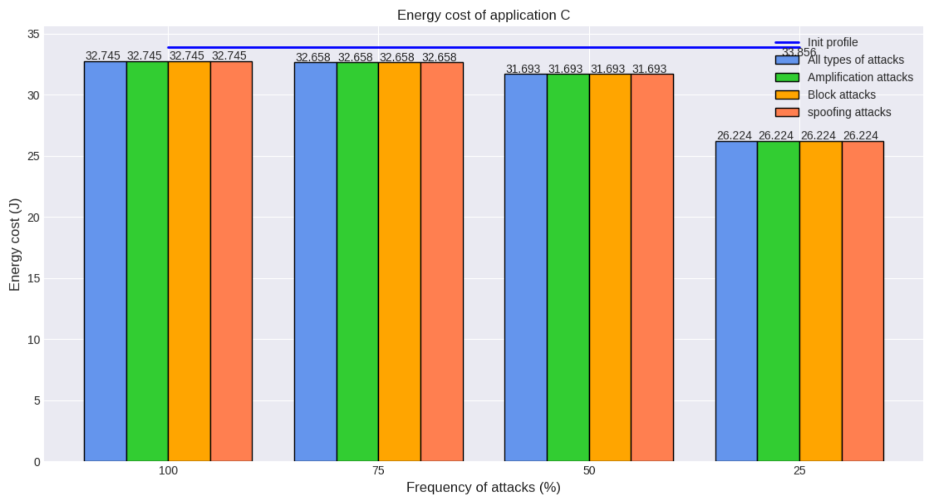

Figure 5.40: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 0.5

When Val is decreased, the gain related to energy saving is increased as expected. This behaviour is presented in figure 5.40 and 5.41. We have the same values regarding the different attacks because they have the same degree of severity.

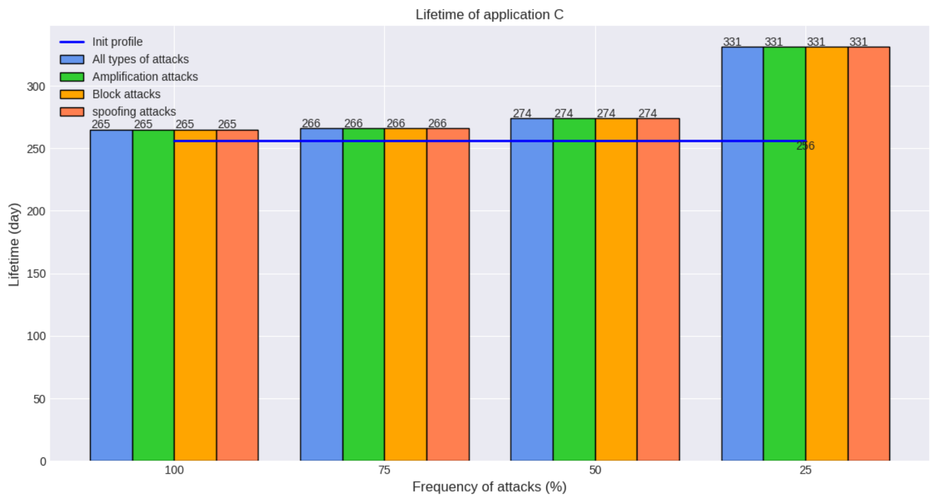

Figure 5.41: The number of days related to the energy cost of a day when Val value is equal to 0.5

Val value of 0.25

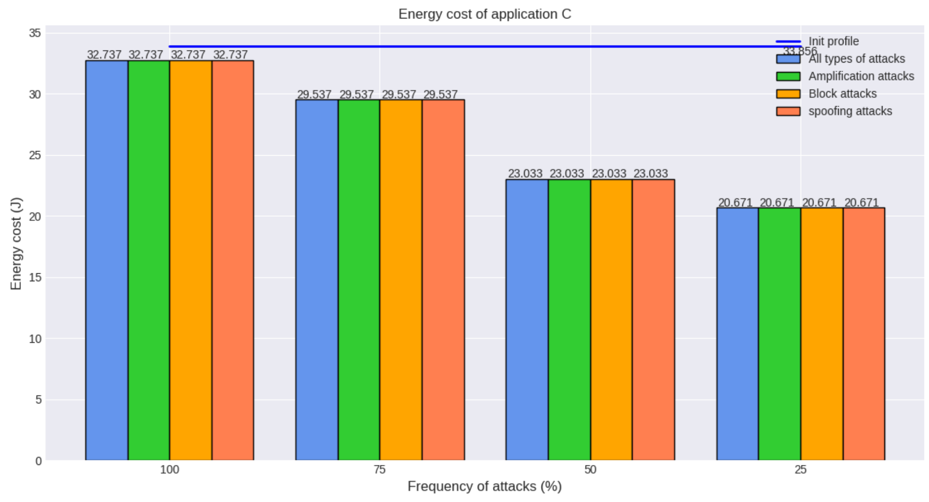

Figure 5.42: Energy cost of one day regarding each frequency and type of attacks when Val value is equal to 0.25

When the value of Val is 0.25, the energy-saving is more visible. Also, we can see the relation regarding the frequency of attacks and the energy cost. With the decrease of this frequency, energy consumption also decreases.

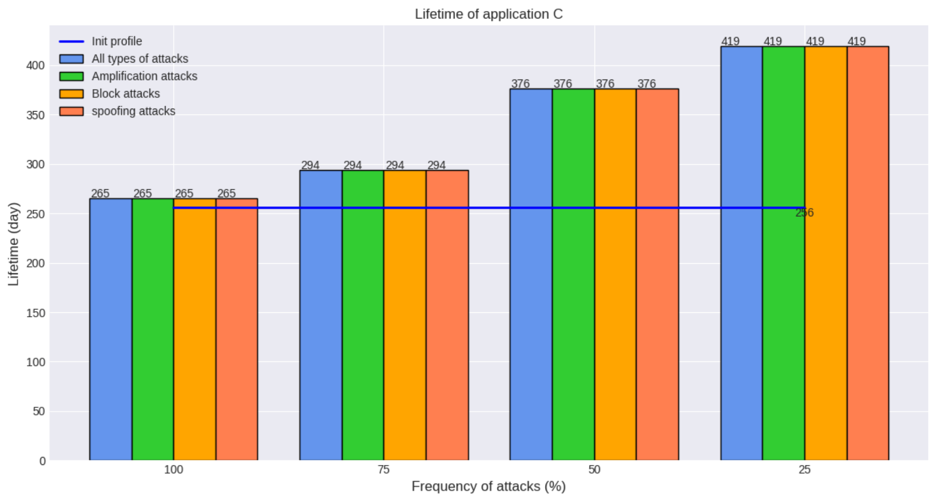

Figure 5.43: The number of days related to the energy cost of a day when Val value is equal to 0.25

5.3.6 Attack rate

Taking into consideration the previous results, the frequency of attacks with the best energy-saving results is at 25%. The behaviour is normal and makes sense because it represents the expected behaviour of the controller. Another essential variable is Val value. We previously demonstrated the pattern related to this variable, but the information embraces other components as well. This sub-section seeks the analysis of Val in a complementary way.

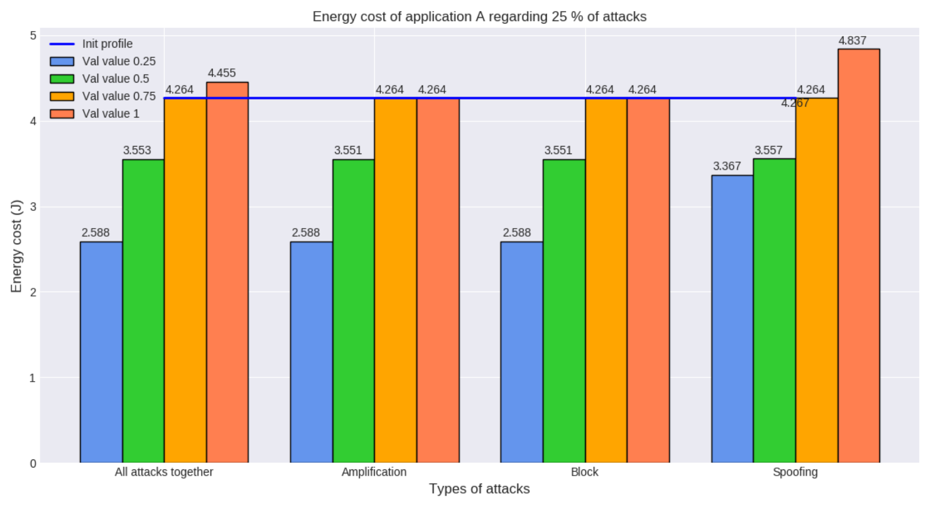

Figure 5.44: Variation of Val values regarding application A and the different attacks

For each application, the attack frequency is the same, and the Val values were varied regarding each type of attack. In figure 5.44, regarding application A, we can see the increase of the cost values when the Val increases. The same applies to application B and C.

