Wireless Communication Protocols for Home Automation

Exploring the security and privacy aspects of smart home IoT devices communicating over the Z-Wave protocol

by

Vasileios Merdis

April 24, 2019

Supervisors:

Dr.ir. Marten J. van Sinderen (UT)

Dr. Anna Sperotto (UT)

Jeroen Slobbe (Deloitte)

Dominika Rusek (Deloitte)

UNIVERSITY OF TWENTE.

Deloitte.

A thesis submitted in fulfillment of requirements

for the degree of Master of Science

in

Computer Science

Cyber Security

Faculty of Electrical Engineering, Mathematics and

Computer Science

University of Twente

P.O Box 217

7500 AE Enschede

The Netherladns

Acknowledgement

I am thankful to many people for their support, contribution, dedication and friendship, that helped me to complete with success the master in Cyber Security. I will never forget those years as a student at the University of Twente and as an intern in Deloitte Netherlands.

I would like to thank my supervisors at the University, Marten van Sinderen and Anna Sperotto, for the perfect communication during the thesis period, their feedback and guidance when I needed it. Thank you also for the support and the immediate actions you took regarding the issue with the responsible disclosure.

Next, I want to thank my supervisors at Deloitte, Jeroen Slobbe and Dominika Rusek. Their help, support and motivation were driven me to complete my thesis successfully and deliver a valuable result. They helped me to understand and love the topic of IoT and the protocols in smart homes, and without their weekly pressure, I wouldn't be able to finish on time. Next, special thanks to Jilles Groenendijk, who welcomed me to his lab and introduced me to the area of hardware hacking. Many thanks to all the people in the Cyber Team of Deloitte, who made the last six months, the most exciting period of my life. Thank you and looking forward to continue working with all of you.

Last but not least, I am very thankful to my whole family, my parents Konstantinos and Vasiliki and my sister Agathi, for their unconditional love and continuous support. I dedicate this dissertation unto them. Finally, I want to thank God for everything and for bringing amazing people into my life. I want to thank all the people and friends who I met the last years in the Netherlands. Everything would be different without you.

Abstract

Smart home IoT devices are on rise nowadays since they bring a lot of benefits to the users. However, there are also risks concerning the security and the privacy of the inhabitants of a smart environment. Smart home ecosystem is very complex with numerous of smart devices and communication protocols which make up it. A communication protocol plays the role of the common language that smart appliances need to speak in order to be able to exchange information with multiple devices in a smart environment.

In this thesis, we examined and evaluated the security and privacy aspects of the Z-Wave protocol. Z-Wave is a wireless protocol for automation appliances connection at home, and it is the world market leader in wireless control with over 100 million products sold worldwide. It is a Radio Frequency based communication technology and uses the concept of the mesh networking with controllers and slaves.

In order to explore the security and privacy aspects of the Z-Wave protocol, hands-on testing on a Z-Wave light bulb and smart door lock were performed, including both software and hardware hacking. During the protocol hacking, we eavesdropped on the Z-Wave network, whereas during the hardware hacking we verifying whether the hardware of the devices contained any sensitive information that can be exploited.

The results showed that using no security on the Z-Wave devices, led to serious risks for the security and privacy of the users in a smart home environment. We captured some Z-Wave packets sent from the controller to the devices, and we performed a replay attack on the light bulb since the security was not enabled by default. The door lock was more secure but a crucial vulnerability was discovered while we performed hardware hacking. Z-Wave protocol supports three levels of security: no security, S0 security, and S2 security. After the experiments, it was observed that even the S0 framework can preserve the security and privacy of the users, and it is up to them to implement it or not.

Contents

List of Figures

List of Tables

List of Abbreviations

AES	Advanced Encryption Standard
AMI	Advanced Metering Infrastructure
BLE	Bluetooth Low Energy
CPU	Central Processing Unit
CBC	Cipher Block Chaining
CBC-MAC	Cipher Block Chaining Message Authentication Code
CMAC	Cipher-based Message Authentication Code
CTR	Counter
CCM	Counter with CBC-MAC
DoS	Denial of Service
DSK	Device Specific Key
DDoS	Distributed Denial of Service
DNS	Domain Name System
EEPROM	Electrically Erasable Programmable Read Only Memory
ECB	Electronic Codebook
ECDH	Elliptic Curve Diffie–Hellman
EoF	End of Frame
EU	European Union
FSK	Frequency Shift Keying
GFSK	Gaussian Frequency Shift Keying
GRC	GNU Radio Companion
HAN	Home Area Network
HMAC	Hash-based Message Authentication Code

HVAC	Heating, Ventilation and Air Conditioning
IoT	Internet of Things
IP	Internet Protocol
ISM	Industrial, Scientific and Medical
ISP	Internet Service Provider
IV	Initialization Vector
LED	Light-Emitting Diode
LPWAN	Low Power Wide Area Network
LTS	Long Term Support
MAC layer	Media Access Control layer
MNO	Mobile Network Operator
NFC	Near Field Communication
NIF	Node Information Frame
NRZ	Non-Return-to-Zero
OFB	Output Feedback
OpenHAB	Open Home Automation Bus
OWASP	Open Web Application Security Project
PCB	Printed Circuit Board
PHY layer	Physical layer
PRNG	Pseudo Random Number Generator
PUF	Physically Unclonable Function
QoS	Quality of Service
QR	Quick Response
RF	Radio Frequency
RFID	Radio Frequency Identification

S0	Security 0
S2	Security 2
SaaS	Security as a Service
SDN	Software Defined Networking
SDR	Software Defined Radio
SIS	Server ID SUC
SoF	Start of Frame
SPI	Serial Peripheral Interface
SUC	Static Update Controller
TCP	Transmission Control Protocol
TOR	The Onion Router
UI	User Interface
USB	Universal Serial Bus
VPN	Virtual Private Network
WAN	Wide Area Network
XOR	Exclusive OR

'Smart' in device names

means 'no privacy'

1 Introduction

The Internet of Things (IoT) is entering mainstream commercial markets and becomes an integral part of our lives. It is a system of interconnected electronic devices embedded with software, sensors, actuators and network connectivity which enables them to connect and transmit data (Ashton et al. 2009). IoT bring more convenience in the daily life in our homes, which are equipped with smart devices. Users have now the ability to access and control entirely their home's electronic systems via their smartphone or laptop. Evans 2011 predicts that there will 50 billion IoT devices connected to the internet in the world by 2020, using different communication protocols.

At the same time, IoT cause issues, related to people's security and privacy (Islam, Shen, and Wang 2012; Rehman, Gruhn, et al. 2018; Apthorpe et al. 2017). Since all these "things" connect to the Internet, they can be hacked as if it was a common laptop or computer. In some cases it is even easier for attackers to take control of the smart devices, as they lack security measures. Attackers have the knowledge and the power to take control over IoT devices and by creating botnets consisting of those devices, can perform distributed denial of service (DDoS) attacks. On October 21, 2016, the first IoT botnet on that scale, where multiple DDoS attacks targeted systems operated by Domain Name System (DNS) provider Dyn (Lewis 2017). These attacks were executed through a botnet consisting of numerous smart home devices, such as IP cameras and home printers, resulted in malfunction of many services with the most well known Amazon, Netflix and Twitter.

An attacker can also exploit users' privacy for a breakdown which can result in lethal consequences and physical loss of their personal belongings. Glisson et al. 2015 have shown that it is possible to hack into a domestic pacemaker and retrieve all the (medical) data stored in the device. It is also possible to re-configure the parameters of the device to cause a heart attack (Storm 2015). Furthermore, an attacker can monitor digitally the daily life of the inhabitants of a household and decides, based on their routine, if they are absent from home in order to conduct the burglary (Jacobsson, Boldt, and Carlsson 2016).

Wireless communication protocols play an important role in designing a smart home ecosystem. They can be classified by date rate, range, network topology, power consumption and security features. As a matter of fact, wireless connectivity is not monopolized by one single technology. Many IoT devices, applicable for a home environment, are usually served by radio frequency technologies that operate on unlicensed spectrum and that are designed for short-range connectivity with limited Quality of Service (QoS) and security requirements as described by Mahmoud and Mohamad 2016. The most common short range communication technologies for smart home applications, that are often mentioned in the literature, are Wi-Fi (Halow), Bluetooth, BLE, ZigBee and Z-Wave.

1.1 The Smart Home Concept

IoT technology enables smart home concept to improve our lives and provide sustainability and convenience to our living environment. Chan et al. 2009 define as a *Smart Home* a residence that is equipped with all the necessary technology and appliances that allows to monitor its inhabitants and provides independence and well-being to their everyday life. Smart homes are one of particular example where different components of home appliances communicate with each other. It is convenient, efficient and at the same time exciting to live in a smart environment full of comforts. With just one click from a smartphone or laptop or tablet, someone can handle the lights, the locks, can change the temperature and in general, has the ability to control the whole house, remotely from any room inside the house or from any location in the world, to satisfy his needs.

1.2 Smart Home Devices

The devices and the technology is what determines a home, smart and automated. According to the study made by Acar et al. 2018, smart devices can be categorized into three categories in terms of their capabilities namely, *Hub-like* devices, *User-controlled* devices and *Sensor-like* devices.

- *Hub-like*: Smart home hubs, in general, are network devices that connect other devices to each other and to the Internet, allowing the communication over a home automation network. These hubs have the capability of connecting a wide range of various smart sensors (e.g. thermostats and smart locks) by using wireless technologies. They are considered to be the brain of a smart home environment, as they bring together numerous devices and systems in a centralized platform. Some of the most popular smart home hub-like devices nowadays are the SamsungThings Hub[1] and the Wink Hub[2]. They both support Wi-Fi, ZigBee, Z-Wave and Bluetooth protocols.

- *User-controlled*: These devices can be configured and controlled by the users either locally (manually) or remotely via a smartphone or a laptop. Example of these devices are smart lights and smart locks.

- *Sensor-like*: These devices have built-in sensors which they can only use to perform and sense the environment. They are built to send notifications to their connected services either when an anomaly occurs (based on their configuration) or periodically. Temperature sensors are the most significant example of this kind of devices.

However, there are many criteria based on which home IoT devices can be distinguished. Many studies accept that this set of devices can be divided into constrained devices and powerful equipment (Bormann, Ersue, and Keranen 2014; Lévy-Bencheton et al. 2015). Constrained devices are the devices with limited resources, either characteristic (e.g. memory storage capacity) or processing which are used for a specific purpose. Powerful equipment refer to devices powered by the main supply which often receive more computational power or other features in order to complete supplementary tasks including security. Constrained devices are the user-controlled and sensor-like devices, whereas the smart hubs belong to powerful equipment.

[1] `https://www.samsung.com/us/smart-home/smartthings/hubs/samsung-smartthings-hub-f-hub-us-2/`
[2] `https://www.wink.com/products/wink-hub/`

1.3 Connectivity

The key role in a smart home environment is both the interconnection of the smart devices that are used and the continuous data processing and connectivity to the several network types within the smart home environment. Following the article of Lévy-Bencheton et al. 2015, the connectivity needs to be continuously present and active in the devices and also related to several communication protocols.

Home Area Network (HAN) is defined by Batalla, Vasilakos, and Gajewski 2017 as the set of all the smart devices and the communication rules that harmonically perform together for creating a vital smart home environment. Integral part of the smart home ecosystem is the Wide Area Network (WAN), an external system, which compromises high speed networks through the Internet Service Providers (ISP) and Mobile Network Operators (MNO), or Low Power WAN (LPWAN) based on communication through low power networking protocols (i.e. LoRaWAN, Sigfox). Figure 1.1 below shows a typical smart home environment as described by Lévy-Bencheton et al. 2015.

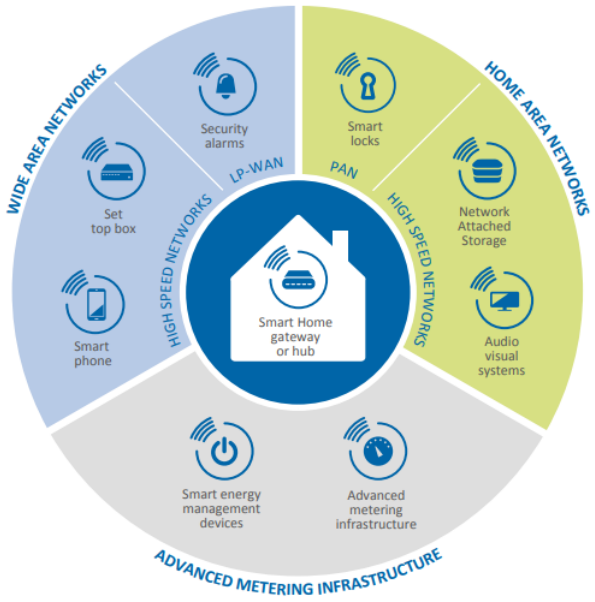

Figure 1.1: Smart Home Environment (Lévy-Bencheton et al. 2015)

The authors add another component to their smart home environment concept, in case of smart

meters, the Advanced Metering Infrastructure (AMI). It is an architecture for automated and two-way communication between a smart utility meter and the associated utility company.

Within a smart home environment, connectivity can be wired or wireless and the communication technologies can be categorized by their characteristics like different range, speed, different frequencies, energy consumption, different standards and security issues. Many IoT devices are using wireless connectivity for communication since it offers flexibility, mobility, convenience and cost efficiency needed for sensors and actuators networks. However, wireless communication has also some disadvantages, mainly regarding security since the transmission is via the air and anyone within the network range can intercept it.

1.3.1 Smart Home Wireless Protocols

In general, devices within a smart home ecosystem are capable to connect to peer devices or network via various wireless communication protocols. A communication protocol is a system of rules that allow two or more entities of a communication system to exchange messages via any kind of variation of a physical quantity[3]. In simple words, communication protocols play the role of the common language that smart appliances need to speak in order to be able to exchange information with multiple devices in a smart environment. The paper of Al-Sarawi et al. 2017 classify the IoT protocols in Low Power Wide Area Network (LPWAN) and Short Range Network. SigFox is an example of LPWAN, while Bluetooth, ZigBee and Z-Wave are some of the well-known communication protocols for short range networks.

Many of the most popular wireless technologies in the Internet of Things we often meet in different IoT appliances in a smart home ecosystem. The study of Ray and Bagwari 2017 describes Bluetooth, Bluetooth BLE, ZigBee, Z-Wave, 6LowPan, Thread, Wi-Fi, SigFox and LoRaWan as the major communication protocols for smart home devives. Other studies add to the aforementioned protocols other technologies like Near Field Communication (NFC), RFID, INSTEON and WirelessHART (Al-Sarawi et al. 2017; Yassein, Mardini, and Khalil 2016; Mendes et al. 2015).

[3] https://en.wikipedia.org/wiki/Communication_protocol

Short range wireless communication and low power consumption protocols are ideal for home environments. Among all these different wireless smart home protocols, the most popular, in terms of the number of the connected products, are shown by the statistics in Figure 1.2. These include Wi-Fi, Bluetooth, Bluetooth BLE, ZigBee and Z-Wave, and are often mentioned in the literature.

