3-24-2016

A Misuse-Based Intrusion Detection System for ITU-T G.9959 Wireless Networks

Jonathan D. Fuller

A MISUSE-BASED INTRUSION DETECTION SYSTEM FOR ITU-T G.9959 WIRELESS NETWORKS

THESIS

Jonathan D. Fuller, Captain, USA

AFIT-ENG-MS-16-M-016

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-16-M-016

A MISUSE-BASED INTRUSION DETECTION SYSTEM FOR ITU-T G.9959 WIRELESS NETWORKS

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Science

Jonathan D. Fuller, BS

Captain, USA

March 24, 2016

AFIT-ENG-MS-16-M-016

A MISUSE-BASED INTRUSION DETECTION SYSTEM FOR ITU-T G.9959 WIRELESS NETWORKS

THESIS

Jonathan D. Fuller, BS
Captain, USA

Committee Membership:

Maj Benjamin W. Ramsey, PhD
Chair

LTC Mason J. Rice, PhD
Member

Maj John M. Pecarina, PhD
Member

Abstract

Wireless Sensor Networks (WSNs) are becoming ubiquitous providing low-cost, low-power, and low-complexity systems where control and communication are tightly integrated. Although much security research into WSNs has been accomplished, researchers struggle to conduct a thorough analysis of closed-source proprietary protocols. Of the numerous available and under-analyzed proprietary protocols, those based on ITU-T G.9959 recommendation specifying narrow-band sub-GHz communications have significant growth potential. The Z-Wave protocol is the most common implementation of this recommendation. Z-Wave developers are required to sign nondisclosure and confidentiality agreements, limiting the availability of tools to perform open source research.

As the Department of Defense begins its integration of WSNs into their secure infrastructure, it is crucial to fully understand the security implications involved while using available systems and protocols. In this way, accurate risk and vulnerability assessments can be accomplished with information gathered during protocol security evaluations. Furthermore, with knowledge gained, security researchers can develop tools and methods to reduce the attack surface thereby allowing for safer WSN integration into their network infrastructure.

Motivated by the need for security evaluations, this work discovers new vulnerabilities in the Z-Wave protocol and supported devices. These vulnerabilities allow an attacker to inject rogue devices into the network and perform a type of covert channel attack by hiding information in Z-Wave packets. Given existing vulnerabilities and exploitations, defensive countermeasures are needed. This work thus engineers a Misuse-Based Intrusion Detection System (MBIDS) capable of monitoring Z-Wave

networks and designs three experiments to test the detection accuracy of the system against attacks. Results from the experiments demonstrate the MBIDS can accurately detect intrusions in a Z-Wave network achieving a mean misuse detection rate of 99%.

Overall, this research contributes new vulnerabilities and exploitations in Z-Wave networks and an MBIDS capable of detecting rogue devices and manipulated packet injection attacks, enabling more secure Z-Wave networks.

This work is dedicated to the ***The Most High God*** *whom I serve.*

So, whatever you eat or drink, or whatever you do, do all to the glory of God.

1 Corinthians 10:31

For while we were still weak, at the right time Christ died for the ungodly. For one will scarcely die for a righteous person–though perhaps for a good person one would dare even to die–but God shows his love for us in that while we were still sinners, Christ died for us.

Romans 5:6-8

Acknowledgements

There are many people I would like to thank, for without them, this thesis would not be completed:

- Major Benjamin Ramsey for being my advisor and for his patience and direction while guiding me through this thesis process.
- Lieutenant Colonel Mason Rice for agreeing to serve on my committee and providing me with invaluable advice not only for this thesis but also my career as a US Army officer.
- Major John Pecarina for agreeing to serve on my committee and providing many new ideas towards improvement and completion of this thesis.
- Christopher Badenhop and Joseph Hall for taking time out of your own research aiding me in establishing an underlying Z-Wave-capable software-defined radio for this work.
- Dr. Timothy Lacey and the CCR IT Staff who made every effort to provide Lissard Lab workspace and any equipment needed.
- Dr. Barry Mullins for introducing me to the world of computer and network vulnerability analysis and exploitation.

But most of all, I would like to thank my precious wife, my best friend. This work holds no value when compared to you. I love you deeply and am grateful for your respect, support, understanding, and love. I look forward to spending the rest of my life with you.

Jonathan D. Fuller

Table of Contents

Page

Abstract iv

Dedication vi

Acknowledgements vii

List of Figures xii

List of Tables xiv

List of Abbreviations xv

I. Introduction 1

1.1 Background and Motivation 1
1.2 Problem Statement and Research Goals 3
1.2.1 Problem Statement 3
1.2.2 Research Goals 3
1.3 Approach 4
1.4 Assumptions and Limitations 4
1.5 Thesis Overview 5

II. Background and Literature Review 6

2.1 Introduction 6
2.2 The Z-Wave Protocol 6
2.2.1 PHY Layer 7
2.2.2 MAC Layer 7
2.2.3 Adaptive Layer 8
2.2.4 LLC Layer and Application Layer 9
2.3 Z-Wave Attack Classes 9
2.3.1 Reconnaissance 10
2.3.2 Denial of Service 11
2.3.3 Packet Injection 12
2.4 Home Automation Networks 13
2.5 Vulnerability Analysis and Exploitation in Z-Wave Networks 15
2.5.1 Packet Capture and Injection in Z-Wave Networks 15
2.5.2 Gateway Attacks in Z-Wave Networks 18
2.6 Intrusion Detection Systems 20
2.6.1 Network IDS 20
2.6.2 Wireless IDS 20
2.6.3 Host IDS 21

Page

2.6.4 Signature-Based Detection . . . 23
2.6.5 Anomaly-Based Detection . . . 23
2.6.6 Stateful Protocol Analysis . . . 24
2.6.7 Misuse-Based Detection . . . 25
2.7 Summary . . . 26