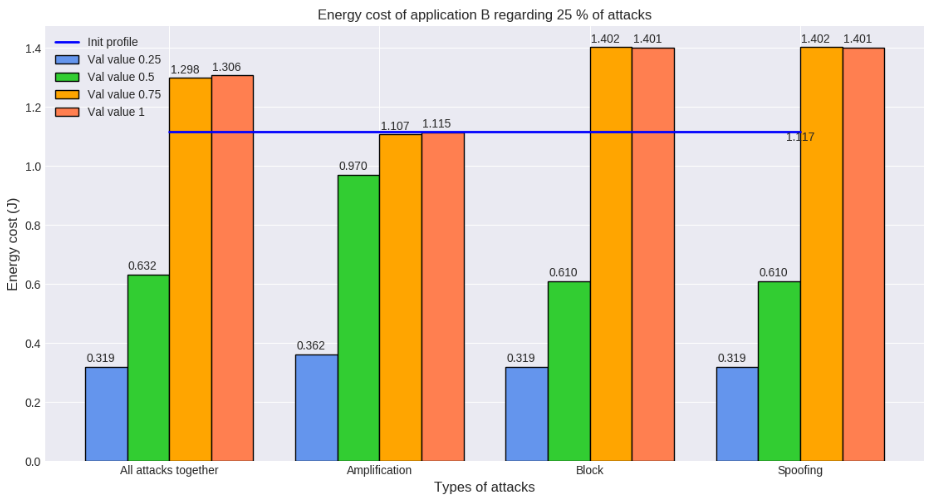

Figure 5.45: Variation of Val values regarding application B and the different attacks

In all figures, we can see one notorious behaviour identified previously and reported. For Val with values of 0.75 or 1, we have the same result. In image 5.44, we can see, regarding the amplification and block attacks, the same energy cost when Val has values of 0.75 and 1. In figure 5.46, for all attacks and the same Val values, the same behaviour happens. This pattern occurs in all attacks because, in application C, all the attacks have the same severity. Regarding the application B, the same appear in attacks of spoofing and block for the same reasons. Also, in A, the amplification and block attacks have an equal severity value.

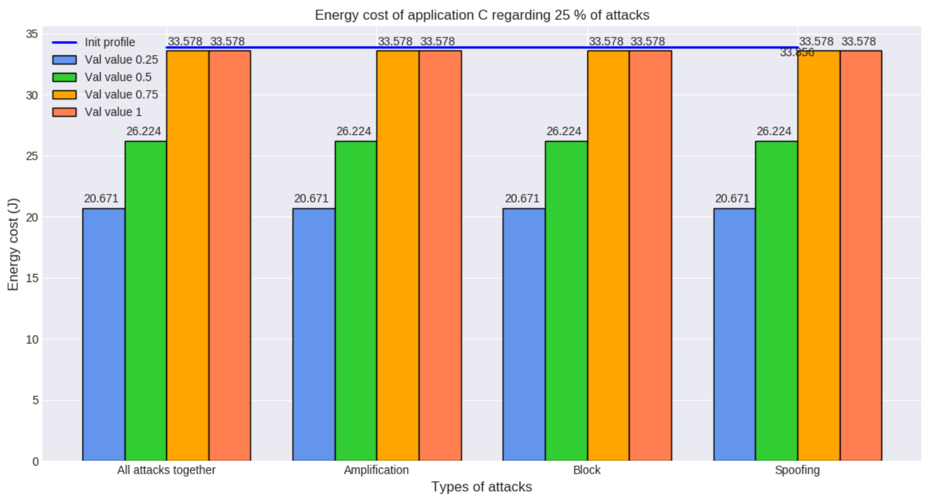

Figure 5.46: Variation of Val values regarding application C and the different attacks

5.4 Summary

At the beginning of the controller analysis, an initial examination was made. With this assessment, it was possible to show the controller behaviour regarding the presence or absence of attacks. The pattern consisted of upgrading or downgrading security.

In the complementary analysis, three applications were presented with different properties. These properties are related to the initial security profile, the Val value, the exchange rate of messages and finally, the severity given to each type of attack. Regarding the exchange rate, the number of requests for each application was calculated for one day. Then, the energy consumption cost of a single day was computed, taking into consideration the message exchange rate. With this energy cost, the number of days was reached for the initial security profile of each application. The first conclusion achieved by the simulation of the variables set is related to the message exchange rate. Increasing this rate increases energy consumption. Figures 5.26 and 5.42 demonstrated this reason, even they belong to different applications and the controller begin at different security profiles. The difference in the cost is very high. Finally, three frequency of attacks was provided.

Regarding application A, the controller can perform energy saving at attacks frequencies equal to 0.25 and 0.5%. This energy-saving potential is possible when Val has values 0.5 and 0.25. The same applies to application B. Also, for Val same to 0.75 we can have, in some situations, energy saving. This situation is regarding the controller behaviour explained previously.

Security in application C is fundamental. This application has the highest value regarding the security profile, the Val value or the attacks severities. However, with the use of the controller, we can have a slight gain related to energy saving. This gain is possible because the controller uses a second table regarding energy cost. Such as observed in the other applications, when Val is equal to 0.5 and 0.25, it is possible to have energy-saving for attacks frequencies of 50 or 25%. For a rate of 75%, an energy-saving is potential when the Val value is 0.25. This situation only is visible in this application.

After this analysis, the most critical variable of the controller is the weight given to security. Then, the set of energy and security tables is second. These two variables allow security adaptation and can provide energy saving. Independent of the initial security profile or even the severity of each attack, all applications perform security adaptation. Also, all applications can achieve energy saving using the Val values 0.5 and 0.25. This behaviour is remarkable and expected. It is possible because the controller can downgrade a security profile under the initial profile. Finally, the last important variable is regarding the attacks rates because with the increase in these rates, the energy-saving decreases. With the attack rate of 25%, all applications can perform energy saving. In a nutshell, the optimal variables to carry out energy-saving are Val, with values of 0.25 and 0.50, and attacks frequencies, respectively the ones with values of 25 and 50%.

Chapter 6

Conclusion and future work

In this chapter, the conclusion of the work performed is presented. Also, points related to future work are presented and discussed.

6.1 Conclusions

In this thesis, an overview regarding the IoT was made. The functional and security requirements were searched, and then a set of the most used ones was presented. Then, the reader might read regarding attacks that can be performed in WSN. Also, the most used protocols, in the IoT context, were displayed. Finally, works were presented and analysed to handle the security-energy trade-off. In the context of these works was identified some gaps and relevant aspects. These were not treated or were not addressed suitably. An absence of papers, bearing in mind security achievement in the application layer, especially at CoAP, is an example. Also, another lack of works regarding energy-security trade-off, particularly the ones that handle the previous compromise in an adaptive way are scarce.

Regarding the previous aspects and the established objectives and metrics, a controller proposal and formalisation was presented. Then an analytic validation was performed. The simulated parameters, used in this validation, were the attacks with severities, scenarios with strikes, the security weight given by stipulated applications and the static security profiles, to achieve security adaptation regarding the agreed objectives. Also, real energy consumption was used to estimate energy costs regarding each security profile and specified application.

In a nutshell, the goals proposed were:

- the measurement of the security algorithms to rank and to used them at a given context.
- The resistance of external attacks.
- The accomplishment of secure communications. This accomplishment must be performed between applications and devices using the proposed security algorithms.
- The measure completion of the node energy cost. This measurement is regarding static and adaptative security.
- The change fulfilment of the security algorithm in use, taking into account the security requirements of applications and context.