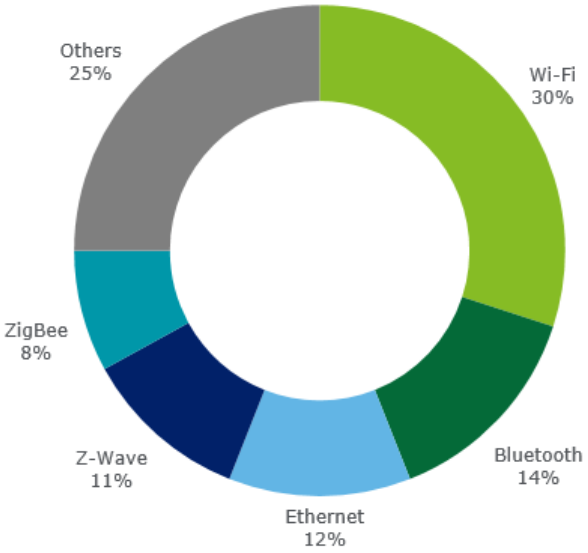

Figure 1.2: Smart Home Wireless Protocols[4]

1.3.2 Smart Home Platforms

The smart home communication protocols are only half of the connectivity's equation. There are also different smart home platforms that are available nowadays that provide enhanced automation capabilities. Several companies introduce cloud-backed systems that are easier for users to set up and that provide a programming framework for third-party developers to build smart-home applications (Fernandes et al. 2017). Examples of such frameworks are Apple HomeKit[5],

[4]Source: `https://www.smarthomedb.com/analytics`
[5]`https://developer.apple.com/homekit/`

Samsung SmartThings[6], Google Weave[7] and OpenHAB[8].

There are platforms that are compatible with multiple communication protocols and some are not, while some can interact with other platforms and some are standalone. For users, it is more convenient to control all the smart devices using a single interface. However, this new technology implies security and compatibility issues. Gyory and Chuah 2017 describe the security flaws found in Samsung SmartThings framework and other issues related to open source frameworks. Further and more extensive analysis of smart home platforms can be found on the article of Fernandes et al. 2017 and PhD thesis of Kalin 2017.

1.4 IoT Penetration Testing for Home Devices

Penetration testing verifies whether there are any weak points or vulnerabilities in the infrastructure. The knowledge of the vulnerabilities is the first step to mitigate them. There are many penetration tools available on the Internet, most of them difficult for users without the appropriate security background. However, Visoottiviseth et al. 2017 developed a penetration testing system for IoT devices called PENTOS in order to increase the users' security awareness. It is developed for penetration testing the physical IoT devices to find the vulnerabilities and it aims at users who are not security experts as it provides an automated easy step-by-step instruction to novice users to perform penetration testing when deploying smart devices at their homes.

Recently, the researchers Chen et al. 2018 described as crucial the improvement of penetration testing coverage in order to mitigate malicious attacks. They described the three types of manual penetration testing: interface testing, transportation testing and system testing. Interfaces that interact with external devices can be vulnerable and these are the main target for interface testing. Transportation testing focuses on design flaws or wrong implementation in communication protocols and weak encryption. Finally, system testing audits the operating systems, firmware for insecure system settings or any other related vulnerabilities.

[6] https://www.smartthings.com/
[7] https://internetofthingsagenda.techtarget.com/feature/Google-takes-on-IoT-with-Brillo-and-Weave
[8] https://www.OpenHAB.org/

Open Web Application Security Project (OWASP) IoT Project (OWASP 2019a) has developed IoT testing guidance with the goal to help manufacturers, developers and consumers build more secure products in an IoT environment. The project consists of a list of Top 10 vulnerabilities (OWASP 2019b) and IoT Security Guidance (OWASP 2019c) for manufacturers and testers. Additionally OWASP has created a specific checklist for conducting penetration testing in Tester IoT Security Guidance, which relate to the categories listed in the Top 10 vulnerabilities. The guidance is meant to be a basic set of guidelines for improving the security of the smart home IoT devices and creating a more secure environment for the users.

1.5 Scope

The primary goal of this research is to examine and evaluate the Z-Wave protocol in practice and determine whether the inhabitants can rely or not on security and privacy when using their IoT devices. Z-Wave is a wireless protocol for automation appliances connection at home, developed by ZenSys and promoted by the Z-Wave Alliance (Gomez and Paradells 2010). It is the world market leader in wireless control with over 100 million products sold worldwide, supported by over 700 manufacturers[9], and provides low-power and low-bandwidth mesh connectivity for home automation applications.

In this section, the main research question and the subquestions are proposed. The methodology to find answers for them is also explained. The subquestions contribute to the answers to the primary research question of the thesis. In the sections that follow, the subquestions will be answered which will lead to answering the main research question. Therefore, the thesis addresses the following primary research question:

To what extent does the Z-Wave communication protocol preserve the security and privacy of a smart home environment and what can the users do to mitigate potential risks?

In order to answer the main research question, it is important to consider the following subques-

[9] https://z-wavealliance.org/about_z-wave_technology/

tions:

1. *What is the architecture of the Z-Wave wireless protocol and what are its security features?*
2. *What kind of information is leaked through wireless communication with the Z-Wave protocol?*
3. *How can eavesdroppers attack the Z-Wave enabled smart devices?*
4. *What factors lead to the leakage of information or the abuse of the devices?*
5. *What are the applicable countermeasures against the security and privacy issues of the Z-Wave protocol?*

To seek answers to the above mentioned research question and subquestions, first of all is necessary to have a better insight on the bigger picture of the smart home ecosystem. For this reason, we searched for literature focused on the security and privacy challenges within a smart home. Many researches have been carried out on smart home wireless communication protocols while some of them examined their security characteristics. It is critical to discern the possible attacks on IoT home devices and the vulnerabilities of the communication protocols that can be exploited by an attacker. A number of proposed measures were found on the literature, based on the security and privacy threats as proposed by other researchers. However, there are only a few studies on the security of the Z-Wave technology comparing to other popular smart home protocols. This will not only help with getting information about the current state of smart homes, but also validate the results from our experiments.

After we performed a general structured literature review on the threats on smart homes and the measures for preserving security and privacy, we proceeded on understanding the Z-Wave wireless protocol's stack and its security principles. It is important to understand in depth the different layers of the protocol and focus on the security implementations. The approach is two fold, theory and practice. Chapter 3 presents a detailed overview of the Z-Wave communication protocol as described in official websites, the Z-Wave Alliance web page and literature in general.

For testing the security of the Z-Wave protocol, we perform an analysis on some popular Z-Wave

enabled domestic home devices. This analysis includes network traffic sent from the controller to the devices and backwards. Despite the broad adoption of transport layer encryption, smart home traffic meta-data is sufficient to identify the leaked information and its source. By analyzing the packets we can determine what kind of information is leaked through the communication and also vulnerabilities that can be exploited (refer Chapter 4). Hardware hacking is necessary and the devices are going to be assessed for verifying whether they contain any sensitive information that can be abused. Later in Chapter 5, we validate the findings from the experiment and based on the feedback from the literature, we designate the security and privacy aspects of the Z-Wave protocol.

1.6 Research Contribution

The thesis aims to contribute in the domain of cyber security for smart homes, and can be categorized into two groups, from research point of view and from society's perspective.

From a research point of view, the literature (Chapter 2) draws that IoT and domestic appliances are in high demand nowadays and at the same time exposed to security risks. By exploring one of the most popular communication protocols for home automation in use, and analyzing its security implementations, future researchers will have the ability to dive deeper into smart home ecosystem while comparing all the existing IoT technologies. Z-Wave is a proprietary protocol which growing rapidly and for that reason there are not so many studies about it, and especially regarding its security.

From society's perspective, the project is inspired by the unawareness and/or ignorance of people when turning their homes into a "smart" place of living. In the market there are numerous of smart appliances, in any price and any type, that are available for bringing convenience and joy into our lives. People, especially those without the appropriate knowledge on IoT or cyber security, tend to purchase these devices based on their functionality and price rather than on their security characteristics. The results of the thesis experiments (Chapter 5) aim to to improve the users' aspect of a more secure way of communication within their smart home, and bring them a better overview of how a smart home should be a "safer" place.

1.7 Thesis Structure

The thesis is divided into seven chapters followed by bibliography. Each chapter has some sections and subsections based on the information that is being discussed. The thesis is structured as follows. In Chapter 2, we present the background and prior research on the security and privacy threats for smart homes and the research approach for the literature review. In this Chapter we also discuss some of the measures that have already been proposed to for preserving security against protocols' vulnerabilities. In Chapter 3, the Z-Wave communication protocol is described in details giving a better insight on the protocol's stack. In Chapter 4, the experimental setting is presented, and the results are shown in Chapter 5. Chapter 6 contains the discussion part over the findings and the research limitations. To summarize the conclusions, in Chapter 7, the research questions are answered and possible future enhancements are discussed.

2 Background & Related Work

The primary goal of this chapter is to collect suitable information on the wireless communication within the smart home concept that can be used for further research in this area and will help to determine whether Z-Wave preserves the security and privacy of the inhabitants in a smart home. Several researches have been carried out focusing on security and privacy challenges of home IoT devices. This Chapter starts with presenting the research methods used for the literature review. Then, defining the security and privacy issues within the smart home ecosystem and presenting countermeasures for these issues, as described from other researchers, are following.

2.1 Research Approach

For stating the issues that the smart home ecosystem is facing, a review on prior researches had to be done. Below, in this subsection, the literature approach and search are described.

2.1.1 Literature approach

The literature was chosen mainly from the following academic databases:

- IEEE Xplore Digital Library
- Scopus
- ResearchGate
- Google Scholar

Besides these four platforms, some of the literature was acquired from websites of associations and governmental bodies. The literature chosen for this research is written in the English language and published the last decade (between 2009 - 2019). The material includes papers, reports and presentations on conferences. Additionally, Google search engine was used to find sources and other useful information for smart home automation architecture.

2.1.2 Literature search

The focus of this research is both on the security and privacy issues resulting of the popularity and the increasingly deployment of the domestic IoT devices, and on the main countermeasures that have already been proposed by other researchers. For this reason, a better understanding of the wireless communication protocols is needed. The search on the aforementioned databases was performed by using a combination of the following keywords: *IoT devices smart home network wireless communication protocols security privacy considerations risk analysis attacks.* Finally, only the papers which content is compatible with the smart home environment are used as a reference for the review. The rest are excluded from our research, unless according to our personal judgment, their methods can be applicable for smart home automation.

2.2 Security and Privacy Challenges

The main threats and vulnerabilities for smart homes are presented in this subsection, as described in the literature. They are divided into security threats and privacy threats.

2.2.1 Security Threats in Smart Homes

Wireless communication between devices implies various security issues and although the communication standards provide some necessary security mechanisms, suppliers do not always implement them as they focus more on performance and functionality of their products. The results of Zillner and Strobl 2015 outline that certain devices using the ZigBee protocol for communication are not entirely secure. The authors proved that the nodes hardware may not be tamper resistant because ZigBee is targeted for low cost applications. The security key could be easily obtained from the device memory if an attacker manages to acquire a node from the operating network that has no anti-tamper measures. Moreover, the home automation system that the authors tested found incapable of resetting or changing the applied network key. Thus, even if an authorized user notices unwanted behavior in the network, there would be no chance of blocking the intruder out.

Fouladi and Ghanoun 2013 have analyzed the Z-Wave protocol based on its encryption, authentication and key exchange protocols. They were able to take control over the remotely accessible critical infrastructure devices like smart door locks. The data transmission was encrypted by an algorithm that was hidden in the source code running the protocol. However, the researchers extracted the secret key from the Z-Wave packet exchange applying reverse engineering on the protocol. Following the previous research, Schwarz 2016 writes about a general problem in the way encryption keys are distributed to new devices in Z-Wave network. When a new device joins the network, a hardware pseudo random generator generates a symmetric key which is then encrypted using a default key. This default key always consists of 16 bytes of value 0. If an attacker knows this, can easily steal the encrypted key, by sniffing the initial device pairing, and then decrypt it using the default key.

Another study from Arias et al. 2015 examined attacks using infected firmware updates through the USB port. Arias et al. tested a NEST Thermostat[10] that is a device for controlling the heating, ventilation and air conditioning in a smart environment. It contained two wireless communication protocols, a Wi-Fi interface and a ZigBee interface. After attempting to attack this device, they discovered that there was no encryption or a digital signature for firmware updates, thus making them sensitive and vulnerable for an attacker. Finally, as shown a compromised device, in this case a thermostat, can be used as a beachhead to other nodes within the network.

In their paper, Ho et al. 2016 studied the security of commodity home smart locks and they concluded that existing smart locks are vulnerable to many attacks. Users can use their mobile devices to control the lock by installing the lock's mobile app, and then pairing their mobile device with the lock using the Bluetooth Low Energy (BLE) protocol. The authors described how and why smart locks are vulnerable to Bluetooth relay attacks due to the lack of defensive hardware in current mobile devices via which the users control the locks. Relay attacks can easily be considered as a form of *active attacks* as described by Komninos, Philippou, and Pitsillides 2014. These attacks focus on altering system resources or affecting its operation. Masquerading, replay attacks and Denial of Service (DoS) are other common forms of active attacks.

[10] `https://nest.com/thermostats/`

Several smart devices have been tested by Sivaraman et al. 2015. The Philips Hue Connected bulb[11] that uses the ZigBee protocol to communicate with the user's app implements access control in the form of a white-listed set of users, that an attacker can easily extract and manipulate it in order to take control. The Belkin WeMo[12] motion sensor and switch kit that connect to the Internet via Wi-Fi, was easily hacked, from anywhere in the world, by obtaining the IP address of the WeMo devices and the ports listening on, and learn the commands and their arguments supported by these devices. Finally, they captured and analyzed Wi-Fi packets to and from the Withings Smart Baby Monitor[13] that comes with an IP camera, and found all the data exchange to be in plain text. Thus, by a "man-in-the-middle" attack allows the attacker to replace the source IP address to his own to gain access to the camera feed. This attack raises also the privacy concerns in a smart home environment.

Bugeja, Jacobsson, and Davidsson 2016 specify security challenges based on different architecture layers, device issues, communication issues and service issues. Different communication protocols used in a smart home require the use of bridges or hubs, and with hardware limitations lead network engineers to implement weaker encryption schemes. In addition to the heterogeneity of protocols, operating system, protocol stack or firmware might not support dynamic patches. Hoang and Pishva 2015 present in their paper a list of all the common attacks on smart home appliances.

Batalla, Vasilakos, and Gajewski 2017 make an extensive research on the security threats of the HAN devices against privacy, services availability, proper operation, authorization data leakage, altering of stored data, interception of information and repudiation of actions. They categorized and described the threats according to the 3-layer software stack (perceptual layer, network layer and application layer).

[11] https://www2.meethue.com/en-us/products/bulbs
[12] https://www.belkin.com/us/Products/smarthome-iot/c/wemo/
[13] https://support.withings.com/hc/en-us/categories/200118087-Smart-Baby-Monitor

2.2.2 Privacy Threats in Smart Homes

Almost every single smart device in a home network collects information based on the user's behaviour. Any malicious use of this information can expose the privacy of the user. The NEST Thermostat collects information, such as the location is being used and device information while controlling the temperature. Arias et al. 2015 tested and proved that this device is vulnerable to privacy data leakage. User information is stored within the unit and uploaded to the manufacturer's cloud service or shared with energy providers for efficient power generation (Mombrea 2014). Arias et al. 2015 and Arabo, Brown, and El-Moussa 2012 sound the alarm because even if the information is not shared, an authorized third party can still gain access to the data from a compromised device and then use it for their own (malicious) purposes.

Privacy can easily be compromised by the behavior of household devices. Notra et al. 2014 and Sivaraman et al. 2015 explored the network behavior of IoT devices including the Philips Hue light bulb and Nest smoke alarm. Based on their experimental evaluation, they highlighted some potential privacy concerns for the user related to activity monitoring. While the smoke alarm encrypts the conversations, the content of the transferred data is under question as it has the potential to carry private user information. Another vulnerability discovered in the smart Philips Hue light bulbs that uses the ZigBee protocol to communicate with the user's app. Request/response exchanged between the bridge and the app are all in plain text. This gives an attacker a great insight of the current state of affairs inside the victim's house.

Komninos, Philippou, and Pitsillides 2014 name the *passive attacks* which can take the form of eavesdropping or traffic analysis. By eavesdropping the authors refer to the unauthorized interception of an on-going communication with unawareness of the involving parties. In traffic analysis the attacker monitors the network traffic patterns in order to extract useful private information from them. These attacks are difficult to detect as they don't alter any data than stealing it.