III. Persistent Backdoor Attacks in Z-Wave Gateways . . . 27

3.1 Introduction . . . 27
3.2 Rogue Controller Injection . . . 27
3.2.1 Reconnaissance . . . 28
3.2.2 Gateway Vulnerabilities . . . 31
3.2.3 Adding a Rogue Controller to the Z-Wave Network . . . 33
3.3 Hel Attack . . . 37
3.3.1 Information Hiding in the Z-Wave MAC Frame . . . 37
3.3.2 Space Available to Hide Information . . . 38
3.3.3 Hel Attack . . . 39
3.4 Summary . . . 42

IV. Methodology . . . 43

4.1 Introduction . . . 43
4.2 Problem Definition . . . 43
4.2.1 Goals and Hypotheses . . . 43
4.2.2 Approach . . . 44
4.3 System Boundaries . . . 51
4.4 System Services and Outcomes . . . 52
4.5 Parameters and Factors . . . 53
4.5.1 Workload Parameters . . . 53
4.5.2 System Parameters . . . 55
4.6 Performance Metrics . . . 55
4.6.1 Packet Capture Device - AFIT Sniffer . . . 56
4.7 Evaluation Technique and Environment . . . 56
4.7.1 Evaluation Techniques . . . 56
4.7.2 Experimental Environment . . . 57
4.8 Design of Experiments . . . 58
4.8.1 Experiment 1: Rogue Device Detection . . . 58
4.8.2 Experiment 2: Manipulated Packet Injection Detection . . . 61
4.8.3 Experiment 3: Increased Detection Rate Post Enhancement . . . 63
4.9 Summary . . . 66

Page

V. Results and Analysis . . . 68

5.1 Introduction . . . 68
5.2 Results and Analysis of Experiment 1 . . . 68
5.2.1 Test 1: Valid Source ID and Destination ID . . . 68
5.2.2 Test 2: Valid Source ID and invalid Destination ID . . . 69
5.2.3 Test 3: Invalid Source ID and valid Destination ID . . . 69
5.3 Results and Analysis of Experiment 2 . . . 71
5.3.1 Test 1: All Valid Byte Fields . . . 71
5.3.2 Test 2: Invalid Packet Length . . . 71
5.3.3 Test 3 and 4: Invalid Source ID or Invalid Destination ID . . . 72
5.3.4 Test 5 and 6: Invalid Command Class or Invalid Command . . . 73
5.3.5 Test 7: Invalid Payload . . . 74
5.4 Results and Analysis of Experiment 3 . . . 76
5.4.1 Test 1: Re-Evaluation of Experiment 1 . . . 76
5.4.2 Test 2: Re-Evaluation of Experiment 2 . . . 76
5.5 Defense-in-depth Strategies . . . 77
5.5.1 Hide WLAN SSID . . . 77
5.5.2 WLAN security with a robust password . . . 78
5.5.3 MAC address filtering . . . 79
5.5.4 Enable UI authentication on the Z-Wave gateway . . . 80
5.5.5 Use a Reverse Proxy Server . . . 80
5.5.6 Disable unused network services . . . 81
5.5.7 Enable end-device encryption . . . 81
5.5.8 Inspect log files . . . 81
5.6 Summary . . . 82

VI. Conclusion . . . 83

6.1 Introduction . . . 83
6.2 Conclusions of Research . . . 83
6.2.1 Goal 1: Misuse Case Identification . . . 83
6.2.2 Goal 2: Rogue Device Detection . . . 83
6.2.3 Goal 3: Manipulated Injected Packet Detection . . . 84
6.2.4 Goal 4: Enhanced MBIDS . . . 84
6.3 Significance of Research . . . 85
6.3.1 Z-Wave Network Attacks . . . 85
6.3.2 MBIDS Employed . . . 86
6.4 Recommendations for Future Research . . . 88
6.5 Supporting Documentation . . . 90

Page

Appendix A. Z-Wave Command Classes 91

Appendix B. DoD Integration of WSNs 100

B.1 Introduction 100
B.2 Scenario 1 101
B.3 Scenario 2 101
B.4 Conclusion 102

Bibliography 103

List of Figures

Figure		Page
1.	Z-Wave protocol reference model	7
2.	Z-Wave frame structure for APP, LLC, MAC, and PHY layers.	8
3.	Home automation model	14
4.	Gateway access model	15
5.	Logical topology of a NIDS	21
6.	Logical topology of a WIDS	22
7.	Logical topology of a HIDS	22
8.	Rogue controller injection testbed	28
9.	Z-Wave gateway controllers	29
10.	Z-Wave devices	31
11.	Zenmap scan to find the Z-Wave network	32
12.	Rogue controller injection diagram	36
13.	Z-Wave MAC frame structure	38
14.	Injected Z-Wave packet	40
15.	Z-Wave gateway logfile	41
16.	Covert Z-Wave channel attack	42
17.	Packet data flow through the MBIDS	46
18.	Experiments and tests used to achieve the research goals	50
19.	System under test	52
20.	Example payload of injected packet	55
21.	Packet capture accuracy	57
22.	Block diagram of the experiment setup	58

Figure Page

23. Actual experiment setup 59
24. Misuse-Based Intrusion Detection 64
25. Routed Frame pre and post enhancement classification. 65
26. Z-Wave mesh topology routing 66
27. Experiment 1 - Test 1 packet structure 68
28. Experiment 1 - Test 2 packet structure 69
29. Experiment 1 - Test 3 packet structure 70
30. Experiment 2 - Test 1 packet structure 71
31. Experiment 2 - Test 2 packet structure 72
32. Experiment 2 - Test 3 and 4 packet structure 73
33. Experiment 2 - Test 5 and 6 packet structure 73
34. Experiment 2 - Test 7 packet structure 74
35. Z-Wave Misuse-Based Intrusion Detection System 87

List of Tables

Table		Page
1.	Jamming efficiency against Z-Wave frames	12
2.	Hardware and software components	28
3.	Z-Wave gateway default configurations	29
4.	Z-Wave gateway fingerprints	30
5.	Experiment components	60
6.	Results of Experiment 1	70
7.	Results of Experiment 2	75
8.	Results of Experiment 3	78
9.	Brute force password cracking	80
10	Z-Wave Command Classes	91