- The measurement fulfilment of the computational effort regarding security algorithms.

These proposed goals were established to fulfil the identified requirements besides architecture. The critical architecture component is the controller. This component is responsible for performing security adaptation and handle the security energy trade-off.

The controller component was the first architecture component to be analysed and implemented regarding its importance. Regarding the analytical validation performed and the results presented in the 5.4 section, the controller carries out the security adaptation and also handle the security-energy trade-off, saving energy. Also, it deals and fulfils the security requirements of a given application, besides the type and severity of each attack. In a nutshell, the controller implementation ensures the ability to perform security and energy consumption adaptation. Also, these facts become validated with analytical validation. Besides, experiments were conducted, regarding each security algorithm, and then for each one, the energy cost was achieved. These points are the main contributions of the work performed in this thesis context.

With the controller implementation and validation, the following sequence, regarding the analysis and usage of the other architecture modules can be conducted. The battery module can be analysed and integrated with the controller to use real energy data with the security adaptation. Then, a real integration with different applications, preserving their requirements is a logical idea. This idea would permit the analysis of the controller behaviour with real data and not with simulated data. The same applies to the IDS module. Then, the establishment of communications between applications without security is a way to establish a simple bridge between two entities, to set sail to secure communications. Finally, the connection with the various controllers could be performed to exchange implementations details.

6.2 Challenges and future work

In this section, the challenges and future work are presented. The controller has a massive potential to be used with various technologies and in variable contexts. The CoAP environments are the ones, previously identified, besides the different WSN contexts. This high potential is regarding the lack of works in the approached area. In the following points is presented some examples where the controller might be applied:

- The controller implementation in the Contiki OS: One future work is to perform the controller implementation at Contiki, taking into account the technologies and methodologies presented at the architecture chapter. Then, the IoT-LAB platform is suitable to carry out experiments and to create real networks scenarios.

- Perform the expansion of the controller to use more security algorithms: The used security algorithms were identified as the ones most used in the WSN context. That is why they were analysed regarding energy consumption. However, other security approaches might be used and tested, such as the ones that use asymmetric cryptography. Also, this would allow the protection of different types of attacks, besides the ones addressed.

- Use the controller with other nodes architectures: The IoT-LAB provides the usage of different types of nodes. Also, the Contiki OS has support to other nodes. The aim of using the controller with more than one node architecture is to discover the best one, performing a performance analysis of the proposed controller.

- Perform the controller implementation in other OS: As presented, other Operations System are used in IoT. The idea is to analyse the controller regarding each OS, regardless of the used hardware architecture.
- Test various battery models: To discover which is the best regardless of the used node type.

The set of challenges and future work is an example of the controller proposal potential to perform security adaptation and handle the security energy trade-off. The presented controller allows the use of the best suitable security profile regarding a given context. Also, this context can be adapted and extended, besides the suggested one.

This page is intentionally left blank.

References

[451, 2008] (2008). Ieee standard for scada and automation systems. *IEEE Std C37.1-2007 (Revision of IEEE Std C37.1-1994)*, pages 1–143.

[14443, 2018] 14443, I. (2018). ISO/IEC 14443. `https://www.iso.org/standard/73596.html`.

[18092, 2015] 18092, I. (2015). ISO/IEC 18092. `https://www.iso.org/standard/67864.html`.

[Aaron Gifford, 2012] Aaron Gifford, T. (2012). Open source (bsd licensed) secure hash algorithm (sha) implementations in c by aaron gifford, including sha-1, sha-224, sha-256, sha-384, and sha-512. `https://aarongifford.com/computers/sha.html`.

[Adelantado et al., 2017] Adelantado, F., Vilajosana, X., Tuset-Peiro, P., Martinez, B., Melia-Segui, J., and Watteyne, T. (2017). Understanding the limits of lorawan. *IEEE Communications Magazine*, 55(9):34–40.

[Airehrour et al., 2016] Airehrour, D., Gutierrez, J., and Ray, S. K. (2016). Secure routing for internet of things: A survey. *Journal of Network and Computer Applications*, 66:198–213.

[Albreem, 2015] Albreem, M. A. (2015). 5g wireless communication systems: Vision and challenges. In *Computer, Communications, and Control Technology (I4CT), 2015 International Conference on*, pages 493–497. IEEE.

[Alliance, 2018a] Alliance, L. (2018a). About lora Alliance^{TM} — lora Alliance^{TM}. `https://lora-alliance.org/about-lora-alliance`.

[Alliance, 2018b] Alliance, L. (2018b). What is the LoRaWAN^{TM} specification? `https://lora-alliance.org/about-lorawan`.

[Alliance, 2012] Alliance, T. Z. (2012). Standards: Zigbee specification. `http://www.zigbee.org/download/standards-zigbee-specification/`.

[Alliance, 2018c] Alliance, Z. (2018c). About us — zigbee alliance. `http://www.zigbee.org/zigbee-for-developers/about-us/`.

[Association, 2018] Association, G. S. M. (2018). Gsma intelligence. `https://www.gsmaintelligence.com/`.

[Banerjee et al., 2017] Banerjee, U., Juvekar, C., Fuller, S. H., and Chandrakasan, A. P. (2017). eedtls: Energy-efficient datagram transport layer security for the internet of things. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*, pages 1–6. IEEE.

[Bormann and Shelby, 2016] Bormann, C. and Shelby, Z. (2016). Block-wise transfers in the constrained application protocol (coap). *RFC*, 7959:1–37.

[Chung and Roedig, 2007] Chung, A. and Roedig, U. (2007). Efficient key establishment for wireless sensor networks using elliptic curve diffie-hellman. In *Proceedings of the 2nd European Conference on Smart Sensing and Context (EUROSSC2007).*

[Cisco Systems, 2018] Cisco Systems, T. (2018). About cisco - cisco. `https://www.cisco.com/c/en/us/about.html`.

[Commission, 2018] Commission, E. (2018). Ict research innovation. `https://ec.europa.eu/programmes/horizon2020/en/area/ict-research-innovation`.

[Contiki OS, 2018] Contiki OS, T. (2018). Contiki: The open source operating system for the internet of things. `http://www.contiki-os.org/`.

[Dener, 2014] Dener, M. (2014). Security analysis in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 10(10):303501.

[DESIGN Technologies, 2019] DESIGN Technologies, T. B. (2019). Weather station open message format - barani design technologies. `https://www.baranidesign.com/meteohelix-message-decoder`.

[Diaz and Sanchez, 2016] Diaz, A. and Sanchez, P. (2016). Simulation of attacks for security in wireless sensor network. *Sensors*, 16(11):1932.

[Dr. Muhammad, 2017] Dr. Muhammad, M. (2017). Security challenges in next generation 5g mobile networks. `https://www.informationsecuritybuzz.com/articles/security-challenges-next-generation-5g-mobile-networks/`.

[Dragomir et al., 2016] Dragomir, D., Gheorghe, L., Costea, S., and Radovici, A. (2016). A survey on secure communication protocols for iot systems. In *Secure Internet of Things (SIoT), 2016 International Workshop on*, pages 47–62. IEEE.

[Dujovne et al., 2014] Dujovne, D., Watteyne, T., Vilajosana, X., and Thubert, P. (2014). 6tisch: deterministic ip-enabled industrial internet (of things). *IEEE Communications Magazine*, 52(12):36–41.

[Dunkels et al., 2011] Dunkels, A., Eriksson, J., Finne, N., and Tsiftes, N. (2011). Powertrace: Network-level power profiling for low-power wireless networks.

[Electronics, 2018] Electronics, T. R. (2018). Battery electronics 101. `https://www.rc-electronics-usa.com/battery-electronics-101.html`.

[Elena, 2014] Elena, N. (2014). The latest update on 5g from ieee communications society. In *IEEE Communications Society*, `https://www.comsoc.org/blog/latest-update-5g-ieee-communications-society`.

[Elhoseny et al., 2016] Elhoseny, M., Yuan, X., El-Minir, H. K., and Riad, A. M. (2016). An energy efficient encryption method for secure dynamic wsn. *Security and Communication Networks*, 9(13):2024–2031.