Lévy-Bencheton et al. 2015 present in ENISA the threats that are relevant to occur on smart home environments. While the categories may differ from analysis to analysis, the content re-

mains the same. They mention that the need for security in smart homes is still underestimated because industry players tend to believe that few attackers have an incentive to attack individuals and that users prefer a low-cost insecure device over a secure device that might be more expensive. This happens because users are not necessarily aware of which private data could be leaked and of how easy it is for someone to obtain these data.

Recently Zheng et al. 2018, conducted some interviews with smart home owners attempting to investigate the reasons behind the purchase of smart appliances and the actions taken to protect their privacy by any means. Their findings indicate that users prioritize convenience and connectivity. They believe that their privacy is protected based on trust in IoT devices manufacturers but do not verify that these protections are in place. Furthermore, users opinion about external entities collecting smart home data depend on perceived benefits from these entities and last and most importantly, users are not aware of privacy risks from inference algorithms operating on data from non-audio or visual devices.

2.3 Measures to Ensure Security and Privacy

The related work that has been aggregated based on the proposed measures is grouped into two sections, security-based and privacy-based measures, although some of the security measures can be used to preserve the privacy and vice versa. However, adapting standard security (and privacy) controls to smart connected homes is challenging as identified by Lee et al. 2014.

2.3.1 Measures for Security Issues in Smart Homes

Batalla, Vasilakos, and Gajewski 2017 define two scenarios that can ensure security within a smart home. The first one has to do with the homogeneity within the smart home where one vendor supplies all the devices and software. This scenario assumes that the data transmission is performed via a specific wireless communication protocol (e.g. ZigBee or Z-Wave) or directly via the Wi-Fi enabled gateway. However, the devices need to stay connected to the Internet for maintaining cloud services provided by the common vendor. The second scenario is when all users

integrate the devices and configure them by their own. Therefore, Smart Home appliances should be provided with tools, which simplify the configuration and exploitation security installation process.

Komninos, Philippou, and Pitsillides 2014 state in their paper that in order to achieve better security for smart homes, it is necessary to ensure data integrity and authenticity. They describe many techniques from the literature on how to ensure these principles. Cryptographic hashing techniques and the use of load profiling algorithms are among the countermeasures that they analyze. For the authenticity, they study the use of message authentication codes such as HMAC, Physically Unclonable Function (PUF) modules and different encryption schemes.

Finally, Bugeja, Jacobsson, and Davidsson 2016 mention some firewall devices for monitoring, analyzing and blocking threats in real-time. Virtual Private Networks (VPN) are another approach for encrypting devices and connections for increasing the security levels within a smart environment.

2.3.2 Measures for Privacy Issues in Smart Homes

Notra et al. 2014 believe that due to the broad range of IoT devices and and communication protocols, the security needs to be outsourced as a service to an authorized trusted provider (ISP or SaaS provider). They propose a solution at the network layer. An external party maintains a database of access control rules in the cloud that protects the IoT devices. They applied their solution on the Nest smoke alarm, where they have discovered lack of encryption through the network traffic between the bridge and the app. They managed to block all outgoing traffic by enabling firewall with the exception of three destinations representing necessary servers like authentication, notification and token renewal, by applying a rule on the home gateway. By this, the researchers were able to mitigate the privacy risks emerged from this specific device's vulnerability. A similar approach at the network layer to block threats was proposed by Sivaraman et al. 2015 who have defended a three-party architecture in which a specialist provider offers security-as-a-service, then made a prototype of it using open-source SDN platforms, and finally evaluated its effectiveness.

Komninos, Philippou, and Pitsillides 2014 cite that privacy can be achieved through anonymization, trusted aggregators, homomorphic encryption, perturbation models, verifiable computation models and data obfuscation techniques.

Hoang and Pishva 2015 present the implementation of a TOR-based anonymous system to help protect user privacy and make the smart home appliances more secure against cyber attacks. They also expanded their research into a deeper level of TOR, its sub-project the Tails, a live operating Debian-based Linux distribution system which can run on almost any CPU based gadget like USB stick. Their results show that Tails have many security features such as multilayer encryption, data transmission in distributed manner over a huge amount of voluntary nodes around the world, and MAC address spoofing, is indeed a suitable approach to solve recent security problems in TCP-based smart home appliances.

Similar to previous privacy mitigation approach, is the study of Acar et al. 2018. They suggest the use of VPN or TOR-like tools as both methods will prevent an attacker to follow the communication. They also suggest a signal attenuator that can be used to protect the home environment from sniffing the internal network traffic. Finally, they proposed their own privacy-based approach based on generating spoofed traffic. In this way, even if the user is absent from home, generating false activity for the user's presence behavior will mask the actual absence.

2.4 Conclusion

In this chapter, we presented the main threats and vulnerabilities for smart homes based on prior studies. We have seen that smart home automation brings convenience to the users, but at the same time a lot of risks, as the wireless communication protocols that are being used within a smart environment are not entirely secure. Subsequently, some measures to ensure security and privacy in smart homes were discussed. In the later chapters, we will focus on the Z-Wave communication protocol for home automation and explore it based on its security characteristics.

3 Z-Wave Communication Protocol

Z-Wave is a proprietary wireless communication protocol for automation appliances connection at home, developed by ZenSys, a Danish company, in 1999 and promoted by the Z-Wave Alliance since 2005. In 2008, Z-Wave was acquired by Sigma Designs and on April 2018 the Z-Wave technology, including all the business assets, was sold to Silicon Labs for $240 million (Labs 2018).

Z-Wave is an interoperable and Radio Frequency (RF) based communication technology specially designed for monitoring and controlling residential applications including lighting, HVAC and security systems. It discerns to enable reliable transmission of short messages from the control unit to one or more devices in the home network with the minimum of noise. It is a low power and mesh network wireless protocol that appears in broad range of consumer products all over the world.

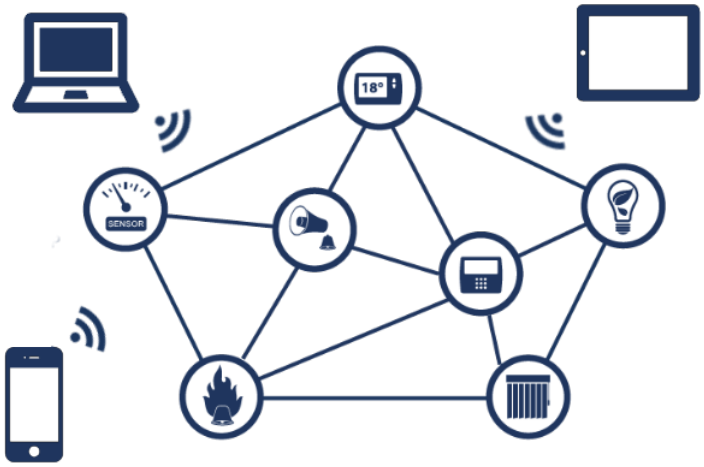

Figure 3.1: Z-Wave network topology[14]

[14]Source: `https://www.athom.com/en/what-is-z-wave/`

3.1 Introduction to the Z-Wave Terminology

In this section, the terminology used for the Z-Wave description is presented for better understanding of the protocol. This includes the network topology, modulation schemes, encoding mechanisms and some cryptographic algorithms.

3.1.1 Network Topology

The Z-Wave protocol uses the concept of the mesh networking. It is a powerful and advantageous, especially for the smart home ecosystem. In mesh networks, a message hops from one node to another until it reaches its destination node. When the controller wants to send a request to a node that is outside its range, then another node undertakes to relay the request. This technology makes the wireless network stronger when more devices are added, as it extends the range. In a Z-Wave network the message hop threshold is five, meaning that the message can be repeated five times before is dropped, and the optimum number is two.

3.1.2 Modulation Type

Frequency Shift Keying (FSK) is a frequency modulation technique used in signal processing, in which the digital information is transmitted through discrete frequency changes of a modified waveform. Figure 3.2 shows a typical FSK signal modulation. Gaussian Frequency Shift Keying modulation (GFSK) is a form of FSK that uses a Gaussian filter. The data pulses pass through the Gaussian filter before the modulation. The Gaussian filter is given by:

$$g(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}(\frac{t}{\sigma})^2}$$

where σ is related to the filter's bandwidth B:

$$\sigma = \frac{\sqrt{ln2}}{2\pi B}$$

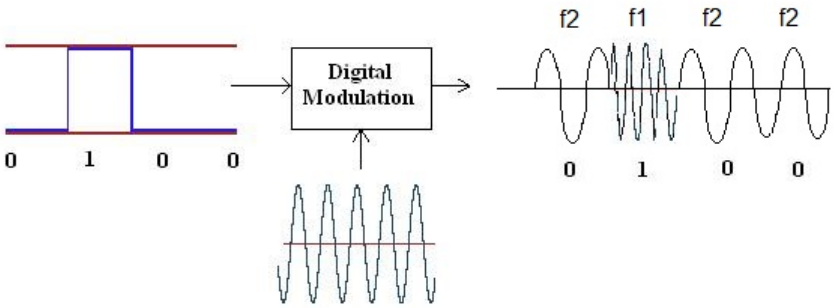

Figure 3.2: FSK modulation type[15]

3.1.3 Serial Encoding Mechanisms

Z-Wave uses Manchester or Non-Return-to-Zero (NRZ) for encoding data within the modulated RF signal. An NRZ signal represents a binary 1 with a positive voltage, and a binary 0 with a negative voltage. In a Manchester encoded signal, the clock signal is combined with the data signal. In other words, Manchester is an NRZ encoding that is XORed with the clock. A Manchester encoded signal represents a binary 0 by a low to high transition, and a binary 1 by a high to low. Figures 3.3 & 3.4 show an example of NRZ and Manchester encoding schemes.

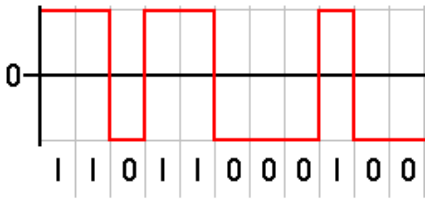

Figure 3.3: Non-Return-to-Zero encoding

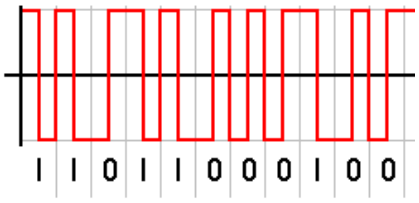

Figure 3.4: Manchester encoding

The Z-Wave encoding scheme depends on the operating data rate. The Manchester encoding is used at 9.6 Kbps of data rate, while at higher data rates of 40 Kbps and 100 Kbps, Z-Wave devices use NRZ encoding.

[15]Source: `http://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-FSK.html`

Bit Rate	Modulation	Encoding	Center Frequency (EU)
9.6 Kbps	FSK	Manchester	868.42 MHz
40 Kbps	FSK	NRZ	868.40 MHz
100 Kbps	GFSK	NRZ	869.85 MHz

Table 3.1: Z-Wave PHY layer configuration

3.1.4 Cryptography

Initialization Vector (IV)

An initialization vector (IV) is a unique binary sequence, typically random, used for encryption operations. It is used to randomize the encryption and to produce disparate ciphertexts.

Advanced Encryption Standard (AES)

Advanced Encryption Standard is one of the most used algorithms for block encryption. It is a symmetric-key algorithm, which means the same key is used for both encrypting and decryption of the data.

AES-ECB

The Electronic Codebook (ECB) mode is the simplest of all the encryption modes of operation. The message is divided into different blocks and is encrypted separately using the same for each block.

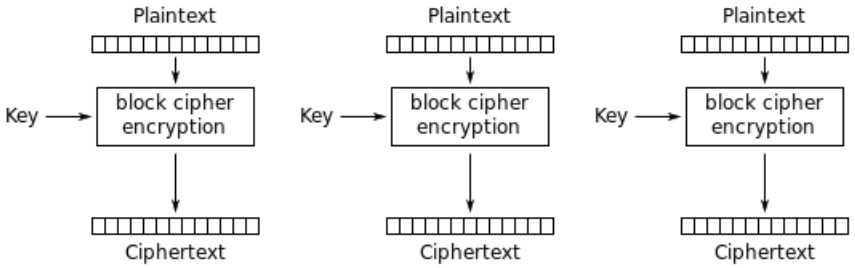

Figure 3.5: ECB mode encryption[16]

[16]Source: `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

AES-OFB

The Output Feedback (OFB) mode allows a block cipher to be used as a stream cipher. The output of the previous block cipher encryption, is used as an input for the next block cipher, and at the same time it is XORed with the plaintext to produce the ciphertext. The IV has the same size as the block that is encrypted.

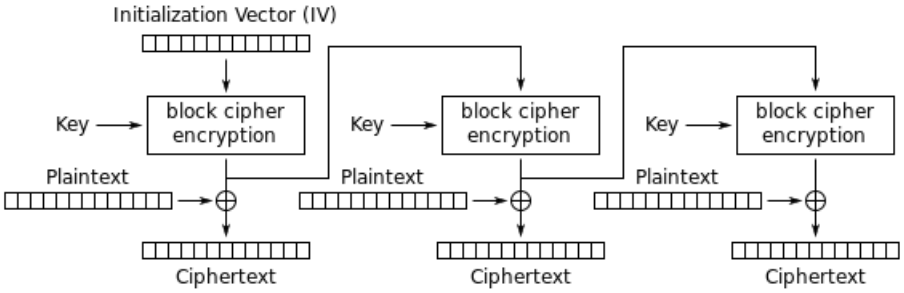

Figure 3.6: OFB mode encryption[17]

AES-CBC

In Cipher Block Chaining (CBC) mode of operation, an XOR operation is applied to the plaintext with the previous ciphertext, and the result is encrypted using the key for producing the ciphertext. This ciphertext is used as an input for the next block.

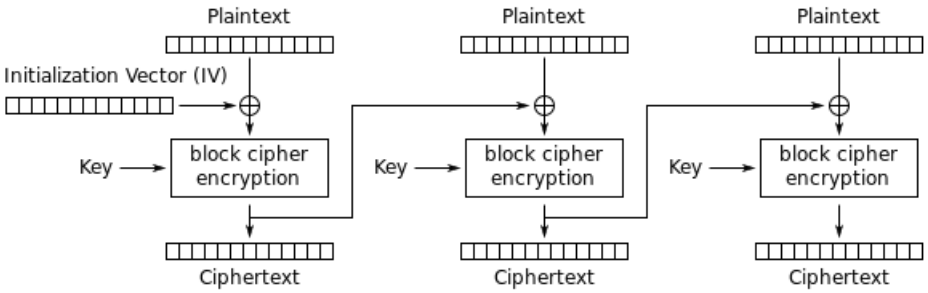

Figure 3.7: CBC mode encryption[18]

AES-CBC-MAC

A Cipher Block Chaining Message Authentication Code (CBC-MAC) is a technique used for

[17]Source: `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`
[18]Source: `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

creating a message authentication code from a block cipher. The message is encrypted using the CBC encryption mode with a fixed zero IV, and the only output ciphertext from the last block forms the MAC.

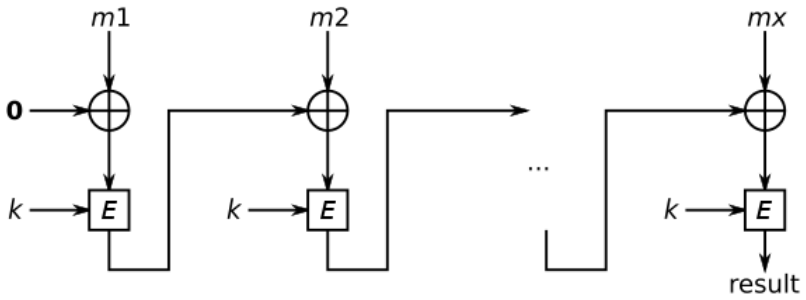

Figure 3.8: CBC-MAC mode encryption[19]

AES-CTR

In the Counter (CTR) mode, an IV/nonce is used as a counter. A nonce is an arbitrary number that can be used only once and the counter is incremented for each block. The XOR operation is performed on the output block and all the encryption blocks use the same key.