List of Abbreviations

Abbreviation		Page
WSNs	Wireless Sensor Networks	1
ITU-T	International Telecommunications Union - Telecommunication Standardization Sector	1
NDAs	Nondisclosure Agreements	2
DoD	Department of Defense	2
DHS	Department of Homeland Security	2
RVAs	Risk and Vulnerability Assessments	2
RF	Radio Frequency	2
MBIDS	Misuse-Based Intrusion Detection System	3
WLAN	Wireless Local Area Network	3
SDR	Software Defined Radio	4
HANs	Home Automation Networks	5
IDS	Intrusion Detection System	5
PHY	Physical	6
MAC	Media Access Control	6
SAR	Segmentation and Reassembly	6
LLC	Logical Link Control	6
ID	Identification	6
NWK	Network	7
ENC	Encryption	7
PSDU	PHY Service Data Unit	7
EoF	End of Frame delimiter	7
MSDU	MAC Service Data Unit	8

Abbreviation		Page
SDK	Software Development Kit	9
UI	User Interface	16
USRP	Universal Software Radio Peripheral	16
SSH	Secure Shell	18
CFLs	Compact Fluorescent Lamps	19
NIDS	Network IDS	20
WIDS	Wireless IDS	20
HIDS	Host IDS	21
URL	Uniform Resource Locator	30
SUT	System Under Test	51
CUT	Component Under Test	52
LAND	Local Area Network Denial	64

A MISUSE-BASED INTRUSION DETECTION SYSTEM
FOR ITU-T G.9959 WIRELESS NETWORKS

I. Introduction

1.1 Background and Motivation

Wireless Sensor Networks (WSNs) are technologies where computing, communications, and control are tightly coupled [FR15a]. WSNs consist of multiple sensor nodes that collect information, respond to commands, and report updates to other nodes in the network [HGSW11] and are used in military surveillance, health care, environmental science, smart metering, and home automation [SWH11, RM12, RM13]. The use of WSNs is increasing because they extend communications ranges at low-cost, low-power, and low-complexity [PTBR14].

A security analysis of WSNs is accomplished discussing the security implications of various protocols when used in critical infrastructure analyzing attack classes including reconnaissance, packet injection, denial-of-service, and man-in-the-middle [RM12]. Reaves and Morris [RM12] analyze IEEE 802.11-based protocols, IEEE 802.15.4-based protocols, and proprietary protocols. However, since proprietary protocols are closed-source, there are few security research publications regarding their use. Therefore, researchers have struggled to conduct a thorough analysis.

Of the numerous wireless protocols, those based on the International Telecommunications Union - Telecommunication Standardization Sector (ITU-T) G.9959 recommendation have significant growth potential in WSNs. The ITU-T G.9959 specifies short range narrow-band sub-GHz communications [ITU15]. The most common

implementation of the ITU-T G.9959 recommendation is commercially known as Z-Wave, marketed by Sigma Designs. However, security evaluations of the Z-Wave protocol are difficult because developers are required to sign Nondisclosure Agreements (NDAs), limiting the availability of tools to perform open source research. New and emerging network protocols are often initially believed secure, but are not vetted until tools exist for security research [GBM+12].

Furthermore, the increase in the use of WSNs furthers their integration and application in many industries and organizations, including the Department of Defense (DoD). The Department of Homeland Security (DHS) seeks to develop and test the effectiveness of WSNs for law enforcement and first responder applications [DHS10]. Smart metering initiatives are also currently directed for all military bases. While ITU-T G.9959-based wireless systems, including Z-Wave, are currently not prevalent in DoD infrastructures, their growth potential motivates the need for in-depth security evaluations of its use before its integration. When integration occurs, the vulnerability analysis and exploitation presented herein allows DoD system integration engineers the ability to conduct valid Risk and Vulnerability Assessments (RVAs) and develop tools and methods to reduce the attack surface before full system integration.

Recent works discuss the evaluation of the Z-Wave protocol including vulnerability analysis and exploitation. These works are categorized as either a gateway approach or a Radio Frequency (RF) approach.

The gateway approach mainly investigates the security implications of using Z-Wave gateway controllers from varying vendors. Many gateway controllers are not engineered with security in mind. Researchers have therefore discovered and analyzed vulnerabilities exploiting them gaining full control of the device. As the *gateway* into the Z-Wave network, access to the gateway controller allows for full control of Z-Wave devices.

The RF approach focuses on packet capture and replay or injection. Z-Wave RF traffic is captured using open source hardware and software tools. Captured packets are dissected for evaluation whereby the attacker can now craft their own packets and inject them into the network maliciously modifying network operations.

1.2 Problem Statement and Research Goals

1.2.1 Problem Statement.

Vulnerabilities in the Z-Wave protocol and network implementations have been discovered and are discussed in Section 2.5. However, any means of proposing detection or prevention of attacks in Z-Wave networks is lacking. Although the discovery of vulnerabilities and exploitations are necessary to conduct RVAs, proper security countermeasures are needed in order to safely employ Z-Wave networks.

1.2.2 Research Goals.

It is hypothesized that by conducting deep Z-Wave packet inspection, known good packets and malicious packets are distinguishable. Furthermore, their differences can be used to create a monitoring tool for Z-Wave networks.

This thesis focuses on designing and engineering a Misuse-Based Intrusion Detection System (MBIDS) that allows an investigator or system administrator to monitor Z-Wave traffic in real-time for any transmissions intended to disrupt normal network operations. The MBIDS is designed to operate on the Wireless Local Area Network (WLAN) that the Z-Wave gateway controller resides on. As packets are transmitted within the Z-Wave network, the MBIDS captures and evaluates them for validity. If the packet is not valid, is from a rogue device, or is manipulated, the MBIDS logs the packet as a misuse case and notifies the investigator or system administrator of malicious activity.

There are four primary goals for this research.

- Discover misuse cases by capturing, dissecting, and evaluating Z-Wave packets.
- Identify the misuse detection rate of the MBIDS against rogue device attacks.
- Evaluate the effectiveness of the MBIDS against manipulated injected packet attacks.
- Implement enhancement strategies and evaluated the hypothesized increase misuse detection rate.