[Engineers and Electronics, 2018] Engineers, I. o. E. and Electronics (2018). Ieee - about ieee. `https://www.ieee.org/about/index.html`.

[Fambon et al., 2014] Fambon, O., Fleury, E., Harter, G., Pissard-Gibollet, R., and Saint-Marcel, F. (2014). Fit iot-lab tutorial: hands-on practice with a very large scale testbed tool for the internet of things. *10èmes journées francophones Mobilité et Ubiquité, UbiMob2014*.

[Fielding et al., 1999] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (1999). Hypertext transfer protocol–http/1.1.

[Florian Kauer, 2018] Florian Kauer, T. (2018). Github - koalo/iotlab_energy: Energy consumption experiment for the iot-lab. `https://github.com/koalo/iotlab_energy`.

[Force, 2018] Force, I. E. T. (2018). Ietf — internet engineering task force. `https://www.ietf.org/`.

[Forum, 2018] Forum, N. F. C. (2018). What is nfc? - nfc forum — nfc forum. `https://nfc-forum.org/what-is-nfc/`.

[FreeRTOS OS, 2017] FreeRTOS OS, T. (2017). Freertos - market leading rtos (real time operation system) for embedded systems with internet of things extensions. `https://www.freertos.org/`.

[Gai et al., 2016] Gai, K., Qiu, M., Tao, L., and Zhu, Y. (2016). Intrusion detection techniques for mobile cloud computing in heterogeneous 5g. *Security and Communication Networks*, 9(16):3049–3058.

[Ge et al., 2014] Ge, X., Cheng, H., Guizani, M., and Han, T. (2014). 5g wireless backhaul networks: challenges and research advances. *IEEE Network*, 28(6):6–11.

[GitHub-kokke, 2019] GitHub-kokke, T. (2019). Tiny aes in c: Small portable aes128/192/256 in c. `https://github.com/kokke/tiny-AES-c`.

[Gomez et al., 2012] Gomez, C., Oller, J., and Paradells, J. (2012). Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors*, 12(9):11734–11753.

[Google, 2018] Google, T. (2018). About us — nest. `https://nest.com/about/`.

[Granjal et al., 2013] Granjal, J., Monteiro, E., and Silva, J. S. (2013). Application-layer security for the wot: extending coap to support end-to-end message security for internet-integrated sensing applications. In *International Conference on Wired/Wireless Internet Communication*, pages 140–153. Springer.

[Granjal et al., 2015] Granjal, J., Monteiro, E., and Silva, J. S. (2015). Security for the internet of things: a survey of existing protocols and open research issues. *IEEE Communications Surveys & Tutorials*, 17(3):1294–1312.

[Granjal and Pedroso, 2018] Granjal, J. and Pedroso, A. (2018). An intrusion detection and prevention framework for internet-integrated coap wsn. *Security and Communication Networks*, 2018.

[Group, 2018a] Group, F. (2018a). Wirelesshart security. `https://fieldcommgroup.org/wirelesshart-security`.

[Group, 2015] Group, T. (2015). Thread stack fundamentals. `https://www.threadgroup.org/support`.

[Group, 2018b] Group, T. (2018b). What is thread. `https://www.threadgroup.org/What-is-Thread`.

[Hall, 2013] Hall, J. (2013). C implementation of cryptographic algorithms. *Texas Instruments*.

[IEEE, 2012a] IEEE, T. (2012a). 802.15.6-2012 - ieee standard for local and metropolitan area networks - part 15.6: Wireless body area networks. `https://ieeexplore.ieee.org/document/6161600/`.

[IEEE, 2012b] IEEE, T. (2012b). 802.15.6-2012 - ieee standard for local and metropolitan area networks - part 15.6: Wireless body area networks. `http://www.ieee802.org/15/pub/TG6.html`.

[IEEE, 2015] IEEE, T. (2015). 802.15.4-2015 - ieee standard for low-rate wireless networks. `https://ieeexplore.ieee.org/document/7460875/`.

[IEEE, 2016] IEEE, T. (2016). 802.11-2016 - ieee standard for information technology–telecommunications and information exchange between systems local and metropolitan area networks. `https://ieeexplore.ieee.org/document/7786995/`.

[Institute of Technology, 2019a] Institute of Technology, T. R. (2019a). `https://www.kth.se/en`.

[Institute of Technology, 2019b] Institute of Technology, T. R. (2019b). Scada and central applications an introduction. `https://www.kth.se/social/upload/535629dcf2765437a2fd88f3/Lecture%209%20-%20SCADA%20System.pdf`.

[Intel, 2017] Intel, T. (2017). Different wi-fi protocols and data rates. `https://www.intel.com/content/www/us/en/support/articles/000005725/network-and-i-o/wireless-networking.html`.

[IoT-LAB, 2018] IoT-LAB, T. (2018). Iot-lab: a very large scale open testbed. `https://www.iot-lab.info/`.

[ISA, 2018] ISA, T. (2018). About isa - the home for automation professionals - isa. `https://www.isa.org/about-isa/`.

[ISA, 2009] ISA, T. I. S. o. A. (2009). Wireless systems for industrial automation: Process control and related applications isa 100.11a. `https://www.isa.org/isa100/`.

[Islam et al., 2012] Islam, K., Shen, W., and Wang, X. (2012). Security and privacy considerations for wireless sensor networks in smart home environments. In *Proceedings of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 626–633. IEEE.

[Kent and Seo, 2005] Kent, S. and Seo, K. (2005). Security architecture for the internet protocol. Technical report.

[Kim et al., 2008] Kim, A. N., Hekland, F., Petersen, S., and Doyle, P. (2008). When hart goes wireless: Understanding and implementing the wirelesshart standard. In *Emerging Technologies and Factory Automation, 2008. ETFA 2008. IEEE International Conference on*, pages 899–907. IEEE.

[Kumar and Singh, 2016] Kumar, S. and Singh, R. (2016). Secure authentication approach using diffie-hellman key exchange algorithm for wsn. *International Journal of Communication Networks and Distributed Systems*, 17(2):189–201.

[Laboratories, 2018] Laboratories, T. S. (2018). About us — silicon labs. `https://www.silabs.com/about-us`.

[Labs, 2018] Labs, S. (2018). Z-wave specification. `https://www.silabs.com/products/wireless/mesh-networking/z-wave/specification`.

[Maerien et al., 2015] Maerien, J., Michiels, S., Hughes, D., Huygens, C., and Joosen, W. (2015). Seclooci: A comprehensive security middleware architecture for shared wireless sensor networks. *Ad Hoc Networks*, 25:141–169.

[Mattsson et al., 2018] Mattsson, J., Fornehed, J., Selander, G., Palombini, F., and Amsüss, C. (2018). Controlling Actuators with CoAP. Internet-Draft draft-mattsson-core-coap-actuators-06, Internet Engineering Task Force. Work in Progress.

[Montenegro et al., 2007] Montenegro, G., Kushalnagar, N., Hui, J., and Culler, D. (2007). Transmission of ipv6 packets over ieee 802.15. 4 networks. Technical report.

[Muradore and Quaglia, 2015] Muradore, R. and Quaglia, D. (2015). Energy-efficient intrusion detection and mitigation for networked control systems security. *IEEE Transactions on Industrial Informatics*, 11(3):830–840.

[Networks, 2018] Networks, N. G. M. (2018). Ngmn - vision and mission. `https://www.ngmn.org/about-us/vision-mission.html`.

[Nguyen et al., 2015] Nguyen, K. T., Laurent, M., and Oualha, N. (2015). Survey on secure communication protocols for the internet of things. *Ad Hoc Networks*, 32:17–31.

[Nixon and Round Rock, 2012] Nixon, M. and Round Rock, T. (2012). A comparison of wirelesshart and isa100. 11a.

[Nunoo-Mensah et al., 2015] Nunoo-Mensah, H., Boateng, K. O., and Gadze, J. D. (2015). Comparative analysis of energy usage of hash functions in secured wireless sensor networks. *International Journal of Computer Applications*, 109(11).

[OASIS, 2012] OASIS, T. (2012). Advanced message queueing protocol (amqp) v1.0. `https://www.oasis-open.org/standards#amqpv1.0`.