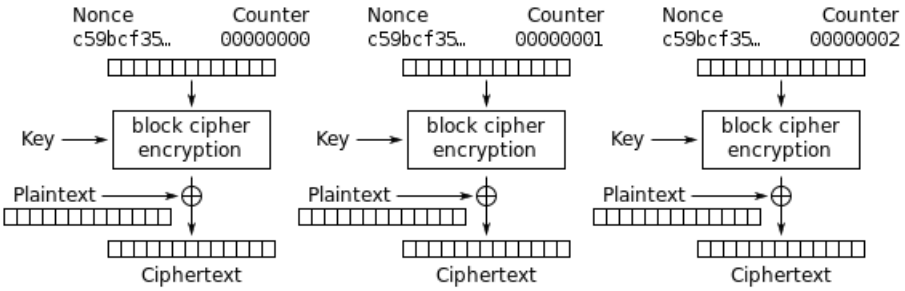

Figure 3.9: CTR mode encryption[20]

AES-CMAC

Cipher-based Message Authentication Code (CMAC) is a block cipher-based message authentication code algorithm. It provides the authenticity and the integrity of a binary data. CMAC is similar to the CBC-MAC mode, with the difference that CMAC XORs the last block with a

[19]Source: `https://en.wikipedia.org/wiki/CBC-MAC`
[20]Source: `https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation`

secret value called tweak. This means that the last block is operated differently from the others, removing the man-in-the-middle attack vector.

AES-CCM

Counter with CBC-MAC (CCM) is an authenticated encryption algorithm for providing both authentication and confidentiality. It combines the CBC-MAC mode of operation with the Counter mode of encryption.

ECDH

Elliptic Curve Diffie–Hellman (ECDH) is a variant of the public key (or asymmetric) Diffie-Hellman algorithm for elliptic curves. It is a key-agreement protocol, where two parties have an elliptic curve public and private pair of keys in order to establish a shared secret over an insecure channel. This means that ECDH defines how keys are generated and exchanged between the parties. ECDH protocol solves the problem of the man-in-the-middle attack. An eavesdropper can intercept the communication but cannot decode the shared secret.

3.2 Network Communication

In a smart home environment that uses Z-Wave protocol, the maximum number of devices is restricted to 232 and there are two different types, controllers and slaves, as described in the paper of Yassein, Mardini, and Khalil 2016. The controllers are the devices that initiate the control commands and send out the commands to the other nodes. Slave devices are the nodes in the network that receive the commands, reply on and execute them. The slaves can also forward the commands to other nodes, in case the controller is unable to reach these nodes directly based on the frequency range. To achieve this, Z-Wave uses the concept of mesh networking. Based on this technology, any node is able to talk to adjacent nodes directly or indirectly, making the Z-Wave able to cover all the areas of the home. Thus, the more nodes a network contains, the stronger and most robust it is. An illustration of a Z-Wave network topology is shown in Figure 3.1.

Controllers

A controller is a device that is able to communicate with all the nodes within a Z-Wave network. In every network there is one primary controller, the one that creates the whole network from the beginning, and it is described as the ”master” controller. The primary controller is responsible for including and excluding nodes in the network and carries the reliable information about the network topology. However, additional controllers can be added to the network by the primary controller. These are called secondary controllers but they cannot participate in the inclusion and exclusion process. Controllers are categorized into portables and statics. A portable is the controller that can change position within the Z-Wave network, if needed, for finding the fastest route to the destination. Static controllers do not change position and are usually secondary controllers that need to have power continuously. A Z-Wave network can contain a Static Update Controller (SUC) that receives notifications regarding the updates in the network topology from the primary controller. There can only be one SUC in the network and this only after the primary controller's request. Optionally, a SUC can have enabled ID Server functionality (SIS). A SIS controller, which is typically a primary controller, allows other controllers to include and exclude nodes on its behalf. An example of a primary controller is the Aeon Labs Aeotec Z-Stick Gen5 that we used for setting our network, as described in Chapter 4.

Slaves

Slave nodes are the nodes that receive a command from the controller and perform an action based on the command they received, and they do not contain a routing table. However, routing slaves can also send commands to other nodes, if they are requested by the controller. Finally, there are enhanced slaves that contain a real time clock and an EEPROM for storing application data, like weather stations. For the experiment that we conducted in Chapter 4, we used two devices as simple slaves, a smart light bulb and a wireless electronic deadbolt door lock.

Network ID & Node ID

Each device in a Z-Wave network is designated as a unique node and must distinguish via an identity. The protocol uses an identifier, called Network ID (or Home ID), to distinguish the Z-Wave network from other networks. The Network ID has a length of 4 bytes (32 bits) and is factory programmed to the controllers. During the inclusion phase, the primary controller

assigns the Network ID to each node. This formulates the Z-Wave network, as nodes can communicate with each other only when they belong to the same network by carrying the same Network ID.

Node ID is an identifier that the Z-Wave protocol uses to address all the individuals nodes within the network. The Node ID has a length of 1 byte (8 bits) and must be unique. This means that under the same network (one Network ID), two nodes cannot have the same Node ID.

Table 3.2 displays the differences between the Network ID and Node ID.

	Network (Home) ID	**Node ID**
Definition	The Network ID is the common identification of a Z-Wave network	The Node ID is the individual address of a node within the same network
Controller	Factory established by default	Controller has its own Node ID predefined (typically 0x01)
Slave	No Network ID by default, assigned by the primary controller	Assigned by the primary controller

Table 3.2: Network ID & Node ID

Inclusion Procedure

The inclusion process is the main function for creating a Z-Wave network and assigning the Home ID and Node ID to the paired slaves. The process is initiated on the controller and then adding the device on the pairing mode. This is achieved by pressing a specific button on the devices or by physically resetting them. Only when a device is assigned to Z-Wave network, the controller can communicate with it. When a device enters the inclusion mode, the controller listens to a Node Information Frame (NIF), which transmitted by the device. NIF is a way for the slave device to send its capabilities, like supported commands, to the controller. Then, once the controller receives the NIF, replies with the Home ID and Node ID. The last step of the inclusion process is the confirmation of the slave device that sends back to the controller, before the key exchange occurs between the controller and the device(s). Figure 3.10 shows the main inclusion process in a Z-Wave network.

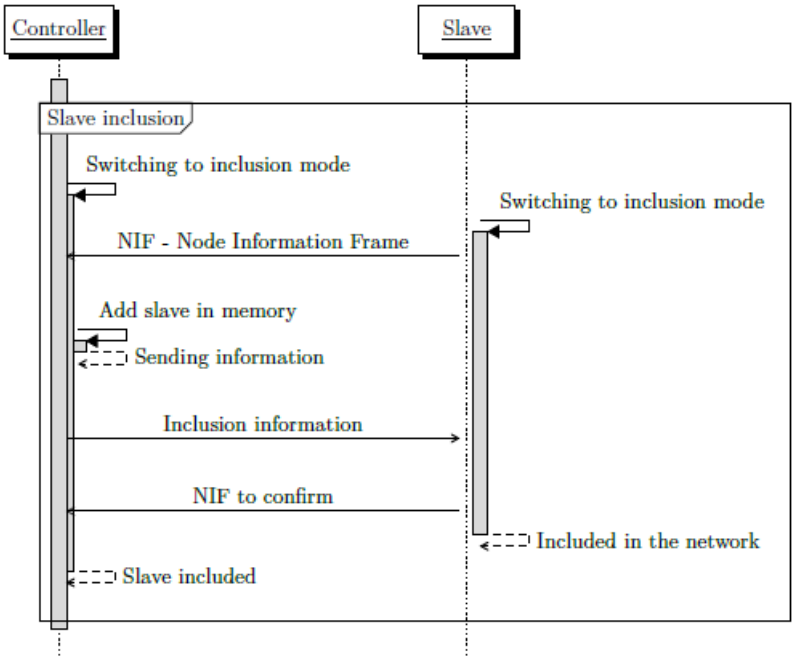

Figure 3.10: Inclusion procedure in a Z-Wave network (Rouch et al. 2017)

3.3 Z-Wave Protocol Stack

The Z-Wave protocol is a low bandwidth protocol designed for reliable wireless communication in a low cost control network. The protocol's main purpose is to transfer short control messages in a reliable manner from a control unit to one or more nodes within the network.

The protocol consists of five main layers: the Physical layer (PHY), the Media Access Control (MAC) layer, the Transport layer, the Network layer, and finally the Application layer.

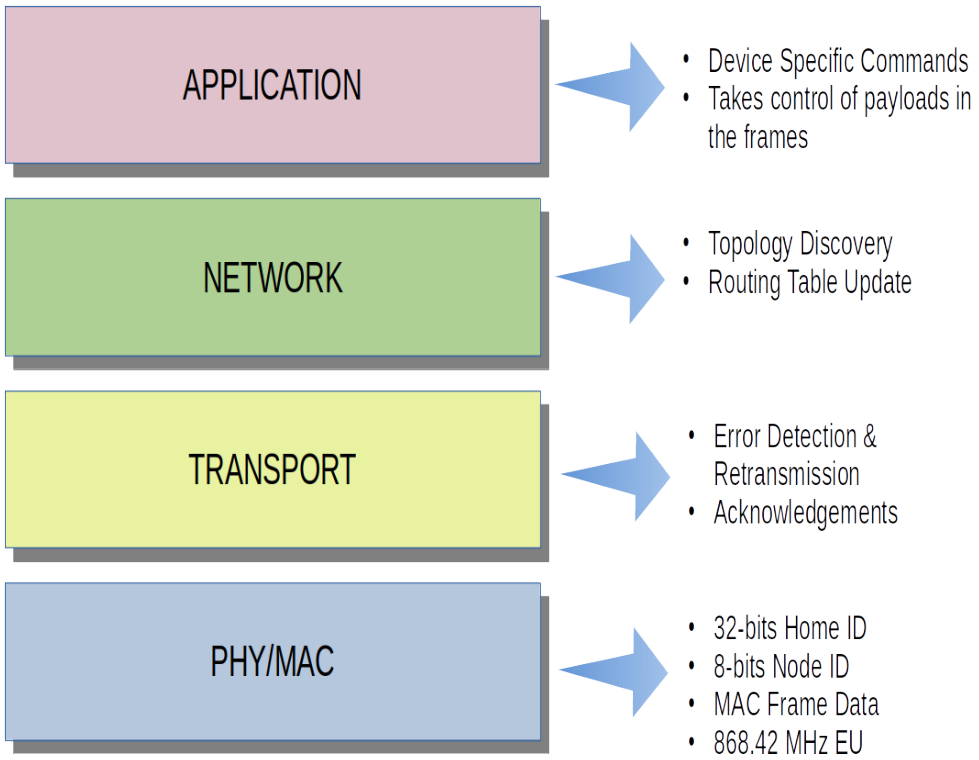

Figure 3.11: Z-Wave protocol stack

Physical/MAC Layer

Physical (PHY) and Media Access Control (MAC) layers are defined in ITU-T Recommendation G.9959, which contains specifications for sub GHz radio communication protocols (ITU-T 2015). The physical layer is responsible for the modulation type, the coding of the data frame, the selection of the frequency, the transmission and reception of the MAC data frame. The data stream consists of a preamble, start of frame (SoF), MAC frame data and end of frame (EoF). Z-Wave devices operate under multiple frequencies depending on the different parts of the world. In Europe, Z-Wave works in 868.42 MHz ISM band and uses either Gaussian Frequency Shift Keying (GFSK) or Frequency Shift Keying (FSK) for signal modulation, offering data rates of 100 Kbps and 9,6 to 40 Kbps respectively. The data stream is encoded using Manchester code or Non-Return-to-Zero (NRZ). The MAC data frame contains information such as the Home ID, Source ID, Frame Control, length, Destination ID, Data payload and checksum. This data frame is passed on the next layer, the Transport layer.

Transport Layer

Z-Wave transport layer is responsible for transferring the data between the nodes, retransmission, packet acknowledgements and checksum checks. Each Z-Wave frame layout consists of the 32 bits Home ID, 8 bits source node ID, the frame control that defines the type of the frame (multi-cast, single-cast or broadcast), 8 bits of length and destination node ID, data payload (including the command class and command) ending with 8 bits of checksum value. The checksum is responsible for the transport layer to detect and discard false data frames.

Single-cast frames are the frames that are transmitted to one specific node with in the network, while a multi-cast frame is transmitted to more than one nodes. For a single-cast frame, the acknowledgement is necessary for ensuring reception. However, these frames do not get acknowledged and the communication cannot be reliable. Broadcast frames are multi-cast frames received by all the nodes within a Z-Wave network, without being acknowledged by any node.

Network Layer

The network layer controls the routing of the packets based on the network topology from the routing table. This layer is also responsible for sending a frame with a correct and valid repeater list and for ensuring the data transmission from one node to another. The Z-Wave protocol enables automatic topology scans and routing table updates, for optimizing the frame routing. This is beneficial in case one or more nodes (slaves) change position or removed from the network after the installation.

Application Layer

The Z-Wave application layer is responsible for defining the data frame and decoding and executing the commands. The application frame consists of the header, that determines if the type of the frame, the command class, the commands and the command parameters. If the Z-Wave network contains a controller, then the third party software associates the command parameters. The command classes identify the functionality of each device within the Z-Wave network and can contain multiple commands. The commands specify the specific action from the command class that needs to be taken. The command parameters contain information related to the spe-

cific command, which also defines the number of them (Labs 2019). Currently, according to the Silicon Labs, Z-Wave identifies 98 active command classes. Figure 3.12 illustrates a Z-Wave single-cast frame format in different layers.

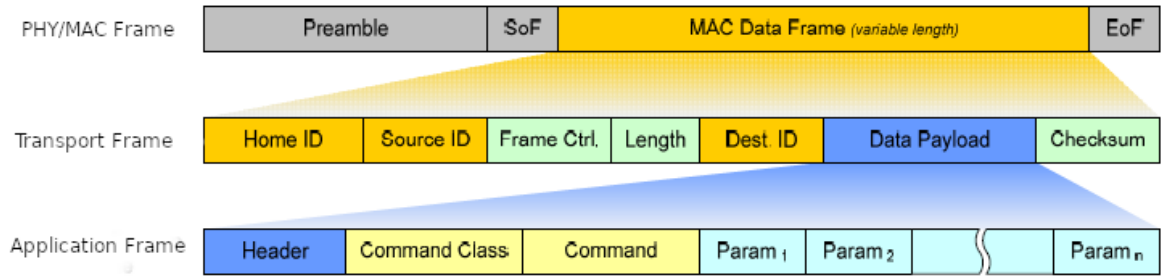

Figure 3.12: Z-Wave frame format

3.4 Z-Wave Security

Security is a very crucial issue, especially when it comes to smart homes, where the end-users often do not realize the risks or they tend to ignore the importance of the security on IoT. Z-Wave devices are related to users' activity within a smart hoe environment and usually they gather personal information or control sensitive appliances. Therefore, security plays an important role in the Z-Wave network, including all the devices without any exceptions. Z-Wave protocol supports three levels of security: no security, S0 security and S2 security. As a first level of security for a Z-Wave network is the unique Home ID that identifies the network. Every new device that enters the network is assigned with the Home ID by the primary controller. However, the communication occurs in plain text by default, which allows an attacker within the radio range to capture and decode the data frame.