1.3 Approach

A Python-based monitoring tool is developed that receives captured packets from a Z-Wave-capable Software Defined Radio (SDR). The tool uses signatures and states that are derived from an in depth study of Z-Wave packet transmissions. This study compares the ITU-T G.9959 recommendation with the packet byte fields in order to establish a framework for further evaluation. This framework allows for the development of signatures and protocol states for packet comparison. A real-world Z-Wave network is then engineered with a gateway controller and multiple devices on a WLAN backbone to test the effectiveness of the MBIDS against two new attacks presented herein evaluating the efficacy of this approach.

1.4 Assumptions and Limitations

This research provides a proof of concept system for detecting rogue devices and maliciously injected packets in Z-Wave networks. The Z-Wave network is controlled by a Raspberry Pi 2 Model B with General Purpose Input/Output RaZberry Pi Daughter Card version 2.0.0 (henceforth referred to as RaZberry Pi). Although the exploitations presented herein and the system developed are applicable to other Z-Wave

gateways, scripts are written specifically to work in conjunction with the RaZberry Pi because of its low-cost and available documentation [RaZ15], allowing for a clear understanding of its software framework.

1.5 Thesis Overview

Chapter II presents relevant technical details of the Z-Wave protocol, an overview of Home Automation Networks (HANs), and background information of Intrusion Detection System (IDS). Also presented are related works in the area of Z-Wave research, including vulnerability analysis and exploitation of the protocol and gateway controllers. Chapter III motivates the need for countermeasures against two new Z-Wave attacks and others within their taxonomy, namely rogue device attacks and manipulated packet injection attacks. Chapter IV outlines the methodology used to design, set up, and conduct the experiments to test the effectiveness of the MBIDS. Chapter V provides a discussion and analysis of the experiment results. The conclusions drawn from the experiment results, the significance of the MBIDS, and areas for future research are given in Chapter VI.

II. Background and Literature Review

2.1 Introduction

This chapter presents Z-Wave background information and related research including protocol specifics and a discussion of vulnerabilities and exploitations of Z-Wave networks. Section 2.2 provides relevant technical details of the Z-Wave protocol and Section 2.3 discusses various attack classes in Z-Wave networks. An overview of HANs is provided in Section 2.4 providing a clearer understanding of the design of experiments in Section 4.8. Section 2.5 discusses vulnerabilities and exploitations in Z-Wave networks. Section 2.6 discusses basics of an IDS providing necessary information supporting the choice of an MBIDS over other types of IDSs. This chapter then concludes and is summarized in Section 2.7.

2.2 The Z-Wave Protocol

All Z-Wave products adhere to the ITU-T G.9959 Physical (PHY), Media Access Control (MAC), Segmentation and Reassembly (SAR), and Logical Link Control (LLC) layer specification. This ensures interoperability between vendor devices, but differ at the Application Layer (Figure 1). ITU-T G.9959-based networks operate in the industrial, scientific, and medical bands: 908.4 MHz in North America and additional frequencies in other regions. Control nodes send commands and slave nodes respond to commands in Z-Wave networks. As a meshed topology network, slave nodes also forward commands to other nodes not directly reachable by the control node. Message forwarding has a hop limit of four nodes and a maximum of 232 nodes are allowed. For network overlap, each Z-Wave network has a unique 4B Home Identification (ID) specified by the controller. The Z-Wave protocol consists of

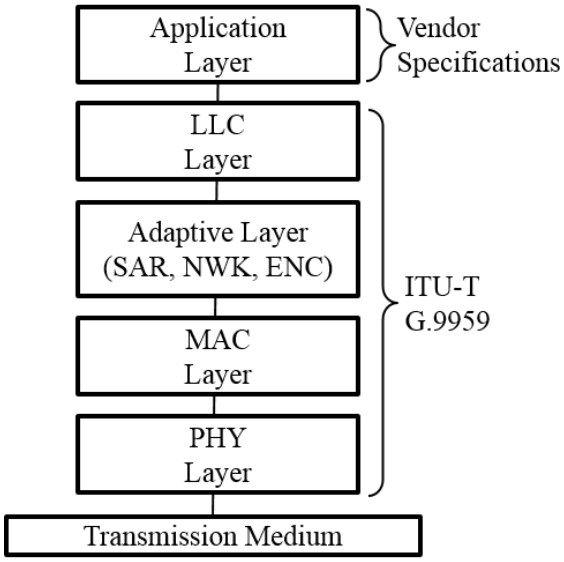


Figure 1. Z-Wave protocol reference model

five layers where the Adaptive Layer is a set of three layers: SAR, Network (NWK), and Encryption (ENC) as illustrated in Figure 1.

2.2.1 PHY Layer.

The PHY layer uses carrier sense multiple access with collision avoidance to moderate access to the wireless medium. The protocol supports three data rates: 9.6 kbps (R1), 40 kbps (R2), and 100 kbps (R3). Figure 2 illustrates the PHY Protocol Data Unit consisting of the PHY Header, PHY Service Data Unit (PSDU) and the End of Frame delimiter (EoF) . The maximum PSDU size is 170B while operating at R3. However, if the transceiver is operating at R1 or R2, the maximum PSDU size is 64B.

2.2.2 MAC Layer.

The MAC layer is responsible for the transfer of data between nodes in the network. It is also responsible for frame acknowledgment, data validation, and retransmission. The three types of MAC frames are singlecast, multicast, and acknowledgment. The MAC frame structure is shown in Figure 2. When a node transmits a