[of Miami, 2019] of Miami, T. U. (2019). Weatherstem. `https://miamidade.weatherstem.com/umiami`.

[Olsson, 2014] Olsson, J. (2014). 6lowpan demystified. *Texas Instruments*, page 13.

[OpenWSN OS, 2018] OpenWSN OS, T. (2018). Openwsn - confluence. `https://openwsn.atlassian.net/wiki/spaces/OW/overview`.

[Packetizer, 2019] Packetizer, T. (2019). Secure hashing algorithm (sha-1) - source code. `https://www.packetizer.com/security/sha1/`.

[Rault et al., 2014] Rault, T., Bouabdallah, A., and Challal, Y. (2014). Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, 67:104–122.

[Red Hat, 2018] Red Hat, T. (2018). About. `https://www.redhat.com/en/about`.

[Rescorla and Modadugu, 2012] Rescorla, E. and Modadugu, N. (2012). Datagram transport layer security version 1.2.

[RIOT OS, 2018] RIOT OS, T. (2018). Riot - the friendly operating system for the internet of things. `https://riot-os.org/`.

[Ritchie, 1993] Ritchie, D. M. (1993). The development of the c language. *ACM Sigplan Notices*, 28(3):201–208.

[Saint-Andre, 2011] Saint-Andre, P. (2011). Extensible messaging and presence protocol (xmpp): Core.

[Samsung, 2018] Samsung, T. (2018). About us — samsung. `https://www.samsung.com/pt/aboutsamsung/home/`.

[Selander et al., 2019] Selander, G., Mattsson, J., Palombini, F., and Seitz, L. (2019). Object Security for Constrained RESTful Environments (OSCORE). RFC 8613.

[Shelby et al., 2014] Shelby, Z., Hartke, K., and Bormann, C. (2014). The constrained application protocol (coap).

[Sicari et al., 2015] Sicari, S., Rizzardi, A., Grieco, L. A., and Coen-Porisini, A. (2015). Security, privacy and trust in internet of things: The road ahead. *Computer networks*, 76:146–164.

[SIG, 2018] SIG, B. (2018). About us — bluetooth technology website. `https://www.bluetooth.com/about-us`.

[Singh et al., 2015] Singh, M., Rajan, M., Shivraj, V., and Balamuralidhar, P. (2015). Secure mqtt for internet of things (iot). In *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on*, pages 746–751. IEEE.

[Snort, 2019] Snort, T. (2019). Snort - network intrusion detection & prevention system. `https://www.snort.org/`.

[Standard, 2014] Standard, O. (2014). Mqtt version 3.1. 1. *URL http://docs. oasis-open. org/mqtt/mqtt/v3*, 1.

[Standardization, 2018] Standardization, I. O. f. (2018). Iso - international organization for standardization. `https://www.iso.org/home.html`.

[Standards, 2018] Standards, O. f. t. A. o. S. I. (2018). Oasis — organization for the advancement of structured information standards. `https://www.oasis-open.org/`.

[Stanford-Clark and Truong, 2013] Stanford-Clark, A. and Truong, H. L. (2013). Mqtt for sensor networks (mqtt-sn) protocol specification. *International business machines (IBM) Corporation version*, 1.

[Taddeo et al., 2010] Taddeo, A. V., Micconi, L., and Ferrante, A. (2010). Gradual adaptation of security for sensor networks. In *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*, pages 1–9. IEEE.

[Taddeo et al., 2011a] Taddeo, A. V., Morales, L. G. G., and Ferrante, A. (2011a). A framework for security and workload gradual adaptation. In *Security and Cryptography (SECRYPT), 2011 Proceedings of the International Conference on*, pages 178–187. IEEE.

[Taddeo et al., 2011b] Taddeo, A. V., Morales, L. G. G., and Ferrante, A. (2011b). System policies for gradual tuning of security and workload in wireless sensor networks. In *Wireless Telecommunications Symposium (WTS), 2011*, pages 1–6. IEEE.

[TinyOS, 2018] TinyOS, T. (2018). Github - tinyos/tinyos-main: Main development repository for tinyos (an os for embedded, wireless devices). `https://github.com/tinyos/tinyos-main`.

[Tsikoudis et al., 2016] Tsikoudis, N., Papadogiannakis, A., and Markatos, E. P. (2016). Leonids: A low-latency and energy-efficient network-level intrusion detection system. *IEEE Transactions on Emerging Topics in Computing*, 4(1):142–155.

[Van Rossum et al., 2007] Van Rossum, G. et al. (2007). Python programming language. In *USENIX annual technical conference*, volume 41, page 36.

[Vilajosana et al., 2017] Vilajosana, X., Watteyne, T., and Pister, K. (2017). Minimal ipv6 over the tsch mode of ieee 802.15. 4e (6tisch) configuration.

[Vinoski, 2006] Vinoski, S. (2006). Advanced message queuing protocol. *IEEE Internet Computing*, 10(6).

[Winter, 2012] Winter, T. (2012). Rpl: Ipv6 routing protocol for low-power and lossy networks.

[Yang et al., 2015] Yang, N., Wang, L., Geraci, G., Elkashlan, M., Yuan, J., and Di Renzo, M. (2015). Safeguarding 5g wireless communication networks using physical layer security. *IEEE Communications Magazine*, 53(4):20–27.

[Zimmermann et al., 2017] Zimmermann, L., Mars, N., Schappacher, M., and Sikora, A. (2017). Development of thread-compatible open source stack. In *Journal of Physics: Conference Series*, volume 870, page 012001. IOP Publishing.

This page is intentionally left blank.