3.4.1 Z-Wave S0 Security Framework

The S0 security framework includes a security command class which is used to encrypt all the communications using the AES-128 encryption. This makes Z-Wave secure to a certain extent. Each Z-Wave controller generates a 16 bytes shared network key using a Pseudo Random Number

Generator (PRNG). During the pairing process, the network key is sent to the device (slave) from the controller over the radio. Although the network key is encrypted using a temporary key, before being exchanged to the devices, it is being leaked and proven by Fouladi and Ghanoun 2013 that the temporary key that is used for the encryption is 16 bytes of zero:

$$K_T = 00\,00\,00\,00\,00\,00\,00\,00\,00\,00\,00\,00\,00\,00\,00\,00$$

After the - exchange of the network key, the controller derives an encryption key for encrypting the data by encrypting 16 bytes of 0x55, and an authentication key for authenticating the data, by encrypting 16 bytes of 0xAA using the network key.

$$K_E = AES - ECB_{K_N}(0x55)$$

$$K_A = AES - ECB_{K_N}(0xAA)$$

For encrypting the message Z-Wave uses AES-OFB and then it enhances it, for preventing replay attacks, with authentication using AES-CBC-MAC to generate the message authentication code.

$$C = AES - OFB_{K_E}(IV, P)$$

$$MAC = AES - CBC - MAC_{K_A}(IV, P||C)$$

The initialization vector (IV) is 16 bytes long and the plain text (P) is the payload that contains the Z-Wave header, source node, destination node, length, command class and all the command parameters.

As a result of the hole during the exchange of the network key, any attacker using a specific equipment is able to sniff the radio data during the inclusion process. He can easily decrypt the key, since the encryption key is all zero, and then gain access to the network and eventually to all the devices. For this reason, and because all the devices within a Z-Wave network share the same key, the Z-Wave Alliance introduced the S2 framework in 2017, updating the security command class to security command class S2.

3.4.2 Z-Wave S2 Security Framework

The Z-Wave S2 security framework was introduced in order to bypass the problems with the S0 security. This latest framework also operates with the concept of a shared network key and is based on AES-128 for data links. Nevertheless, Elliptic Curve Diffie-Hellman (ECDH) was added for the key exchange protocol. S2 nodes use an ECDH shared key to derive a temporary key, which allows the controller to send the network keys securely to a node within the network and establish a secure communication.

The Z-Wave protocol divides the network into three security classes, the *S2 Access Control*, *S2 Authenticated* and *S2 Unauthenticated*. The S2 Access Control class is the most trusted class and it is referred to access control devices, such as door locks. The S2 Authenticated class is the most common security class for a Z-Wave network and applies to most devices. The S2 Unauthenticated class is the least trusted security class and is designed for controllers without a user interface for authenticating a node or for some low risk devices, such as lights. Each S2 security class has its own AES-128 network key.

In a wireless environment the authentication of the devices is of high importance. The Z-Wave S2 authentication process allows the controller to verify that the device which asks to join the network is truly the physical device that enters the network. A Device Specific Key (DSK) plays the role of authenticating the pairing, by entering a code to the controller or by scanning a QR code. However, this is not always feasible due to the limitation of UI on IoT smart devices.

	No Security	**S0 Security**	**S2 Security**
Message Encryption	None	AES-OFB	AES-CCM
Message Authentication	None	AES-CBC-MAC	AES-CMAC
Number of keys	-	Single	Multiple
Key Exchange	-	Symmetric	ECDH

Table 3.3: Comparison of Z-Wave security levels

4 Experiments

The goal of this chapter is to describe how to set up a real working Z-Wave environment and to perform hands-on testing of the devices in order to explore the security and privacy aspects of the protocol. For confidentiality and security reasons, the name and the model and all the information that may reveal the identity of the door lock, will remain confidential.

4.1 Equipment

For this research, a Zipato Light Bulb 2 and a Z-Wave Door Lock were selected as the target devices (Figure 4.2). The Zipato Bulb supports the Z-Wave Plus technology, whereas the door lock has a Z-Wave 300 series (Generation 3) chip. In order to bind these two devices together and be able to control and automate them, a controller is needed. For this purpose, an Aeon Labs Aeotec Z-Stick Gen5 controller was acquired that supports the Z-Wave Plus technology (Figure 4.1). Z-Stick Gen5 is a Z-Wave network controller that helped us build our own gateway, locally hosted and within a low budget. Usually the most expensive part of any smart home is its gateway.

The choice of the smart devices was based on the popularity of smart lights nowadays, and the critical importance of a smart lock. Smart bulbs can be found in most of the households because they make lighting more fun and comfortable and provide convenience to the users as they can control them remotely. The Z-Wave controllable door lock was chosen as it can be applied to any door and since it can be controlled remotely to lock/unlock the door, it implies high risk in case of tampering.

Figure 4.1: Aeotec Z-Stick Gen5

Figure 4.2: Zipato Bulb 2

4.2 Setting up the Z-Wave Network

4.2.1 OpenHAB

OpenHAB is an open source project that provides a framework for an Internet of Things system. It supports many different technologies and systems and in its database there are thousands of registered devices. It is easy to download and install it and runs the server on different platforms and operating systems (Linux, Windows, Raspberry Pi, etc.). OpenHAB website[21] offers a detailed installation overview, providing all the information and prerequisites for downloading, installing and setting up the OpenHAB server. Once the server is set up, then the user needs to install the appropriate binding, in our case is the Z-Wave binding. For our research, we used a laptop running Ubuntu 18.04 LTS on a virtual machine to set up the OpenHAB version 2.

Initial Setup

After the installation of OpenHAB 2, initial setup is required. By typing on a web browser `http://localhost:8080`, the OpenHAB dashboard appears. There is a choice between four packages: Simple, Standard, Expert and Demo. These packages contain some users interfaces and add-ons for configuring the ”things”. The recommended package for starters is the *Standard* package. This installed the ”Paper UI” from where we can install add-ons, discover and configure things, the ”Basic UI” where we can have a look at the created sitemaps, and the ”Habpanel”

[21] `https://www.OpenHAB.org/docs/installation/`

where we can create dynamic dashboards for our devices. However, for our experiment, we installed the *Expert* package. It is similar to the Standard package, with only difference it contains the "Habmin", which is a powerful, modern and professional user interface suited for Z-Wave setups.

The Paper UI is an interface that helps setting up and configuring the OpenHAB instance. From here, we can easily install and uninstall the appropriate bindings, under the Add-on tab. In order to have a complete running Z-Wave environment we installed the Z-Wave binding which supports an interface to a wireless Z-Wave home automation network and its database contains many of the Z-Wave products that are available on the market. The binding uses a standard Z-Wave serial stick (USB controller) to communicate with the Z-Wave devices.

Controller Configuration

After the installation of the OpenHAB and the Z-Wave binding, the next step for creating a Z-Wave network is the configuration of the primary controller. First, we plugged the Aeotec Z-Stick on the laptop via a USB port. Next, we had to search for the controller under the "Inbox" tab on the Paper UI. When the OpenHAB found the controller, it appeared on the screen and by clicking on it, the controller was successfully installed on the dashboard. For configuring the Z-Wave controller, it is necessary to know on which port the Linux operating system assigned the controller. The Linux command `ll /dev/ttyACM*` indicates on which port the USB we plugged is assigned on. However, an extra step is required when configuring the OpenHAB for the first time. In order to be able to read the USB port, we need to edit the `/etc/default/OpenHAB2` file and add an extra line: `EXTRA_JAVA_OPTS=-Dgnu.io.rxtx.SerialPorts=/dev/ttyUSB0`. Finally, a restart appllies all the changes to OpenHAB, in order to recognize the controller as an active device. By typing `sudo service openhab2 restart`, OpenHAB services restart.

Devices Inclusion

Once the controller is configured, the inclusion process of the slave devices begins. This can happen in two ways, either using the controller itself or adding them through the OpenHAB software. For our experiment, we decided to pair the devices through the controller itself. We

unplugged the controller from the laptop and we initiated the inclusion mode by pressing the main button once. Next, we put the devices into inclusion mode. Each device has its own way of entering the inclusion mode, depending on the manufacturer. The blue LED on the controller blinks fast during a network discovery and stays solid for 2 seconds to indicate that the inclusion of the device into the network was successful. Then, the blue LED returns to blinking slowly, indicating that it is ready for further inclusions. Finally, when we had the devices included, we plugged again the Z-Wave controller to the laptop and the two new paired devices appeared on the OpenHAB interface.

4.2.2 Secure Inclusion Mode

When we performed a test on the devices to check whether they operate properly and communicate with the controller without any obstacles, we discovered that the controller was not able to speak to the door lock. For some reason the commands did not send to the device. On the other hand, we faced no problems with the light bulb. We were able to send all the commands, such as turn on/off and change the colors. Once we checked on the log file, we observed that the `COMMAND_CLASS_DOOR_LOCK not found`, refer to figure 4.3. Although the device was successfully paired with the controller, the controller was not able to interact with it. The reason was the *Secure Inclusion Mode*. Z-Wave controller offers three options, in conjunction with OpenHAB, for secure inclusion of the devices: *All Devices*, *Entry Control Devices* and *No Security*. In "All Devices" mode, all devices reporting the security command are securely included into the network. The "Entry Control Devices" mode addresses to important, like locks, that should be securely included into the network. By default the option that was enabled was for entry control devices only. This should work for the door lock, however it appeared invalid for our smart lock.

```
2019-01-24 20:35:11.968 [INFO ] [ing.zwave.handler.ZWaveSerialHandler] - Connecting to serial port '/dev/ttyACM0'
2019-01-24 20:35:11.995 [INFO ] [ing.zwave.handler.ZWaveSerialHandler] - Serial port is initialized
2019-01-24 20:35:11.999 [INFO ] [ve.internal.protocol.ZWaveController] - Starting ZWave controller
2019-01-24 20:35:11.999 [INFO ] [ve.internal.protocol.ZWaveController] - ZWave timeout is set to 5000ms. Soft reset is false.
2019-01-24 20:35:15.335 [INFO ] [g.discovery.internal.PersistentInbox] - Added new thing 'zwave:device:4dca268a:node5' to inbox.
2019-01-24 20:35:15.410 [INFO ] [g.discovery.internal.PersistentInbox] - Added new thing 'zwave:device:4dca268a:node2' to inbox.
2019-01-24 20:45:16.841 [WARN ] [nal.converter.ZWaveDoorLockConverter] - NODE 5: Command class COMMAND_CLASS_DOOR_LOCK not found
```

Figure 4.3: Door lock log file

Once we altered the secure inclusion mode to ”All Devices”, we excluded the lock from the network and we included it again, this time securely. This fixed the problem and the door lock was working properly. Nevertheless, this triggered our attention about the light bulb. We assumed that the bulb was insecurely included into the network. When we checked the log file, our assumption was validated, refer to figure 4.4. No security required for all the commands that the bulb supports.

```
2019-01-15 10:53:34.249 [DEBUG] [nal.protocol.ZWaveTransactionManager] - NODE 2: Application Command Request (ALIVE:DONE)
2019-01-15 10:53:34.249 [DEBUG] [ng.zwave.internal.protocol.ZWaveNode] - NODE 2: resetResendCount initComplete=true isDead=false
2019-01-15 10:53:34.249 [DEBUG] [ng.zwave.internal.protocol.ZWaveNode] - NODE 2: Incoming command class COMMAND_CLASS_SWITCH_COLOR, endpoint 0
2019-01-15 10:53:34.249 [DEBUG] [ng.zwave.internal.protocol.ZWaveNode] - NODE 2: SECURITY NOT required on COMMAND_CLASS_SWITCH_COLOR
2019-01-15 10:53:34.250 [DEBUG] [tocol.commandclass.ZWaveCommandClass] - NODE 2: Received COMMAND_CLASS_SWITCH_COLOR V1 SWITCH_COLOR_REPORT
2019-01-15 10:53:34.250 [DEBUG] [.commandclass.ZWaveColorCommandClass] - NODE 2: Color report COLD_WHITE 0
2019-01-15 10:53:34.250 [DEBUG] [nal.protocol.ZWaveTransactionManager] - NODE 2: Commands processed 1.
```

Figure 4.4: Light bulb log file

4.3 Assessing the Z-Wave Devices

For the purpose of this research the most suitable tools for testing RF protocols were selected. These include both software and hardware. However, the Z-Wave protocol is a proprietary protocol and there are not so many studies carried on its security. For this reason, there are not so many available open source tools for sniffing and intercepting the protocol.

4.3.1 Preparation - Tools

Hardware

The tools that were selected for the protocol hacking are a HackRF One from Great Scott Gadgets and a RTL-SDR dongle. The HackRF One is a Software Defined Radio (SDR) peripheral capable of transmission and reception of radio signals from 1 MHz to 6 GHz. The RTL-SDR dongle we used is a NooElec NESDR Mini 2 SDR which has a frequency capability of 24 - 1750 MHz, both covering the Z-Wave frequency. Figure 4.5 shows the HackRF One and figure 4.6 the RTL-SDR dongle.

Figure 4.5: HackRF One

Figure 4.6: RTL-SDR dongle

For the hardware hacking hacking part of the Z-Wave devices, we used a Digital Multimeter which is a test tool for measuring the voltage, the current and the resistance of a Printed Circuit Board (PCB), and a Logic Analyzer used for debugging electronic logic circuits. The Logic Analyzer is used to capture and display multiple signals from a digital system or circuit. Figures 4.7 and 4.8 show the Digital Multimeter and the Logic Analyzer, respectively.

Figure 4.7: Digital Multimeter

Figure 4.8: Logic Analyzer

Software

For this research, the software selected based on open source projects compatible with the hardware mentioned in the previous paragraph. For the HackRF One, some libraries must be installed on the computer to start using the device. We installed the libraries by typing

`sudo apt install hackrf` on the Linux terminal. GNU Radio Companion is also needed alongside with the HackRF One. GNU Radio is a free open-source software development toolkit that provides signal processing blocks to implement software radios. GNU Radio Companion (GRC) is included in the GNU Radio installation and it provides a simple graphical interface. In order to install the GRC, the following command was run from the terminal: `sudo apt install gnuradio`. In addition to this, we installed the Osmocom source that allows the HackRF One to enter the receive mode and communicate with other devices. The following command installed the Osmocom source: `sudo apt install gr-osmosdr`.

In order to be able to receive packets using the RTL-SDR dongle, we installed a package called `rtl-zwave` by running the following commands:

```
git clone https://github.com/andersesbensen/rtl-zwave.git
cd rtl-zwave
make
make install
```

Moreover, an ITU G.9959 (de)modulator package was installed for being able to encode and decode ITU G.9959 frames implemented by the Z-Wave protocol using the RTL-SDR dongle and the HackRF One. The only available open source package for this purpose is the `Waving-Z`, a simple FSK modulator:

```
git clone https://github.com/baol/waving-z
cd waving-z/
mkdir build
cd build
cmake .. -DCMAKE_BUILD_TYPE=Release
cmake --build .
```

Finally, we installed the Salae Logic Setup software that we used with the Logic Analyzer. An optional software was installed for measuring the exact frequency of the Z-Wave, the SDR-Sharp (sdr#). Z-Wave documentation declares that the Z-Wave operates on 868.42 MHz frequency in the EU, however using the sdr# we retrieved the exact frequency for our Z-Wave environment.