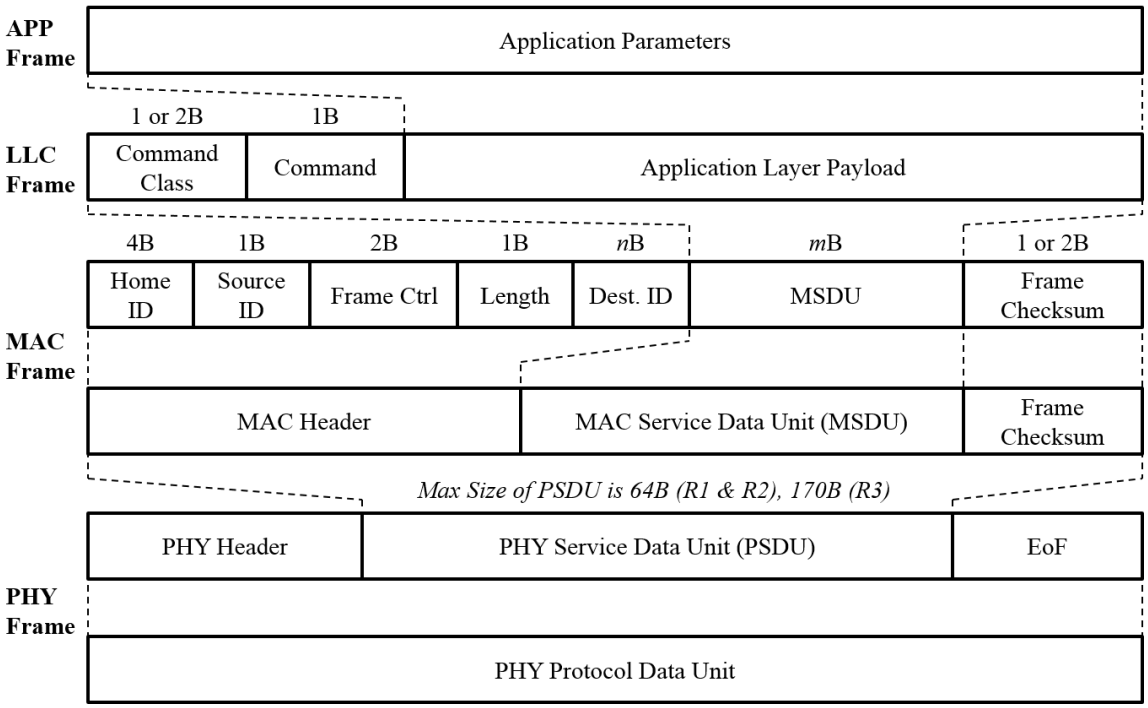


Figure 2. Z-Wave frame structure for APP, LLC, MAC, and PHY layers.

singlecast frame, there is one destination address. Upon receipt of a singlecast frame, a device responds with an acknowledgment frame. Acknowledgment frames and singlecast frames are structured similarly with the exception of a zero-byte MAC Service Data Unit (MSDU) in the acknowledgment frame. Conversely, multicast frames are sent to multiple destination nodes without acknowledgments.

2.2.3 Adaptive Layer.

The Adaptive Layer uses the sublayers to handle mesh network routing at the NWK layer, segmentation and reassembly of datagrams at the SAR layer, and encryption at the ENC layer. The sublayer is selected based on values set in the Frame Control field of the MAC Header or the Command Class field of the LLC Layer (Figure 2). If specific values in the Frame Control field are not set, the packet is forwarded to the following layer without action. The remaining specifics of the Adaptive Layers are out of scope of this research and are not discussed any further.

Of note, the Z-Wave protocol supports the Advanced Encryption Standard with 128-bit keys. Although supported, encryption is not required. In the low-rate, low-power WSNs, memory and power are limited, discouraging vendors from adding features unless necessary [Bee08]. Surveys of similar protocols conclude the use of encryption is not universal [RMSB13].

2.2.4 LLC Layer and Application Layer.

The LLC layer consists of the Command Class, Command, and Application Layer Payload (Application Parameters) (Figure 2). Command Classes are broken into two classes: the device class and Command Class.

A Command Class is related to a certain function or device. An example is the *Binary Switch* Command Class. The Binary Switch uses three Commands: `SET` to turn a device on or off, `GET` to request a device status, and `REPORT` to respond to the request. These three Commands are foundational to all Z-Wave devices.

The device class is subdivided into the basic, generic and special device classes. The basic device class distinguishes between controller, slave or routing-slave device types. The generic device class defines what function the device performs as a controller or slave. Lastly, the special device class allows for more specificity in the device functionality.

The Application Layer consists of the Application Parameters and is implementation specific depending on the Z-Wave developer. The parameters are used by the device in execution of the Command to perform a specific function.

2.3 Z-Wave Attack Classes

While Z-Wave is a proprietary protocol stack, vendors are able to purchase a Software Development Kit (SDK) to produce Z-Wave certified products. Given the nature

of hardware and software development, there are some vulnerabilities introduced by developer-specific implementation faults, as illustrated in [FG13]. This section focuses on potential vulnerabilities of the underlying ITU-T G.9959 recommendation common to all devices and the attack classes that exploit those vulnerabilities. The three classes of Z-Wave attacks discussed in this section are: reconnaissance, denial of service, and packet injection. These three attacks undermine network confidentiality, availability, and integrity, respectively. Reconnaissance attacks involve an attacker passively collecting traffic or actively probing a target network to gain information without interfering with normal network operations. Denial of service attacks prevent wireless system access and cause varying degrees of system unavailability. Packet injection attacks involve the transmission of specially-crafted packets with the intent of manipulating network or device behavior.

2.3.1 Reconnaissance.

A reconnaissance attack occurs when an attacker gains information about a network. The information received can include protocols in use, device types, traffic flow patterns, and even encryption keys if not handled properly. Information received from reconnaissance may prove useful in providing the attacker with an accurate mapping of the system, services, or vulnerabilities enabling more significant attacks in the future. Using a high-gain antenna, observations can be achieved at long distances, allowing the attacker to remain inconspicuous while gathering information.

Z-Wave network information gathered useful to the attacker are: (i) the use of encryption during transmission and (ii) the frame header content which includes the unique Home ID of the Z-Wave network, Source ID of the device being sniffed and Destination ID. Reconnaissance lays the foundation for more sophisticated follow-on attacks.

Demonstrations of Z-Wave exploitation have been published to date all of which rely heavily on the ability to conduct reconnaissance to gain requisite knowledge for crafting follow-on attacks; they are discussed in further detail in Section 2.5.