Appendix A

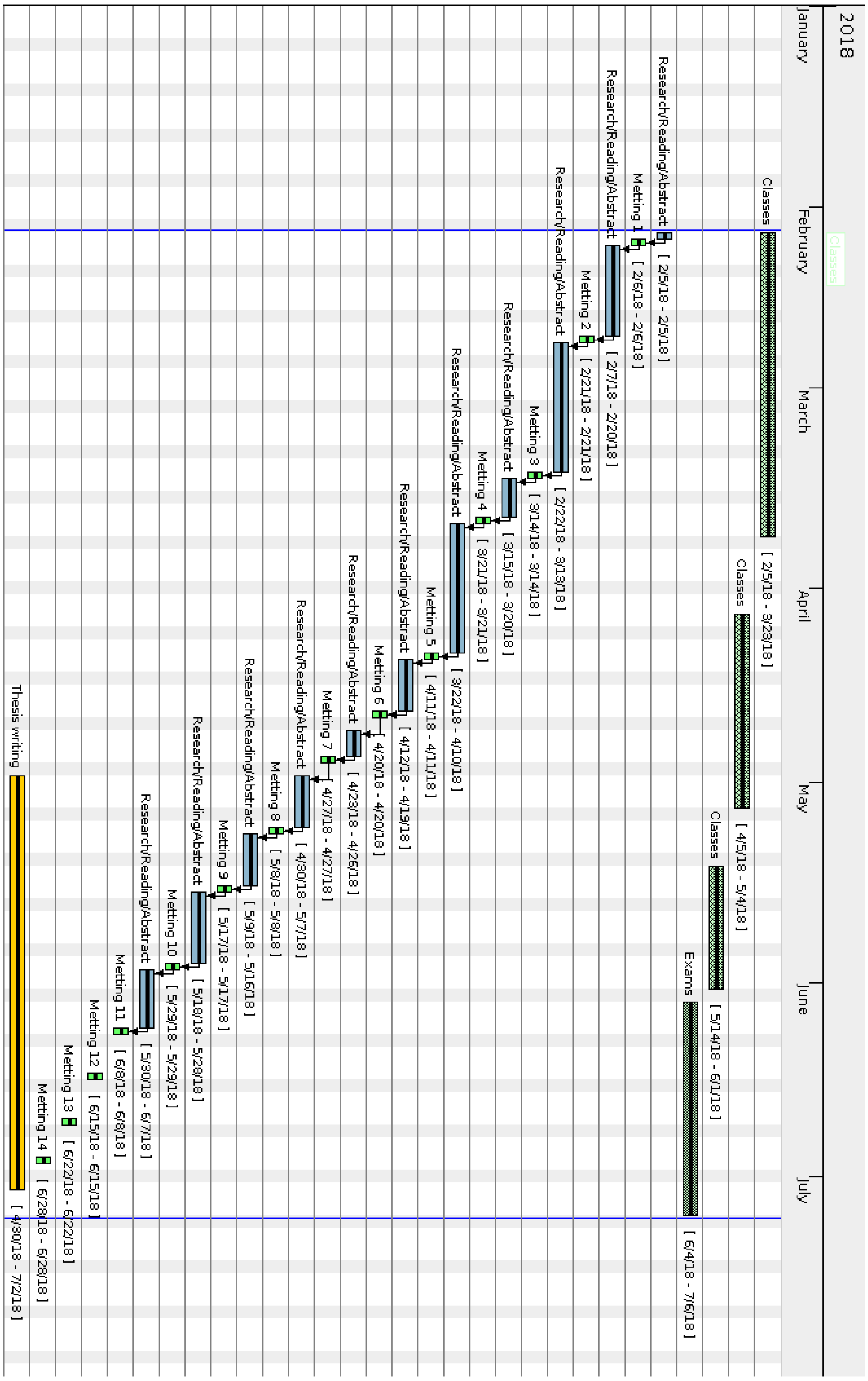

Figure 6.1: Work plan - first semester

This page is intentionally left blank.

Appendix B

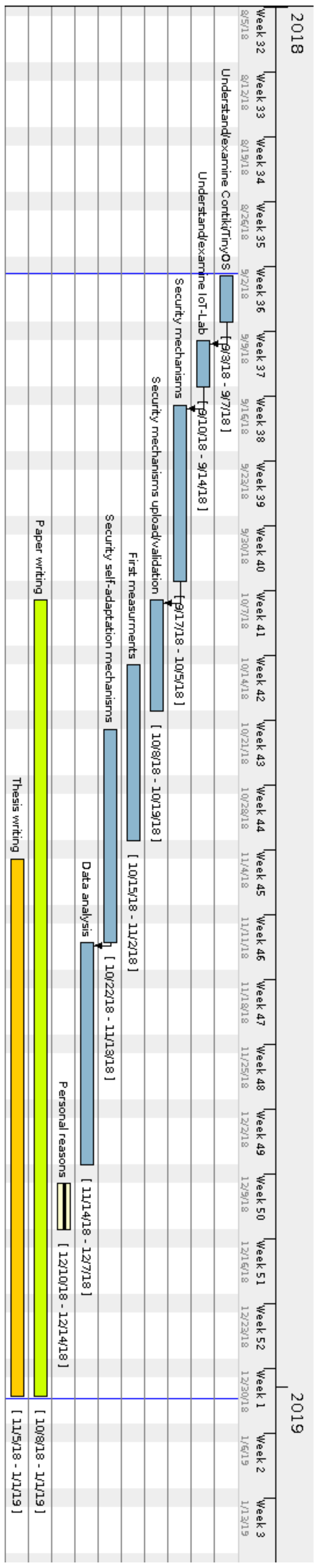

Figure 6.2: Work plan - second semester

This page is intentionally left blank.

Appendix C

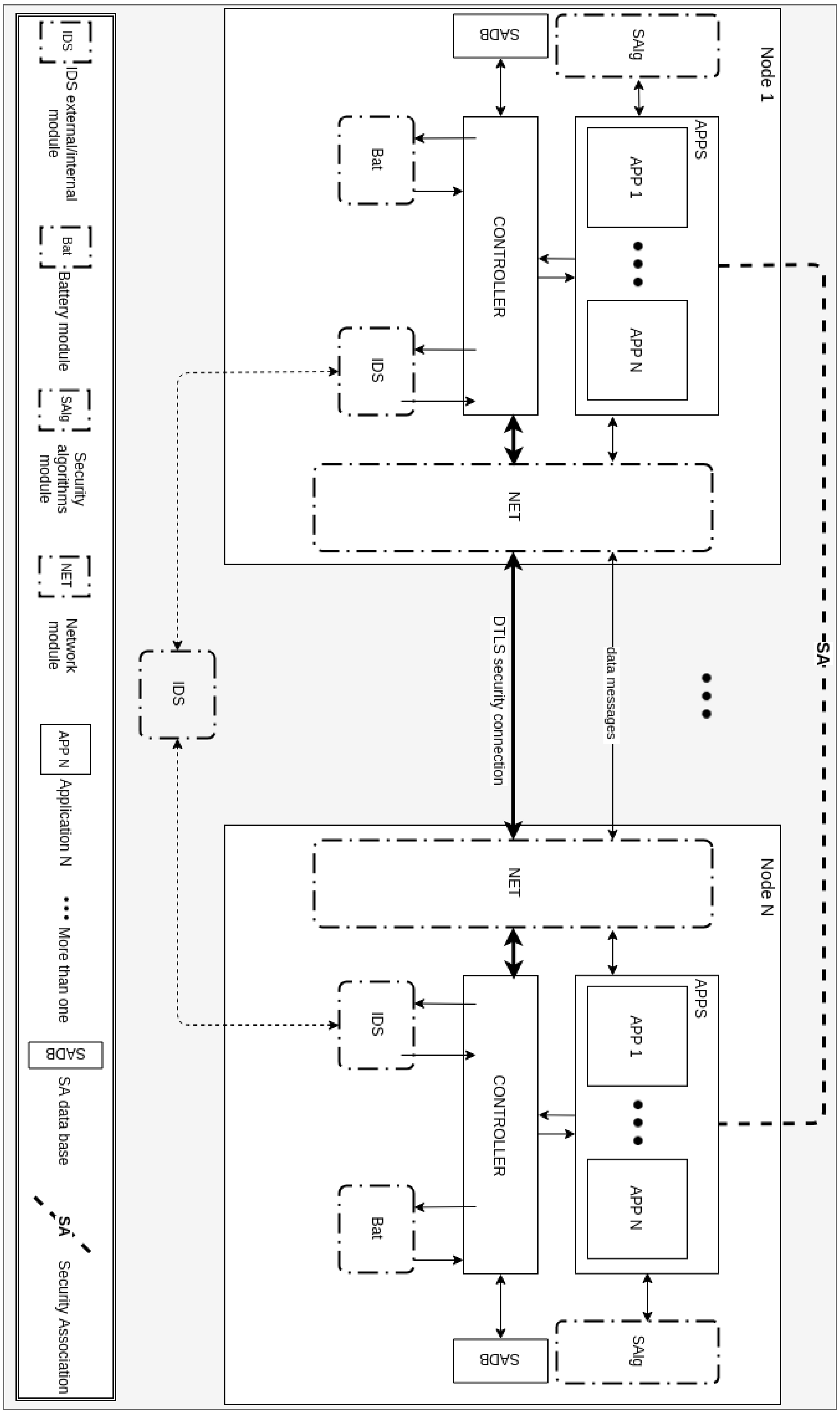

Figure 6.3: Architecture proposed

This page is intentionally left blank.

Appendix D - The used scripts to calculate the energy results

Listing 6.1: The initial Python script to calculate the total consumption of an experiment

```
#!/usr/bin/env python3

import pandas as pd
import numpy as np
import argparse

parser = argparse.ArgumentParser()
parser.add_argument('file',nargs='+')
args = parser.parse_args()

for filename in args.file:
  print(filename)

  df = pd.read_csv(filename,
          skiprows=9,
          sep='\t',
          header=None,
          usecols =[3,4,5,6,7],
          names=['time_s','time_us','power','voltage','current'])

  df['time'] = df['time_s']+df['time_us']/1000000
  df['difftime'] = df['time'] - df['time'].shift(1)
  df = df.iloc[1:]

  df['energy'] = df['difftime']*df['power']

  duration = df.iloc[-1]['time'] - df.iloc[0]['time']
  print("Total duration: %.0f s"%duration)
  print("Average voltage: %.2f V"%df['voltage'].mean())
  print("Average current: %i mA"%round(df['current'].mean()*1000))
  print("Average power: %i mW"%round(df['power'].mean()*1000))
  print("Total energy consumption: %i Ws"%df['energy'].sum())
```