Hardware	Software
HackRF One	hackrf
	gnuradio-companion
	gr-osmosdr
RTL-SDR dongle	rtl-zwave
	waving-z
Logic Analyzer	Salae logic setup
Digital Multimeter	-

Table 4.1: Hardware and Software used in this research

4.3.2 Protocol Hacking

In order to eavesdrop on the Z-Wave network, we used the HackRF One and with conjunction with the GNU Radio we created a listener block for listening to the Z-Wave data. Using this method, we tried to perform replay attacks on the devices. First, we added the osmocom Source module and we specified the frequency we wanted to listen to (Ch0 Frequency was set to 868.42 MHz). We modified the sample rate to 2 millions, that sets the number of samples per second the HackRF One receives (samp_rate was set to 2M). Then, a QT GUI Waterfall Sink and a File Sink were added to our block. We connected the osmocom Source to both sinks. The Waterfall Sink displayed the changes in the frequency while listening to the Z-Wave devices and with the File Sink, the captured data was saved to a predefined file. Figure 4.9 displays the block we created to listen to the Z-Wave devices.

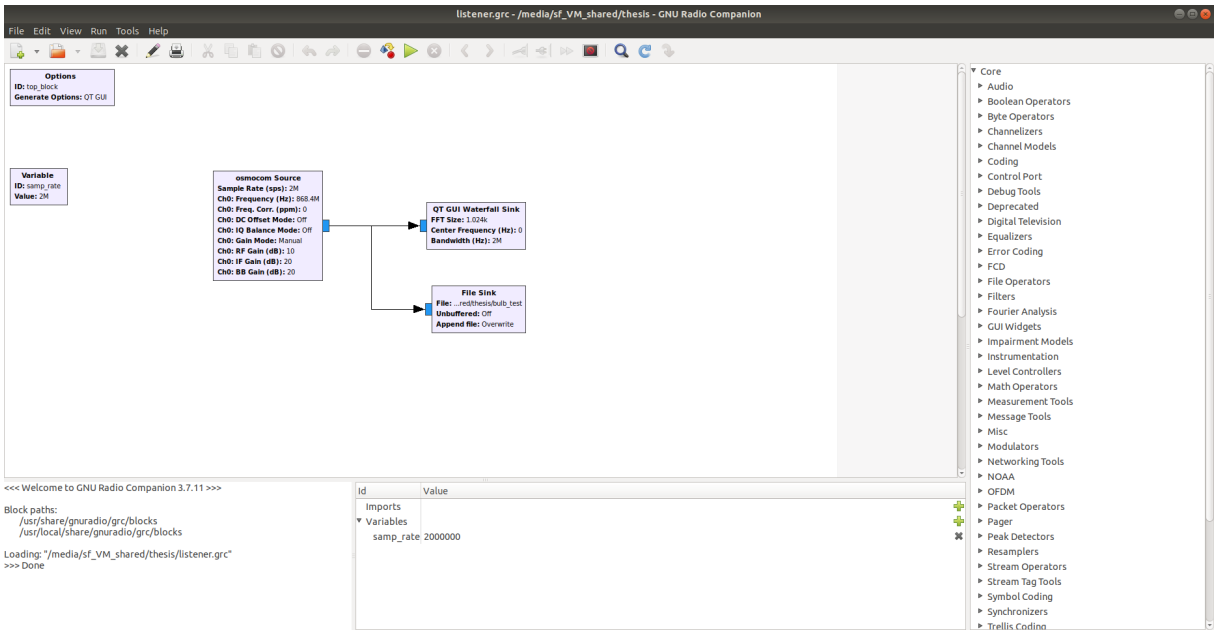

Figure 4.9: GNU Radio Companion for listening

Once we captured the data, we created another block on the GNU Radio, this time for replaying the captured data back to the devices. We added the File Source and we set as an input file, the file we saved the data during the listening. We connected the File Source to the QT GUI Waterfall Sink, for having a visual graph of the frequency the same as before, and to the osmocom Sink function. The settings for the osmocom Sink were to alter the frequency and set the *Ch0: RF Gain (db)* to 0 and the *Ch0: IF Gain (db)* to 47 for amplifying the signal the HackRF One used to send to the devices. Figure 4.10 displays the block we created to replay the packet to the Z-Wave devices.

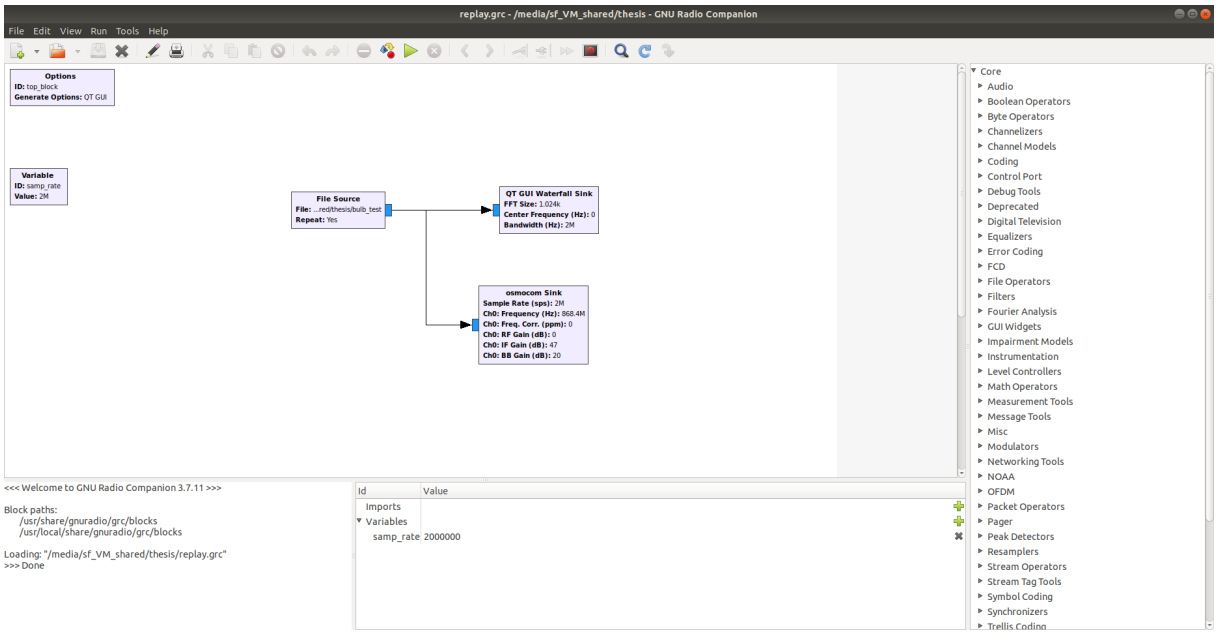

Figure 4.10: GNU Radio Companion for replay

Another tool we used is the Waving-Z and the RTL-SDR dongle for capturing and decoding Z-Wave packet frames. After we plugged the dongle on the computer the following command was ran on the terminal: `rtl_sdr -f 868420000 -s 2000000 -g 25 - | ./wave-in -u` in the directory of the *waving-z/build/*. After capturing data frames for a certain period of time, while sending commands from the controller to the devices, we terminated the capturing part. We analyzed the results to see if we could identify a Z-Wave data packet. Once we had the packet we decoded it and saved it as an `.cs8` file, which is a Unix graphics format, using the following command: `./wave-out -p 'ff ff ff ff ff ff ff ff ff ff ff ff' > hack_device.cs8`. The parameter `p` defines the Z-Wave data frame. Using the HackRF One, we tried to transmit the packet to the device to test if we could take control over the network without using the controller. `hackrf_transfer -f 868420000 -s 2000000 -t hack_device.cs8` used as the transmission command.

The experiments on the protocol hacking were conducted multiple times on both the devices, changing the parameters each time, the security level of the Z-Wave protocol and the time frame of capturing. The results are presented in Chapter 5.

4.3.3 Hardware Hacking

Hardware hacking is the process of verifying whether the hardware of the devices contains any sensitive information that an attacker can use to expose the security or privacy of the users. For testing the devices on the hardware level, the Z-Wave USB controller and the smart door lock will be opened and analyzed. The light bulb will not be opened because this will lead to breaking the bulb apart and it will be of no use anymore.

The first step before assessing the devices, is to open them and to check if there are any tamper proof mechanisms and if the PCB can be reached with ease. In case of the door lock, it is separated into two parts, the inner part and the outer part. The two parts are connected with a wire and they both contain a board. The boards were already visible since there is no protection cover on the device. Some typical screws are keeping the boards on the lock, but they can easily be unscrewed. Once, we separated the boards from the lock, we performed some physical inspection to identify the different components and services on the boards.

The pins were checked with the digital multimeter to identify which protocol (or protocols) they are using for communication, and whether there was any voltage on them that could indicate data flowing through the pins. The device uses 1-Wire and Serial Peripheral Interface (SPI) protocols for transmitting the data. Activity found on some of the pins that might contain some data flow. In order to validate our assumption, the logic analyzer was used. The figure 4.11 below shows the pins connecting to the logic analyzer, which in turn was connected to the computer. We ran the Salae Logic Setup software to display and analyze the data signals. The results of this analysis are presented later in Chapter 5.

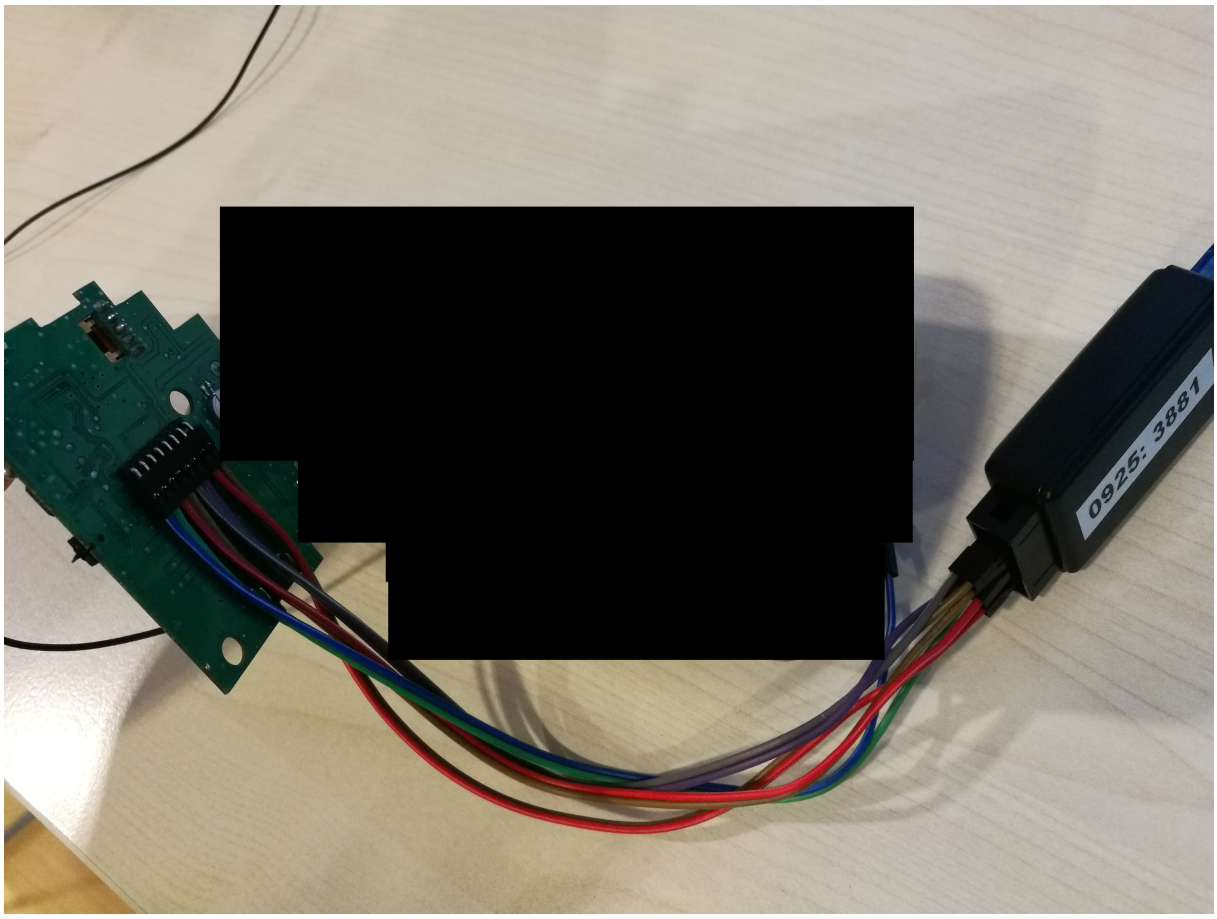

Figure 4.11: Use of Logic Analyzer

After, the assessment of the door lock was finished, the Z-Wave controller was opened. There was a protection cover but no screws were found on the USB. Hence, it was not that difficult to carefully open the device and reveal the board. The inside of the controller can be seen on figure 4.12.

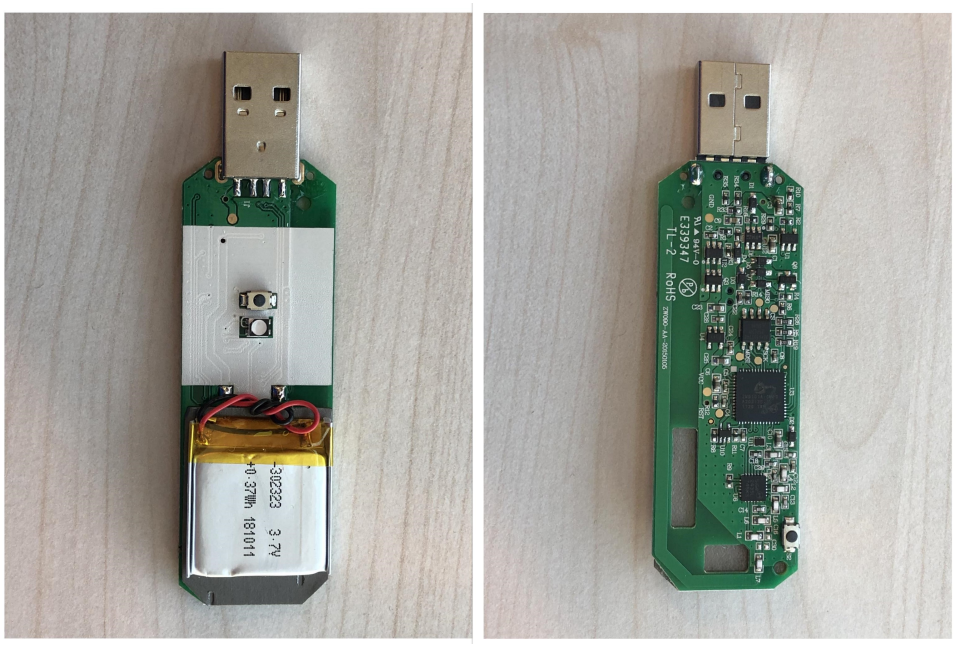

Figure 4.12: The inside of the Z-Wave controller

Following the methods used for the door lock, the controller's assessment began with investigating the components on the board. The Z-Wave chip 500 series and an EEPROM memory chip were identified. However, there were no pins for further investigations. The results from the Z-Wave devices hardware hacking are presented in the following chapter.

5 Results

In this chapter, the results obtained from the carried out experiments are presented. The results are classified into the two main aspects of this research, security and privacy aspects.

The initial experiment that was conducted was to try to perform a replay attack on the devices. First, we started with the light bulb that was included into the Z-Wave network without security. Using the GNU Radio Companion for listening, we captured the data sent from the controller, refer to figure 5.1. The horizontal yellow lines on the graph indicate the frequency while sending a command from the controller to the bulb. The data afterwards was replayed and sent back to the light bulb using the HackRF One. The attack conducted successfully, as we were able to turn ON/OFF the light and also change the colors. Summarizing, without encryption, all the class commands were captured and sent successfully to the slave device.