2.3.2 Denial of Service.

Denial of service attacks prevent access to contentious resources, such as the RF spectrum, with the effect of degrading legitimate system access. Studies have demonstrated such attacks are easily accomplished using off-the-shelf equipment [PIK11]. Denial of service attacks on wireless networks can deny or degrade access to resources from the PHY to Application layer. For example, the PHY layer may be susceptible to narrow-band jamming, the carrier sense algorithms may be exploited to deny access to the medium [XTZW05], or routing nodes can be consumed with overflowing interface queues.

Constant and deceptive jamming are effective for conducting denial of service attacks due to MAC layer collision avoidance characteristics of Z-Wave. While an attacker is operating a constant or deceptive jammer, any node within range will sense the channel as busy and wait to transmit. Even more effective and efficient is reactive jamming [LKP10] because it is difficult to detect [SDv10]. A reactive jammer only transmitting after a preamble and start of frame delimeter of a Z-Wave PHY frame is detected need only corrupt one bit of the PSDU in order to cause an integrity check error and complete loss of the frame. The non-correcting integrity checks used for Z-Wave are capable of detecting, but not correcting, single bit errors. Even worse, the corruption of a single bit in the Z-Wave PHY layer, unlike PHY layers that use spreading techniques such as direct sequence spread spectrum, is achievable using narrow-band jamming. Use of error correction codes, as in IEEE 802.11a, is more robust to bitwise jamming [Nou12]. Table 1 presents a preliminary estimate of

Table 1. Jamming efficiency against Z-Wave frames [BFH+15].

Data Rate	Integrity Check*	Max Payload	Bits to Jam	Jammer Efficiency
9.6 kbps (R1)	8-bit checksum	512 bits	1	512
40 kbps (R2)	8-bit checksum	512 bits	1	512
100 kbps (R3)	16-bit checksum	1360 bits	1	1360

*Z-Wave uses non-correcting integrity checks.

jammer efficiency in bits jammed per bit transmitted against Z-Wave (i.e., ratio of communication effort to jammer effort), based on results in [Nou12].

Depending on the objective, an attacker may use any of the methods described to deny the availability of one or more nodes in the wireless network. For example, a Z-Wave network containing a thermostat (sensor) and water valve (actuator) could be subject to a denial of service attack that prevents the thermostat from reporting the current temperature or obstructs commands . A similar scenario was successfully demonstrated in [RM12], with a gas pipeline Remote Terminal Unit including a wireless pressure sensor, pump, and relief valve.

Denial of service attacks against Z-Wave networks may also result from flooding the network with spoofed packets. This method is discussed further in the following section.

2.3.3 Packet Injection.

Due to the broadcast nature of wireless networks, an attacker armed with information gained from reconnaissance may be able to inject forged packets into the network. The ability to conduct packet injection enables an attacker to masquerade as a legitimate network device while transmitting messages to manipulate overall system operation. Using publicly available hardware and software, researchers have recently

reported the ability to conduct packet injection attacks to manipulate Z-Wave devices. An attacker may be able to (i) flood the network with traffic causing a denial of service at the routing or Application layers, (ii) send false status messages, and (iii) provide the control node with false routing information to poison the network routing table. If the target Z-Wave network has been implemented in a HAN security system, the ability to inject commands and report false state information prove disastrous. For example, an attacker chooses to send an OFF command to an alarm in a security system immediately following every ON command sent by the door sensor. As a result, those occupying the residence are unaware the alarm should be activated.

2.4 Home Automation Networks

A HAN consists of wireless sensors that exchange control and information messages [LK12]. Device types include door locks, security sensors, alarms, environmental controls, light modules, and motion sensors (Figure 3). The user either manages the HAN with a Z-Wave controller from inside the home (local access) or a hybrid controller providing local and global access (Figure 4).

Managing the Z-Wave HAN locally lessens the features and span of network control. Being able to only manage the network from within the home does not provide the user with the ability to control devices while beyond the controller RF transmission range. However, managing the network through a globally connected gateway provides the user with the ability to control their network with a mobile device or other Internet connected computer from any geographic location. For example, if a user forgets to lock their door or arm the alarm system when they leave the house, they can login to the gateway and configure devices. With this accessibility comes added vulnerabilities. Not only does the user have access to their Z-Wave HAN, but anyone that can compromise their WLAN defense can also garner access [FR15b].

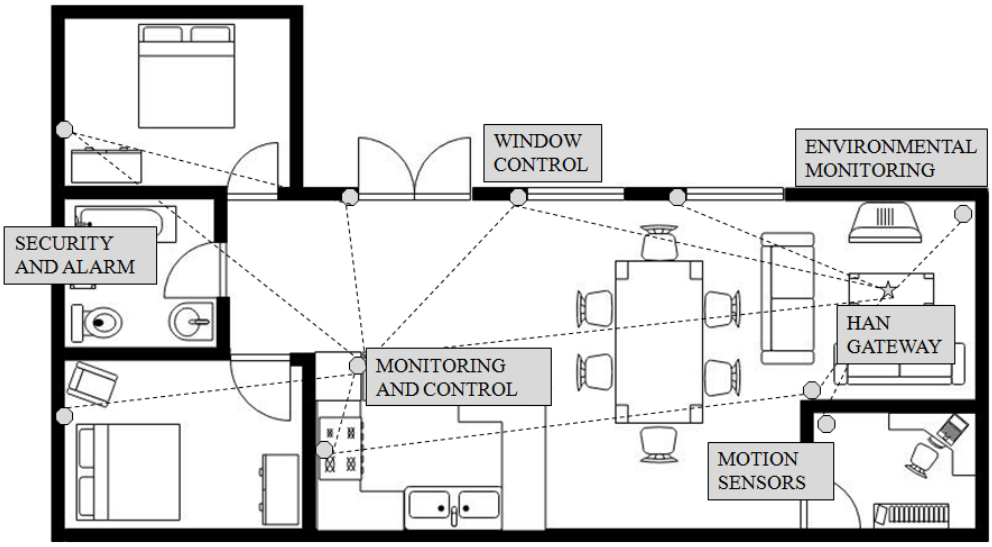


Figure 3. Home automation network model: multiple Z-Wave devices communicating with each other and controlled by the Z-Wave gateway [FR15a].