Listing 6.2: The Python functions used to calculate the total consumption of the experiments

```
...
def read_file(filename, i):
    df = pd.read_csv(filename,
          skiprows=9,
          sep='\t',
          header=None,
          usecols=[0,  3,4,5,6,7],
          names=['real_time','time_s','time_us','power','voltage','current'])

    init_time=df['real_time'][0]  + 30#seconds
    #print init_time
    df = df[ df['real_time'] >= init_time ]
    #print df[0]

    if df.isnull().any().any():
        print("HAS_NAN: ",i)

    return df
```

```
def energy(df, i):
    #df = df_panda
    df['time'] = df['time_s']+df['time_us']/1000000
    df['difftime'] = df['time'] - df['time'].shift(1)
    df = df.iloc[1:]

    #df['energy'] = df['difftime']* df['power']
    energy = df['difftime']*df['power']

    duration = df.iloc[-1]['time'] - df.iloc[0]['time']
    print("Total energy consumption: %.3f Ws - " % energy.sum(), i)
    """print("Number of values %d"%len(df.index))
    print("Total duration: %.0f s"%duration)
    print("Average voltage: %.2f V"%df['voltage'].mean())
    print("Average current: %i mA"%round(df['current'].mean()*1000))
    print("Average power: %i mW"%round(df['power'].mean()*1000))
    #print("Total energy consumption: %i Ws"%energy.sum())
    print("Total energy consumption: %.3f Ws" % energy.sum())
    ###########
    #en = df['energy']
    #pw = df['power']
    print("Energy consumption size: ",energy.size)
    print("Standard Total energy consumption: ",energy.std())
    print("Standard power: ",df['power'].std()) """
    print("#####################################")

    return energy

...

def read_all_files(path_dir):
    files = []
    energy_val = []
    energy_tot = []
    for i in range(1,32):
        file = path_dir+str(i)+"/m3_12.oml"
        #print file
        df = read_file(file, i)
        e = energy(df, i)
        #print energy(e)
        energy_val.append(e)
        energy_tot.append(e.sum())
        #print df.size
        #print e.size
        files.append(df)
    all_files = pd.concat(files) #merge all samples
    all_energy = pd.concat(energy_val) #merge all samples
    return all_files, all_energy, energy_tot

...

if __name__ == '__main__':
    ...
    all_files2_1 , all_energy2_1, energy2_tot_1 = read_all_files("128_bits/just_encrypt/1
    _12_ms_4average/") #AES
    all_files2_2 , all_energy2_2, energy2_tot_2 = read_all_files("192_bits/just_encrypt/1
    _12_ms_4average/")
    all_files2_3 , all_energy2_3, energy2_tot_3 = read_all_files("256_bits/just_encrypt/1
_12_ms_4average/")
 all_files2_4  , all_energy2_4 , energy2_tot_4 = read_all_files ("128_bits/1_12_ms_4average/")
 ##
 all_files2_5  , all_energy2_5 , energy2_tot_5 = read_all_files ("192_bits/1_12_ms_4average/")
 all_files2_6  , all_energy2_6 , energy2_tot_6 = read_all_files ("256_bits/1_12_ms_4average/")
 ...
power_mean = [all_files2_1 ['power'].mean(), all_files2_4 ['power'].mean(), all_files2_2 ['
 power'].mean(), all_files2_5 ['power'].mean(), all_files2_3 ['power'].mean(), all_files2_6 ['
 power'].mean()]
power_std = [ all_files2_1 ['power'].std() , all_files2_4 ['power'].std() , all_files2_2 ['power
 '].std() , all_files2_5 ['power'].std() , all_files2_3 ['power'].std() , all_files2_6 ['power'].
 std() ]
energy_mean = [all_energy2_1.mean(), all_energy2_4.mean(), all_energy2_2 .mean(),
 all_energy2_5 .mean(), all_energy2_3 .mean(), all_energy2_6 .mean()]
energy_std = [all_energy2_1 .std() , all_energy2_4 .std() , all_energy2_2 .std() , all_energy2_5
 .std() , all_energy2_3 .std() , all_energy2_6 .std() ] #in mJ -> *1000
total_energy_mean = [energy2_tot_1.mean(), energy2_tot_4.mean(), energy2_tot_2.mean(),
 energy2_tot_5.mean(), energy2_tot_3.mean(), energy2_tot_6.mean()] #in J
 total_energy_std  = [energy2_tot_1.std() , energy2_tot_4.std() , energy2_tot_2.std() ,
 energy2_tot_5.std() , energy2_tot_3.std() , energy2_tot_6.std() ] #in J
print power_mean
print power_std
print energy_mean
print energy_std
print total_energy_mean
print total_energy_std
 ...
```

This page is intentionally left blank.

Appendix E - The used script to launch experiments

Listing 6.3: The used script to perform experiments at IoT-LAB, in a automated manner

```
#!/bin/bash
get_results2 ()
{
    #get the status  files
    #echo $experiment_directory VS $msg_num$EXP_NAME
    mkdir -p $experiment_directory$iteration;
    echo $experiment_directory$iteration
    #echo GET THE .TXT FILES
    #scp -i ~/.ssh/IOT_LAB_id_rsa lsilva@$LOCAL.iot-lab.info:*.txt
    $$experiment_directory$iteration.
    #sleep 1
    #remove the copied status  files ...
    #echo REMOVE THE .TXT FILES FROM REMOTE SERVER
    #ssh -i ~/.ssh/IOT_LAB_id_rsa lsilva@$LOCAL.iot-lab.info  "rm *.txt"
    sleep  1
    #put the consumption files  in  the  right  folders ..
    echo GET THE .OML FILES FROM LAST EXPERIMENT
    scp -i ~/.ssh/IOT_LAB_id_rsa lsilva@$LOCAL.iot-lab.info:~/.iot-lab/last/consumption/*.
    oml $experiment_directory$iteration'/'.
    #scp -i ~/.ssh/IOT_LAB_id_rsa lsilva@grenoble.iot-lab.info:~/.iot-lab/141064/
    consumption/m3-357.oml m3-357.oml
    sleep  1
    process_id=$! #get the id  of  the  last  job/process
    wait $process_id
}

run_experiment2()
{
    #echo "Function name:  ${FUNCNAME}"
    #echo "The number of positional parameter : $#"
    #echo "All parameters or arguments passed to the function: '$@'"
    #result=$(($1 + $2))
    #echo "Result is:  $result"

    if  [ "$#" -eq 3 ]; then
        #statements
        echo "The number of positional parameter : $#"
        #border_router_firmware=$BASE'/'$i'/'$key'/firmware/border-router.iotlab-m3'
        #nodes_firmware=$BASE'/'$i'/'$key'/firmware/'
        #experiment_directory=$BASE'/'$i'/'$key'/'$j'/'
    else
        #border_router_firmware=$BASE'/firmware/border-router.iotlab-m3'
        nodes_firmware=$BASE'/hello-world.iotlab-m3'
        #mkdir -p /home/mlzboy/b2c2/shared/db;

        experiment_directory=$BASE'/'$j''average'/'
        #echo $experiment_directory
    fi

    for  iteration  in 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27
    28 29 30 31
    do
        #echo $experiment_directory$msg_num$EXP_NAME #directory
        #echo $nodes_firmware$msg_num$EXP_NAME #firmware_directory
        #echo $border_router_firmware

        #launch an experiment
        echo EXPERIMENT_SUBMIT submit —n $j$EXP_NAME —d $EXP_TIME —l
    $LOCAL,m3,$NODE,$nodes_firmware,$PROFILE
        #iotlab—experiment stop
        iotlab—experiment submit —n $j$EXP_NAME —d $EXP_TIME —l $LOCAL,m3,
    $NODE,$nodes_firmware,$PROFILE
        iotlab—experiment wait #wait to experiment — status running
        process_id=$! #get the id of the last job/process
        wait $process_id
        echo Job 'iotlab—experiment wait' exited with status $?
        echo SSH CONNECTION
        ssh —i ~/.ssh/IOT_LAB_id_rsa lsilva@$LOCAL.iot—lab.info "nc_m3—12_20000"
        #sudo tunslip6.py —v2 —L —a m3—12 —p 20000 2001:660:5307:3171::1/64 &"
        #nc m3—357 20000 > m3—357_$msg_num.txt &
        #nc m3—358 20000 > m3—358_$msg_num.txt &
        #hostname

        echo Lets WAIT TO THE EXPERIMENT FINISHED
        sleep 3 #because sudo tunslip6.py waits to the experiment to finished ...
        #sleep $CLOSE_WINDOW_DELAY #to wait for experiment to finished and get results
        get_results2 $experiment_directory $iteration

    done
}

#ssh —i ~/.ssh/IOT_LAB_id_rsa lsilva@grenoble.iot—lab.info
#Enter passphrase for key '/home/lfgsilva/.ssh/IOT_LAB_id_rsa':
#remote server commands
#sudo tunslip6.py —v2 —L —a m3—356 —p 20000 2001:660:5307:3171::1/64
#nc m3—357 20000 > m3—357_5.txt
#nc m3—358 20000 > m3—358_5.txt
###############################
#get the *.txt files
#scp —i ~/.ssh/IOT_LAB_id_rsa lsilva@grenoble.iot—lab.info:*.txt .
#Enter passphrase for key '/home/lfgsilva/.ssh/IOT_LAB_id_rsa':


BASE=$(pwd)
PASS=
USER=

#iotlab—auth —u ${USER} —p ${PASS}
#vars globais
#PORT=20000
NODE=12
#=356
#COAP_SERVER=357
#COAP_CLIENT=358
EXP_TIME=10    #The experiment time
#CLOSE_WINDOW_DELAY=$(($EXP_TIME*60+60)) #10m*60s+60s=11min
#echo $CLOSE_WINDOW_DELAY seconds to wait per experiment
EXP_NAME=without_sec
LOCAL=lyon
#LOCAL=grenoble
#for i in AES #3DES
#do
```

```
#    if [ "$i" = AES ];
#    then
         #statements
#         echo AES
#         for key in 256 #128
#         do
              for j in '1_12_ms_4' #'4_48_ms_16' '17_92_ms_64' '65_95_ms_4'
              do
                   #echo $j

                   if [ "$j" = "1_12_ms_4" ]; then
                        PROFILE=consumption_thesis2
                        run_experiment2 $j #$i $key
                   elif [ "$j" = "4_48_ms_16" ]; then
                        PROFILE=consumption_thesis3
                        run_experiment2 $j #$i $key
                   elif [ "$j" = "17_92_ms_64" ]; then
                        PROFILE=consumption_thesis4
                        run_experiment2 $j #$i $key
                   else
                        PROFILE=consumption_thesis
                        run_experiment2 $j #$i $key
                   fi

                   #sleep $CLOSE_WINDOW_DELAY
                   #get_results $i $j
              done

#         done
     #elif [[ condition ]]; then
     #         #statements
#    else
         #statements
#         echo 3DES
#         for j in 1 2 3 4 5 # number of runs
#         do
              #echo $j
#              run_experiment2 $i $j
              #sleep $CLOSE_WINDOW_DELAY
              #get_results $i $j
#         done
#    fi
#done
exit
```