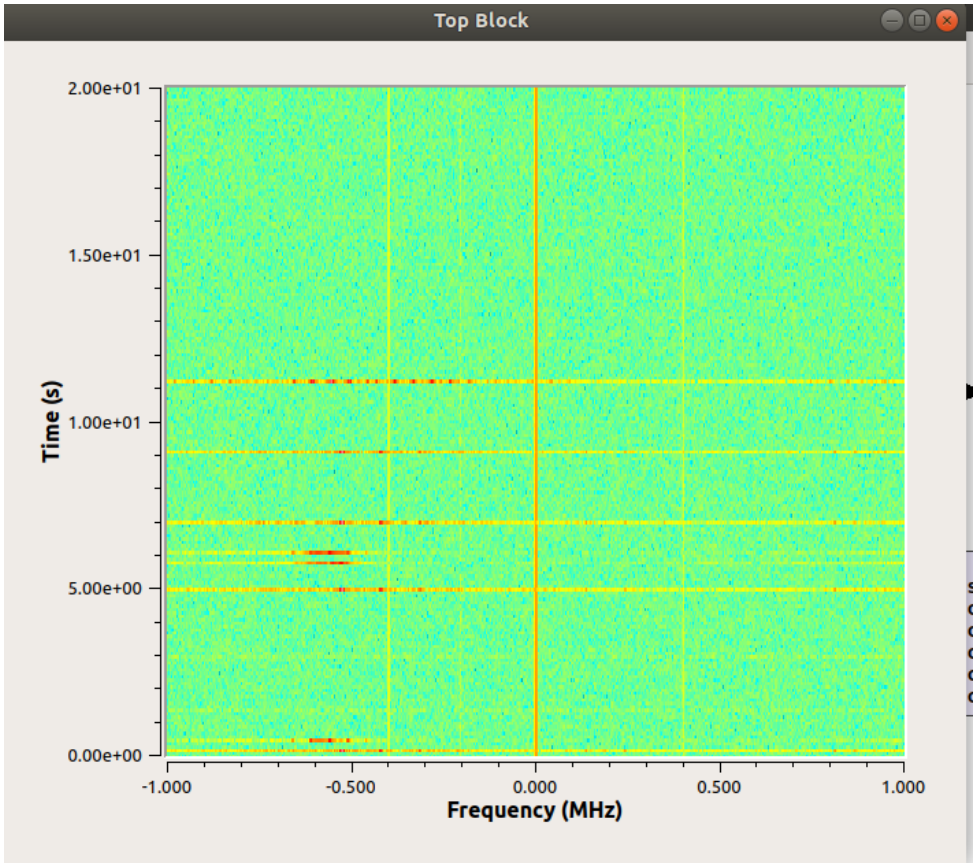

Figure 5.1: Data flow graph of the light bulb

The same experiment was conducted with the door lock, that uses the S0 security framework. Once we captured the packet, we tried to replay it, but the attack was unsuccessful. We tried multiple times, again with the same result. This happened because the devices uses the `COMMAND_CLASS_SECURITY`, as shown on the debug log file, figure 5.2. In order to validate that Z-Wave S0 security framework is resistant to replay attacks, we tested the light bulb, but this time using S0 security for this device as well. The bulb had to be excluded from the network. The secure inclusion mode on the controller changed to "All Devices" and then paired with the bulb. The result of the experiment on the bulb using S0 security was negative. The bulb was now secure against replay attack.

```
[DEBUG] [nal.protocol.ZWaveTransactionManager] - NODE 9: Application Command Request (ALIVE:DYNAMIC_VALUES)
[DEBUG] [ng.zwave.internal.protocol.ZWaveNode] - NODE 9: Decapsulating COMMAND_CLASS_SECURITY
[DEBUG] [mmandclass.ZWaveSecurityCommandClass] - NODE 9: SECURITY_RXD 8B 03 07 D0 01 01 00 0D 33
[DEBUG] [ng.zwave.internal.protocol.ZWaveNode] - NODE 9: Incoming command class COMMAND_CLASS_TIME_PARAMETERS, endpoint 0
[DEBUG] [tocol.commandclass.ZWaveCommandClass] - NODE 9: Received COMMAND_CLASS_TIME_PARAMETERS V1 TIME_REPORT
[DEBUG] [lass.ZWaveTimeParametersCommandClass] - NODE 9: Received time report: Sat Jan 01 00:13:51 CET 2000
[DEBUG] [ding.zwave.handler.ZWaveThingHandler] - NODE 9: Got an event from Z-Wave network: ZWaveCommandClassValueEvent
[DEBUG] [ding.zwave.handler.ZWaveThingHandler] - NODE 9: Got a value event from Z-Wave network, endpoint = 0, command class = COMMAND_CLASS_TIME_PARAMETERS, value = Sat Jan 01

[DEBUG] [nal.protocol.ZWaveTransactionManager] - NODE 9: Commands processed 1.
[DEBUG] [nal.protocol.ZWaveTransactionManager] - NODE 9: Checking command org.openhab.binding.zwave.internal.protocol.ZWaveCommandClassPayload@591f7fd6.
[DEBUG] [nal.protocol.ZWaveTransactionManager] - NODE 9: Command verified org.openhab.binding.zwave.internal.protocol.ZWaveCommandClassPayload@591f7fd6.
[DEBUG] [nal.protocol.ZWaveTransactionManager] - NODE 9: notifyTransactionResponse TID:530 DONE
[DEBUG] [ding.zwave.handler.ZWaveThingHandler] - NODE 9: Got an event from Z-Wave network: ZWaveTransactionCompletedEvent
[DEBUG] [nal.protocol.ZWaveTransactionManager] - Transaction completed - outstandingTransactions 1
[DEBUG] [nal.protocol.ZWaveTransactionManager] - Transaction completed - outstandingTransactions 0
[DEBUG] [nal.protocol.ZWaveTransactionManager] - ZWaveReceiveThread queue empty
```

Figure 5.2: Debug log file

The next experiment we conducted, was to capture packet frames using the Waving-Z, the ITU G.9959 de-modulator, and the RTL-SDR dongle for receiving. Figure 5.3 displays the packet captured when the command to turn ON sent to the light bulb. Since we had the whole packet frame decoded, wee saved it to a `.cs8` file using the command:

`./wave-out -p 'e3 e6 7e b9 01 51 0d 0d 0d 26 01 ff' > bulb.cs8`.

Then using the HackRF One, we transmitted the packet back to the bulb, and the bulb turned ON. By knowing the packet frame, we were able to hack and take control over the light bulb. Being able to understand the Z-Wave packet we captured, we analyzed the frame and found that the command class is `0x26`. Checking the datasheet of the Z-Wave commands provided by Silicon Labs, command class `0x26` is a `COMMAND_CLASS_SWITCH_MULTILEVEL` and knowing that the command to turn ON the light was `0x01ff`, we hypothesized that the command to turn OFF the light would be `0x0100`. We tried to send this packet frame and the light bulb indeed turned 0FF.

```
vassilis@vm-ubuntu:~/Downloads/waving-z/build$ rtl_sdr -f 868420000 -s 2000000 -g 15 - | ./wave-in -s 2000000 -u
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Detached kernel driver
Found Rafael Micro R820T tuner
Exact sample rate is: 2000000.052982 Hz
[R82XX] PLL not locked!
Sampling at 2000000 S/s.
Tuned to 868420000 Hz.
Tuner gain set to 14.40 dB.
Reading samples in async mode...

[x] HomeId: e3e67eb9, SourceNodeId: 1, FC0: 51, FC1: d, FC[speed=1 low_power=0 ack_request=1 header_type=1 beaming_info=0 seq=13], Length: 13, DestNodeId: 13, CommandClass: 26, Payload: 01 ff
```

Figure 5.3: Bulb Z-wave packet frame

The experiment was conducted multiple times on the bulb and for a longer period of time, and from the packets we received, we were able to determine whether the light is turned ON/OFF and whether the colors had changed.

The next step was to perform the same attack on the door lock. Figure 5.4 shows a multiple packet frames captured when a command was sent from the controller to the lock. The command class `0x98` that was sent, refers to `COMMAND_CLASS_SECURITY`. Hence, we were unable to decode the command and attack the lock. The experiment was conducted multiple times, increasing the period of capturing, but no usable command was captured.

```
vassilis@vm-ubuntu:~/Downloads/waving-z/build$ rtl_sdr -f 868420000 -s 2000000 -g 15 - | ./wave-in -s 2000000 -u
Found 1 device(s):
  0:  Realtek, RTL2838UHIDIR, SN: 00000001

Using device 0: Generic RTL2832U OEM
Detached kernel driver
Found Rafael Micro R820T tuner
Exact sample rate is: 2000000.052982 Hz
[R82XX] PLL not locked!
Sampling at 2000000 S/s.
Tuned to 868420000 Hz.
Tuner gain set to 14.40 dB.
Reading samples in async mode...
e3 e6 7e b9 01 41 02 0c 09 98 40 a2 00 00
[x] HomeId: e3e67eb9, SourceNodeId: 1, FC0: 41, FC1: 2, FC[speed=0 low_power=0 ack_request=1 header_type=1 beaming_info=0 seq=2], Length: 12, DestNodeId: 9, CommandClass: 98, Payload: 40
e3 e6 7e b9 09 03 00 00
[ ]
e3 e6 7e b9 01 41 06 20 09 98 81 ba dd 75 05 fc cb 71 01 36 75 44 39 2c 65 74 23 8d 04 ec 5c 02 00 00
[x] HomeId: e3e67eb9, SourceNodeId: 1, FC0: 41, FC1: 6, FC[speed=0 low_power=0 ack_request=1 header_type=1 beaming_info=0 seq=6], Length: 32, DestNodeId: 9, CommandClass: 98, Payload: 81 ba
  cb 71 01 36 75 44 39 2c 65 74 23 8d 04 ec 5c
e3 e6 7e b9 09 03 06 0a 01
[ ]
e3 e6 7e b9 09 41 46 0c 01 98 40 e6 40 00 00
[x] HomeId: e3e67eb9, SourceNodeId: 9, FC0: 41, FC1: 46, FC[speed=0 low_power=0 ack_request=1 header_type=1 beaming_info=4 seq=6], Length: 12, DestNodeId: 1, CommandClass: 98, Payload: 40
e3 e6 7e b9 01 41 08 0c 09 98 40 a8 00 00
[x] HomeId: e3e67eb9, SourceNodeId: 1, FC0: 41, FC1: 8, FC[speed=0 low_power=0 ack_request=1 header_type=1 beaming_info=0 seq=8], Length: 12, DestNodeId: 9, CommandClass: 98, Payload: 40
```

Figure 5.4: Lock Z-wave packet frame

The validation required the experiment to run on the light bulb using the S0 security. Following the steps described above for pairing the light bulb with the controller securely, we tried to capture new packet frames from the bulb, but the only remarkable command class we received was the `COMMAND_CLASS_SECURITY`. Z-Wave S0 security level is resilient to this type of attack. However, the Home ID was always captured in every packet in plain text, even the security was

enabled.

During the hardware hacking the controller was investigated and the datasheets of the components on the board found. We discovered that the bus protocol used to read and write to the internal flash memory is the SPI. However, no pin heads found on the board, making it difficult to read the data stored on its memory. The only reason to achieve this was to de-solder the memory chip and using an adaptor to read its content. We did not proceed on this, because there was a chance of destroying the controller.

For the door lock, the pins analyzed as described in the previous chapter, using a logic analyzer. The results revealed that the bus protocols used for communication are the 1-Wire and SPI. 1-Wire is a device communications bus system that provides low-speed data and signaling and used for transferring the data from the inner part of the lock to the outer and vice versa. SPI is used to read and write to the internal flash memory. Figure 5.5 shows the data signals from the SPI analyzer.

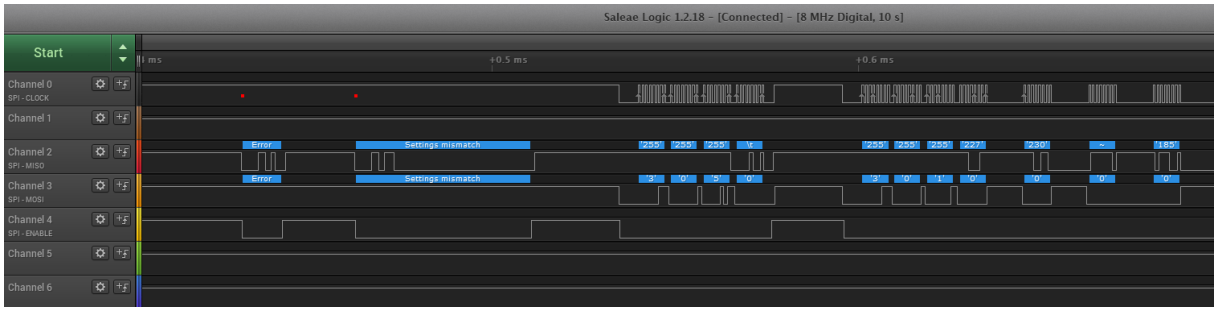

Figure 5.5: Logic Analyzer for SPI

However, the data from the logic analyzer was not usable. No information retrieved from the SPI capturing. The experiment was conducted several times with the same result. The final step for the door lock investigation, was to use the 1-Wire analyzer on the logic analyzer and try to read the data send from one part of the lock to the other. The result of this analysis is quite interesting because it revealed a vulnerability. The outcome is described in the appendix and since the responsible disclosure to the manufacturer company is pending, it will remain confidential.

5.1 Conclusion

In this chapter, we presented the results obtained from the experiments that were carried out for exploring the security and privacy aspects of the Z-Wave protocol. The results showed that using no security on the Z-Wave devices, may lead to serious risks for the users in a smart home environment, putting both their security and privacy under question. An attacker can replay the data sent from the controller to the devices and take control over the Z-Wave network. As shown during the experiments, an attacker can turn ON/OFF the light bulb. However, using security, performing a replay attack is not feasible, because half of the IV is send from one end to another, and then the other end uses the other half of the IV, which means the packets have to be fresh. Hence, it is impossible to spoof a packet or replay a packet.

Privacy is a major topic when it comes to IoT and especially to smart homes. Using the packets captured from the Z-Wave network, an attacker can specify a user's daily behaviour. The packets that captured from the light bulb, showed when the user turned the light ON or OFF anytime during the day. Any domestic activity can be recorded and sensitive information can be revealed to malicious people.

Another issue that was discovered during the Z-Wave assessment, was that the Home ID, which is unique for every Z-Wave network, always appeared in the captured packets. Even with security enabled, an attacker could easily discover the Home ID. Rouch et al. 2017 have proved that by finding the Home ID of a Z-Wave network, an attacker can create an adversary controller and take over the Z-Wave network. This is an issue of the ITU-T G.9959 specification that Z-Wave uses and it is too easy to jam the radio and has also some security issues too.

Taking the aforementioned into account, the security is more than necessary within a Z-wave environment. Even the S0 framework can preserve the security and privacy of the end-users. Despite the security that Z-Wave offers to the devices, it is up to the end-user to implement it or not.

6 Discussion

After the background research and carrying out the experiments, the results from the previous chapter and the challenges faced with exploring the security and privacy aspects of the Z-Wave protocol will be discussed here.

Creating the Z-Wave network with the controller and OpenHab, has shown that the inclusion of some devices can be done without the security enabled. This is a major threat for both the security and privacy of the users, as they are not aware of this issue. The smart light bulb was paired with the controller without security and therefore without using encryption, by default. An attacker could leverage from this and perform a replay attack on Z-Wave devices or, even worse, take control over the network.

Capturing Z-Wave packet frames and decoding them, allowed us to reach a similar conclusion. Using a simple ITU-T G.9959 de-modulator, packet frames were captured in plain text, allowing us to leverage and take control over the Z-Wave network. Since the light bulb that was not secured by default, was transmitting the packets to and from the controller without encryption. The captured packets were enough to allow an attacker to take control over the lighting in a smart home environment. The same did not happen with the lock, as the data payload frame used the security commands of the S0 framework for encrypting the message.