As reported in [Smi14], 65% of households with Internet access have a WLAN. Statistics show that between 11% to 30% of households have unsecured WLANs [Wig15]. A survey of IEEE 802.15.4 Wireless Personal Area Networks in ten US cities found similar percentages of unsecured networks [RMSB13]. Even if the WLANs are secured, there are weaknesses and proven exploits against all WLAN security protocols [Lan05, FNS12, Ahm10].

WLAN access is not the only means to gain access to a gateway device. An attacker can gain physical access to the gateway by intercepting it en route to the retailer or end user [BW15] or social engineering attacks, prompting an attacker to install malware on their device [GSC+14].

Although physical access and social engineering attacks are both viable means to gain access to a gateway, they both provide the target with more opportunities to discover the attack. WLAN compromise allows an attacker to remain hidden while performing attacks solely through the RF spectrum.

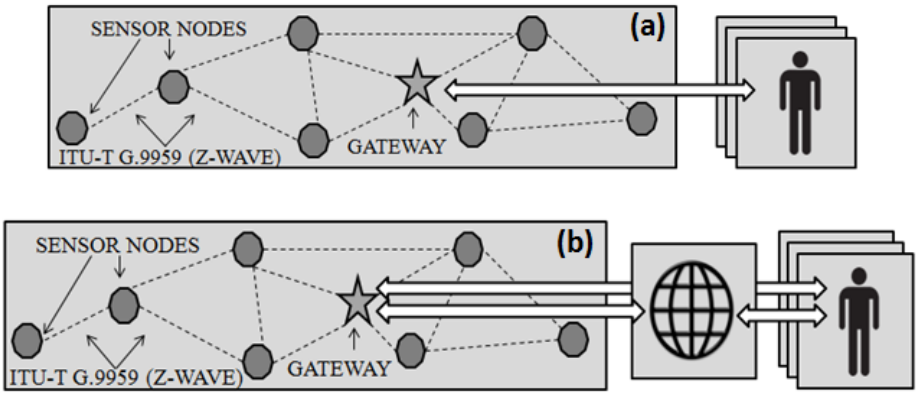


Figure 4. (a) Local gateway access: users manage their Z-Wave HAN via their gateway locally. All device control is performed from inside the home. (b) Global gateway access: users manage their Z-Wave HAN via their gateway locally or globally. Z-Wave devices can be controlled using any Internet connected device [FR15a].

2.5 Vulnerability Analysis and Exploitation in Z-Wave Networks

Previous works show the possibility of exploiting vulnerabilities in Z-Wave networks. These works are categorized as a RF approach or gateway approach exploiting PHY layer transmissions and device vulnerabilities, respectively. The former includes packet capture and injection attacks whereas the latter includes exploitation of Z-Wave gateways.

2.5.1 Packet Capture and Injection in Z-Wave Networks.

Packet injection attacks enable an attacker to masquerade as a legitimate user. Due to the broadcast nature of wireless networks, an attacker armed with tools capable of capturing wireless information can inject forged packets into the network. Since Z-Wave developers are required to sign NDAs, researchers use publicly available tools to conduct packet capture and injection attacks in Z-Wave networks.

2.5.1.1 Z-Force.

Fouladi and Ghanoun [FG13] provide the first public vulnerability research of the Z-Wave protocol. They develop a sniffer project known as *Z-Force*. The sniffer project consists of two Texas Instruments CC1110 boards (one transmits and one receives), custom firmware, and a Windows-based User Interface (UI). Before implementing the sniffer, Fouladi and Ghanoun conduct an in-depth study of the Z-Wave protocol and uncover the details of frame encryption and authentication algorithms. They discover an implementation error used in a Z-Wave compliant door lock that allows them to reset the established network key.

The Z-Force tool allows them to capture the Home ID of the controller and Node ID of the device, reset the network key to a different value, and inject unauthorized commands into the Z-Wave HAN. As a result, they are able to open and close the lock by bypassing the controller. The Z-Force tool has only been demonstrated to work on the 860.4 MHz European frequency and its closed-source nature inhibits successful implementation and replication.

2.5.1.2 Scapy-Radio.

Another analysis of Z-Wave is the *Scapy-Radio* project [PLD14]. Scapy-Radio combines Scapy and gnuRadio software on the Ettus Universal Software Radio Peripheral (USRP) B210 to capture traffic and replay packets back into the Z-Wave HAN. To test their device, Picod et al. [PLD14] implement a Z-Wave HAN using a Raspberry Pi and an Aeon Labs Z-Stick programmed using the open source *Open-ZWave* software. They include two Z-Wave devices into the network: an alarm device and a motion sensor. Capturing the network traffic, Picod et al. [PLD14] record and analyze packets, noticing that when the motion sensor is activated, it sends a packet to the controller.

Upon receipt, the controller sends a packet to the alarm device with a command to activate. After multiple improvements, Scapy-Radio is now able to detect ON commands from the controller to the alarm device and subsequently inject OFF commands to the alarm effectively denying service to the device.

2.5.1.3 AFIT Sniffer.

The AFIT Sniffer [BFH+15] is an extension of Scapy-Radio [PLD14]. Badenhop et al. [BFH+15] develop a Z-Wave packet sniffer using a USRP N210, gnuRadio, Scapy-Radio, and a Z-Wave Wireshark dissector. Using this tool, they passively discriminate between Z-Wave devices by functionality and vendor. To do this, Badenhop et al. observe acknowledgment response times to identify implementation differences in device firmware. Their results show that using this technique, devices are distinguishable if their manufacturers are different suggesting intra-vendor similarities. The AFIT Sniffer is used in the research herein and is further discussed in Section 4.7.

2.5.1.4 EZ-Wave.

The authors of [HRRL16] extend work by [PLD14] providing broad support for the Z-Wave protocol. Specifically, Hall et al. [HRRL16] demonstrate transceiver fingerprinting through preamble manipulation using two low-cost HackRF SDRs. The result is an open source tool *EZ-Wave*: a set of Z-Wave network reconnaissance tools capable of network discovery and enumeration, device fingerprinting, and device status information gathering. The ability to accurately fingerprint Z-Wave devices using EZ-Wave allows an attacker to discover door locks that have the encryption flaw discovered in [FG13] and exploit them.