This page is intentionally left blank.

Appendix F - Energy measurements results

Table 6.1: Energy measurements without cutting

Configurations\Measurements	Total consumption average	Standard deviation	Energy samples average	Standard deviation	Power samples average	Standard deviation
Standard configuration	86.4236980454004	0.437513897341713	0.000159304717092	7.82E-06	0.142980851916706	0.003595063764992
AES 128 ENCRIPT	87.7262385596088	0.019001348496493	0.000161579712137	7.58E-06	0.145010982473993	0.003746933555052
AES 128 DECRYPT	87.2158725810613	0.306708164689261	0.000160747812773	7.81E-06	0.144264305804247	0.002560225085746
AES 192 ENCRYPT	87.419313210866	0.144228837248718	0.000161119268717	7.59E-06	0.144600751418635	0.003299067470024
AES 192 DECRYPT	87.2400661788289	0.088953658801373	0.000160707709735	8.06E-06	0.144228837248718	0.002689116195886
AES 256 ENCRYPT	87.5227553766881	0.075393622793752	0.000161196993816	7.31E-06	0.144672073116447	0.002946422431241
AES 256 DECRYPT	87.1893333188138	0.03052353259471	0.000160586646262	7.61E-06	0.14412810269444	0.002620372480595
3DES 56 ENCRYPT	86.7024881251371	0.066973450121145	0.000159689236794	7.37E-06	0.14331872386417	0.003169564427113
3DES 56 DECRYPT	87.2698318199177	0.016056072379101	0.000160723641649	6.56E-06	0.144252747987219	0.003122378931896
3DES 112 ENCRYPT	86.8997951612195	0.125114482938366	0.000160065438689	7.25E-06	0.143653053294053	0.003776387387372
3DES 112 DECRYPT	87.2294488525273	0.046702755733292	0.00016065772537	7.37E-06	0.144196907231278	0.003155340673768
3DES 168 ENCRYPT	86.6611020966421	0.067127791596818	0.000159614377303	7.56E-06	0.143253915942758	0.00314617094618
3DES 168 DECRYPT	87.2472838596002	0.083456514894483	0.000160693313622	7.43E-06	0.144232743932703	0.003333546273274
SHA1	87.1223941104503	0.259574616636936	0.00016054012031	7.50E-06	0.144090439921492	0.002616832539343
SHA2 256	87.5758090647354	0.028284242429399	0.000161288698415	7.15E-06	0.144756448796184	0.002741903741716
SHA2 384	87.5912633115698	0.365511581888445	0.000161454709723	6.51E-06	0.144893205835189	0.003116489278544
SHA2 512	88.8775439977952	0.059621341963239	0.000163711435228	6.94E-06	0.146917537971844	0.002858545399831

Table 6.2: Energy measurements with cutting

Configurations\Measurements	**Total consumption average**	**Standard deviation**	**Energy samples average**	**Standard deviation**	**Power samples average**	**Standard deviation**
Standard configuration	82.1286126242181	0.435187814478352	0.000159296884695	7.93E-06	0.142973440339342	0.003552754081345
AES 128 ENCRIPT	83.3791180814303	0.01818813551298	0.000161586922955	7.67E-06	0.145017142652182	0.003693742071269
AES 128 DECRYPT	82.8894243824367	0.306655319161514	0.000160751868198	7.94E-06	0.14426753484917	0.002496388057211
AES 192 ENCRYPT	83.0826396096968	0.307140900861103	0.000161123486248	7.70E-06	0.144604185795927	0.00324129312713
AES 192 DECRYPT	82.9147174465398	0.088262637330594	0.000160711991807	8.20E-06	0.144232265603019	0.002629704608199
AES 256 ENCRYPT	83.1838113031866	0.071432880684714	0.000161201176888	7.42E-06	0.144675496349177	0.002887645894462
AES 256 DECRYPT	82.8646826038244	0.028096744470171	0.000160587368043	7.73E-06	0.144128421584062	0.00255889498226
3DES 56 ENCRYPT	82.4031505055396	0.06348221373228	0.000159691495732	7.47E-06	0.143320427755097	0.003120683572649
3DES 56 DECRYPT	82.941850992834	0.015501661700333	0.000160725271073	6.63E-06	0.144253969668196	0.003065703821985
3DES 112 ENCRYPT	82.5923920047385	0.12126700302674	0.000160070920847	7.34E-06	0.143657675491305	0.003736757278855
3DES 112 DECRYPT	82.9025181219642	0.045993262967897	0.000160659044139	7.47E-06	0.144197771687489	0.00310058898631
3DES 168 ENCRYPT	82.3635151663503	0.066712424695144	0.000159616660933	7.67E-06	0.143255605690683	0.003095829470482
3DES 168 DECRYPT	82.918748114498	0.083791268411989	0.000160693932273	7.53E-06	0.144232952495042	0.003279160392889
SHA1	82.7984354544857	0.259196537656949	0.000160540839151	7.63E-06	0.144090706346387	0.002552465927027
SHA2 256	83.2342258467539	0.026986600565374	0.000161292804921	7.26E-06	0.144759786860717	0.002681470602488
SHA2 384	83.246532362147	0.365375423881829	0.000161459497191	6.58E-06	0.144897262915052	0.003062528544608
SHA2 512	84.4733175065375	0.05692547995012	0.00016371892087	7.03E-06	0.146923970962419	0.002787026419244