Furthermore, the unawareness of the security risks and how to enable security over the Z-Wave network, has an impact on the users' privacy as well. An attacker within the radio range, can easily monitor the daily behaviour of the inhabitants within a smart home and acquire useful and sensitive personal data. The results from the experiments during the monitoring of the Z-Wave network, have revealed all the activities that took place involving insecure devices. In a real time smart home environment, we could easily determine whether the inhabitants are present or absent by simply watching over some smart devices.

The security level that was supported by the devices within the Z-Wave network during the experiments was the S0. Only available open source tools for capturing and analyzing Z-Wave

packets were used for this research. However, for further investigation on the encryption layer, a Z-Wave sniffer is needed and the only reliable sniffer for this protocol is the Z-Wave Zniffer provided by Silicon Labs. The Z-Wave Zniffer comes with the development kit and the cost of it is approximately €1000. In this kit, a Z-Wave bridge controller that supports the S2 security framework is included, the only available controller that supports S2 security in practice.

The hardware hacking, also disclosed that the security on the physical level is crucial. The Z-Wave door lock did not bring any protective cover, leaving the board exposed to everyone. Sensitive information, like keys and passwords, are most likely to be stored on the memory of the device. However, for practical reasons and time limitation, we did not focus on extracting the data from the devices' memory. This would imply to take the chip out of the board and try to read it separately. However, this would put the device in a risk of breaking it and not be able to use it anymore. For the same reasons, the light bulb was not opened.

Initial investigation on the security features during the inclusion process, was carried out and the outcome has shown that for most of the devices that are not considered entry control devices, security in not enabled by default. The fundamental recommendation is to use the security command class for all the devices. The Z-Wave controller offers the users a chance to alter the Home ID and the network key of their Z-Wave network, by hard resetting the controller. This is something rare for the smart home infrastructure. Users can use this option to alter the most important properties of the Z-Wave, for their benefit and keep attackers outside of their smart home.

Finally, a controversial topic about Z-Wave is the correlation between the Z-Wave Plus technology and the security. The S2 security framework and Z-Wave Plus technology are two different terms and they are independent. New Z-Wave devices have to implement the latest security level and since they are equipped with the 500 series chips, it seems that every Z-Wave Plus devices uses S2 security and vice versa. However, this is not always true.

7 Conclusion and Future Work

In this thesis, we examined and evaluated the Z-Wave communication protocol based on its security and privacy aspects. After a background research on the threats in smart homes and a detailed description about the Z-Wave protocol, some experiments were conducted in order to assess the the security implementations of the protocol. In the conclusion, this chapter explains the research questions and how they were answered. Future work outlines the remaining challenges and directions for future research.

7.1 Conclusion

In order to answer the primary research question of this thesis, five subquestions were formulated.

1. *What is the architecture of the Z-Wave wireless protocol and what are its security features?*

   The architecture of the Z-Wave protocol was described in Chapter 3, providing a detailed overview of the protocol's stack and its security principles. Security plays an important role for a Z-Wave network. The Z-Wave protocol supports two levels of security, with the latest one (S2 Security) developed to fix some of the vulnerabilities and limitations of the S0 security framework. The biggest security issue of the first security level (S0), is that the devices within a Z-Wave network share the same network key. This implies that if an attacker is able to hack and obtain the key from one device, then he or she can take control over the whole Z-Wave network. S2 level bypasses this issue with the implementation of multiple shared keys.

2. *What kind of information is leaked through wireless communication with the Z-Wave protocol?*

   The assessment of the Z-Wave network took place in two phases. The first one without the security enabled, as it was by default to non entry control devices, and the second with the S0 security enabled. The information that is leaked through the communication, is the Home ID and Node ID of the Z-Wave network. Without security, it is obvious that

the data frame is also leaked. This involves the commands sent from the controller to the end devices.

3. *How can eavesdroppers attack the Z-Wave enabled smart devices?*

   It was shown from the experiments that if security is not enabled, an eavesdropper can perform replay attack on the Z-Wave devices, as well as send packets to control the devices. If the devices do not use encryption, the messages are transmitted in clear text. However, even with the S0 security, the Home ID which is the most important Z-Wave property can be captured from an attacker. Privacy is also under risk when users do not take advantage of the Z-Wave security features. As shown during the experiments, the users' daily home activities can be monitored from an outsider.

4. *What factors lead to the leakage of information or the abuse of the devices?*

   It was consistently shown from the experiments that the use of the Z-Wave S0 security can prevent the leakage of information. Although the S0 security can preserve security and privacy to a great extent, the unique identifiers, Home ID and Node ID, are still leaked through the wireless communication.

   The factors that led to the abuse of the door lock, were the absence of any tamper proof mechanisms and the ability to reach the PCB with ease. Once the pins were spotted, we were able to connect to the device and analyze the data sent. As shown on the results of the experiments, the abuse of the door lock led us to discover a vulnerability that allowed to brute force it. For the Z-Wave controller, no pins were spotted on the PCB, preventing us from physically abusing the device.

5. *What are the applicable countermeasures against the security and privacy issues of the Z-Wave protocol?*

   The fundamental recommendation for the users in order to preserve their security and privacy against the threats of the Z-Wave protocol, is to use the security command class for all the devices. Security is needed for the Z-Wave network in order to preserve security and privacy within a smart home. Users can also leverage the option of resetting the Z-

Wave USB controller. Every time there is a hard reset done on the controller, the value of the Home ID and the shared network key alter. This countermeasure prevents an attacker for continuously capturing data from the Z-Wave network, and therefore abusing users' privacy. Furthermore, by altering the identifiers often, attackers cannot send old packets that might have obtained, for attacking the Z-Wave devices.

Another recommendation for the users should be to choose devices that support Z-Wave Plus technology. A Z-Wave Plus device uses the latest generation of chips (500 series) which has been tested under the Z-Wave Alliance Certification program. In addition to that, security S2 is required alongside with the Z-Wave Plus devices, in order to strengthen the security of the network.

Apart from the Z-wave specifications, the physical quality of the device is also very important when setting up a secure and robust Z-Wave network. Users should not choose the Z-Wave products based only on the Z-Wave characteristics or the price of the device, but also on the quality of the product. As shown during the experiments, a crucial vulnerability was discovered on the door lock, which was purchased based on the price and not on the quality of the product.

7.2 Future Work

Using the available and open-source tools and a low budget of hardware, we explored the security and privacy aspects of the Z-Wave communication protocol. However, a further and more reliable investigation on the encryption level can be done only by using the Z-Wave Zniffer provided by the Z-Wave Alliance. As a future recommendation, the Z-Wave Zniffer should be used to assessing further the security of the Z-Wave protocol.

In the future, Z-Wave devices that allow the S2 security class should be tested for security or privacy issues within a smart home environment. Silicon Labs announced the new Z-Wave generation of hardware (700 series) will be available this year. Z-Wave 700 ensure best-in-class security with built-in Security 2 (S2). Hence, a future security analysis on a Z-Wave network

with 700 series devices that support the S2 Security would be valuable.

A deeper hardware hacking on the existing Z-Wave devices covering the 300 and 500 series and comparing these two Z-Wave chips can be performed as a future research. With the 700 series almost released, a researcher can also add new generation of Z-Wave hardware to the experiment.

Finally, another important future work that needs to be done, is to analyze the open source project that provides a framework for an IoT system used for this thesis, the OpenHAB. Since it is open source project written in Java, a future work would be to dive into its architecture and check its code for potential vulnerabilities that can be exploited and threaten the users' daily life with a smart home.

References

Ashton, Kevin et al. (2009). "That 'internet of things' thing". In: *RFID journal* 22.7, pp. 97–114.

Evans, Dave (2011). "The internet of things: How the next evolution of the internet is changing everything". In: *CISCO white paper* 1.2011, pp. 1–11.

Islam, Kamrul, Weiming Shen, and Xianbin Wang (2012). "Security and privacy considerations for wireless sensor networks in smart home environments". In: *Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference on*. IEEE, pp. 626–633.

Rehman, Shafiq, Volker Gruhn, et al. (2018). "An Effective Security Requirements Engineering Framework for Cyber-Physical Systems". In: *Technologies* 6.3, p. 65.

Apthorpe, Noah et al. (2017). "Spying on the smart home: Privacy attacks and defenses on encrypted iot traffic". In: *arXiv preprint arXiv:1708.05044*.

Lewis, Dave (2017). *The DDoS Attack Against Dyn One Year Later*. URL: `https://www.forbes.com/sites/davelewis/2017/10/23/the-ddos-attack-against-dyn-one-year-later/#392c64651ae9`.

Glisson, William Bradley et al. (2015). "Compromising a medical mannequin". In: *arXiv preprint arXiv:1509.00065*.

Storm, Darlene (2015). "Researchers hack a pacemaker, kill a man (nequin)". In: URL: `https://www.computerworld.com/article/2981527/cybercrime-hacking/researchers-hack-a-pacemaker-kill-a-man-nequin.html`.

Jacobsson, Andreas, Martin Boldt, and Bengt Carlsson (2016). "A risk analysis of a smart home automation system". In: *Future Generation Computer Systems* 56, pp. 719–733.

Mahmoud, M.S. and Auday A.H. Mohamad (2016). "A study of efficient power consumption wireless communication techniques/modules for internet of things (IoT) applications". In: *Advances in Internet of Things, 6*, pp. 19–29.

Chan, Marie et al. (2009). "Smart homes—current features and future perspectives". In: *Maturitas* 64.2, pp. 90–97.

Acar, Abbas et al. (2018). “Peek-a-Boo: I see your smart home activities, even encrypted!” In: *arXiv preprint arXiv:1808.02741*.

Bormann, Carsten, Mehmet Ersue, and Ari Keranen (2014). *Terminology for constrained-node networks*. Tech. rep. URL: `https://tools.ietf.org/html/rfc7228`.

Lévy-Bencheton, Cédric et al. (2015). “Security and resilience of smart home environments good practices and recommendations”. In: *ENISA Google Scholar*.

Batalla, Jordi Mongay, Athanasios Vasilakos, and Mariusz Gajewski (2017). “Secure smart homes: Opportunities and challenges”. In: *ACM Computing Surveys (CSUR)* 50.5, p. 75.

Al-Sarawi, Shadi et al. (2017). “Internet of Things (IoT) communication protocols”. In: *Information Technology (ICIT), 2017 8th International Conference on*. IEEE, pp. 685–690.

Ray, Abhay Kumar and Ashish Bagwari (2017). “Study of smart home communication protocol’s and security & privacy aspects”. In: *2017 7th International Conference on Communication Systems and Network Technologies (CSNT)*. IEEE, pp. 240–245.

Yassein, Muneer Bani, Wail Mardini, and Ashwaq Khalil (2016). “Smart homes automation using Z-wave protocol”. In: *Engineering & MIS (ICEMIS), International Conference on*. IEEE, pp. 1–6.

Mendes, Tiago et al. (2015). “Smart home communication technologies and applications: Wireless protocol assessment for home area network resources”. In: *Energies* 8.7, pp. 7279–7311.

Fernandes, Earlence et al. (2017). “Security implications of permission models in smart-home application frameworks”. In: *IEEE Security & Privacy* 15.2, pp. 24–30.

Gyory, Nathaniel and M Chuah (2017). “IoTOne: Integrated platform for heterogeneous IoT devices”. In: *Computing, Networking and Communications (ICNC), 2017 International Conference on*. IEEE, pp. 783–787.

Kalin, John Howard (2017). “Simulating IoT Frameworks and Devices in the Smart Home”. PhD thesis. Virginia Tech.

Visoottiviseth, Vasaka et al. (2017). “PENTOS: Penetration testing tool for Internet of Thing devices”. In: *Region 10 Conference, TENCON 2017-2017 IEEE*. IEEE, pp. 2279–2284.

Chen, Chung-Kuan et al. (2018). “Penetration Testing in the IoT Age”. In: *Computer* 51.4, pp. 82–85.

OWASP (2019a). *Internet of Things Project*. URL: https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project.

– (2019b). *Top IoT Vulnerabilities*. URL: https://www.owasp.org/index.php/Top_IoT_Vulnerabilities.

– (2019c). *IoT Security Guidance*. URL: https://www.owasp.org/index.php/IoT_Security_Guidance.

Gomez, Carles and Josep Paradells (2010). "Wireless home automation networks: A survey of architectures and technologies". In: *IEEE Communications Magazine* 48.6.

Zillner, Tobias and S Strobl (2015). "ZigBee exploited: The good the bad and the ugly". In: *Black Hat–2015*.

Fouladi, Behrang and Sahand Ghanoun (2013). "Security evaluation of the Z-Wave wireless protocol". In: *Black hat USA* 24, pp. 1–2.

Schwarz, Daniel (2016). "The Current State of Security in Smart Homes Systems". In: *SEC Consult Vulnerability Lab, Vienna*.

Arias, Orlando et al. (2015). "Privacy and security in internet of things and wearable devices". In: *IEEE Transactions on Multi-Scale Computing Systems* 1.2, pp. 99–109.

Ho, Grant et al. (2016). "Smart locks: Lessons for securing commodity internet of things devices". In: *Proceedings of the 11th ACM on Asia conference on computer and communications security*. ACM, pp. 461–472.

Komninos, Nikos, Eleni Philippou, and Andreas Pitsillides (2014). "Survey in smart grid and smart home security: Issues, challenges and countermeasures". In: *IEEE Communications Surveys & Tutorials* 16.4, pp. 1933–1954.

Sivaraman, Vijay et al. (2015). "Network-level security and privacy control for smart-home IoT devices". In: *Wireless and Mobile Computing, Networking and Communications (WiMob), 2015 IEEE 11th International Conference on*. IEEE, pp. 163–167.

Bugeja, Joseph, Andreas Jacobsson, and Paul Davidsson (2016). "On privacy and security challenges in smart connected homes". In: *Intelligence and Security Informatics Conference (EISIC), 2016 European*. IEEE, pp. 172–175.

Hoang, Nguyen Phong and Davar Pishva (2015). "A TOR-based anonymous communication approach to secure smart home appliances". In: *Advanced Communication Technology (ICACT), 2015 17th International Conference on*. IEEE, pp. 517–525.

Mombrea, M (2014). "Googles real plan behind the purchase of the nest thermostat". In: *Retrieved April* 12, p. 2015.

Arabo, Abdullahi, Ian Brown, and Fadi El-Moussa (2012). "Privacy in the age of mobility and smart devices in smart homes". In: *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*. IEEE, pp. 819–826.

Notra, Sukhvir et al. (2014). "An experimental study of security and privacy risks with emerging household appliances". In: *Communications and Network Security (CNS), 2014 IEEE Conference on*. IEEE, pp. 79–84.

Zheng, Serena et al. (2018). "User Perceptions of Smart Home IoT Privacy". In: *Proceedings of the ACM on Human-Computer Interaction* 2.CSCW, p. 200.

Lee, Changmin et al. (2014). "Securing smart home: Technologies, security challenges, and security requirements". In: *Communications and Network Security (CNS), 2014 IEEE Conference on*. IEEE, pp. 67–72.

Labs, Silicon (2018). *Silicon Labs Completes Acquisition of Sigma Designs' Z-Wave Business - Apr 18, 2018*. URL: `https://news.silabs.com/2018-04-18-Silicon-Labs-Completes-Acquisition-of-Sigma-Designs-Z-Wave-Business`.

Rouch, Loïc et al. (2017). "A Universal Controller to Take Over a Z-Wave Network". In: *Black Hat Europe 2017*, pp. 1–9.

ITU-T, International Telecommunication Union (2015). *Recommendation ITU-T G.9959*. URL: `https://www.itu.int/rec/T-REC-G.9959-201501-I/en`.

Labs, Silicon (2019). *Software Design Specification*. `https://www.silabs.com/documents/login/miscellaneous/SDS13781-Z-Wave-Application-Command-Class-Specification.pdf`.