2.5.2 Gateway Attacks in Z-Wave Networks.

There are two types of control nodes in Z-Wave networks. Portable controllers are handheld devices that allow a user to control devices from within RF transmission range. As discussed in Section 2.4, gateway controllers provide the user the ability to manage their Z-Wave network locally or globally..

Given the global accessibility of the gateway and the chain of trust between the gateway and the WLAN backbone, if an attacker gains access to the WLAN, the attacker has access to the gateway. After gaining access, the inherent vulnerabilities in Z-Wave gateways allow an attacker to easily take control of the network [FR15b]. Vulnerabilities include:

- Lack of user validation (e.g., insecure web UI)
- Lack of gateway encryption (e.g., messages are sent unencrypted)
- Enabled and unused services (e.g., port 22 - Secure Shell (SSH))

The resulting implications of the inherent vulnerabilities are seen in the following works.

2.5.2.1 Insufficient Authentication Checks.

The authors of [CBS13] demonstrate the exploitation of several HAN gateways, including the VeraLite Smart Home Controller. Crowley et al. [CBS13] find several vulnerabilities that expose sensitive information from the VeraLite while allowing an attacker to fully control devices on the network. Finding insufficient authentication checks and lack of user validation, a universal plug and play functionality is exposed allowing an attacker to execute Lua code, a programming language designed for embedded clients, as `root` user. Crowley et al. [CBS13] exploit this vulnerability and successfully create a backdoor account on the device.

Another vulnerability found is that the VeraLite does not protect against Cross-Site Request Forgery. It is therefore possible for an attacker to update the device firmware with their own malicious firmware. It has been shown that remote modification of firmware can severely affect a device and expose confidential information [CV11].

2.5.2.2 Non-network Device Exploitation.

Oluwafemi et al. [OGPK13] investigate the feasibility of causing physical harm to HAN users through the explosion of Compact Fluorescent Lamps (CFLs). Four distinct electronic signals are transmitted to CFLs connected to a Z-Wave device until the CFLs emit a visual or auditory spark or fail completely. Although they conclude that harming individuals via their attack vector is difficult, they observe that non-networked devices, such as CFLs, might possibly be connected to networked devices and therefore can be compromised remotely given HAN insecurities.

2.5.2.3 Address Resolution Protocol Poisoning towards Gateway Exploitation.

Recently, Barcena and Wueest [BW15] discuss multiple insecurities in HANs. During their research, Barcena and Wueest [BW15] poison the gateway Address Resolution Protocol to redirect firmware update requests to their own malicious server. After modifying the firmware, the gateway receives the malicious firmware as a legitimate update giving the attacker full control. Barcena and Wueest [BW15] also directly access the controller application since the device does not require user authentication and has an insecure web UI.

These early studies illustrate the possible attacks resulting from wireless traffic capture and injection or compromise of the Z-Wave gateway. All Z-Wave related

research to date discover and exploit vulnerabilities in the protocol or devices. The obvious area of research that is lacking are proposals or methodologies to secure Z-Wave networks. Therefore, previous works, including newly discovered vulnerabilities, motivate the need for countermeasures toward safe Z-Wave employment.

2.6 Intrusion Detection Systems

An IDS monitors events occurring on a network or computer device analyzing them for possible malicious or unwanted activities [NIS07] that attempt to compromise resource confidentiality, integrity, and availability [Yus08]. Additionally, IDSs are used to identify problems with security policies, deterring individuals from circumventing security policies, and logging incidents for further review by the administrator or investigator. There are three main types of IDS deployments: Network, Wireless, and Host.

2.6.1 Network IDS.

A Network IDS (NIDS) analyzes all network traffic (Figure 5). Working in promiscuous mode, the NIDS captures network traffic and evaluates it checking for unusual activity that violates predetermined specifications. Once malicious activity is detected, the investigator or system administrator are notified or the violation is logged for future auditing and review. A NIDS is solely a passive system, only capturing, identifying, alerting, and/or logging any security incidents on the network.

2.6.2 Wireless IDS.

A Wireless IDS (WIDS) monitors the RF spectrum, capturing packets and analyzing them for any unwanted or malicious activity (Figure 6). Similar to the NIDS, the WIDS serves to detect violations and report or log incidents for an administrator

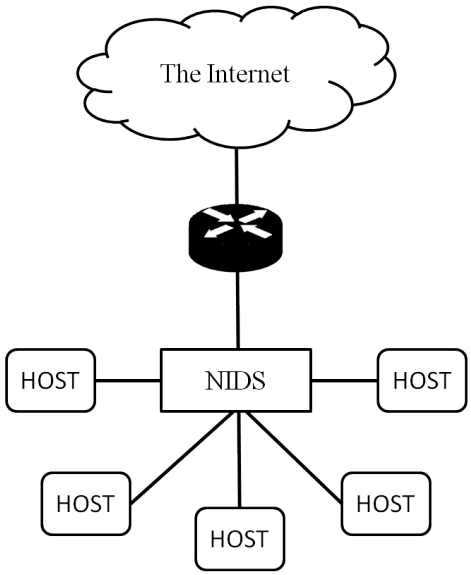


Figure 5. Logical topology of a NIDS

or investigator to review and act upon. Given that wireless networks are particularly vulnerable to spoofing and route poisoning attacks due to the congested transmission medium, previous research proposes wireless intrusion detection through preamble manipulation in IEEE 802.15.4 networks [RMTG15].

2.6.3 Host IDS.

The Host IDS (HIDS) identifies threats to a specific device. The HIDS can operate by taking a snapshot of a system and comparing it to a previous snapshot in order to identify if any critical system fields are modified or deleted [NIS07]. If so, an alert is sent to the administrator or investigator for review or is logged for future auditing. However, the HIDS can also be an agent installed on the individual system to monitor for any suspicious or malicious activity (Figure 7). As in all other IDS deployments, the HIDS is passive, only capturing, identifying, alerting, and/or logging any security incidents on the device. Examples of HIDS agents are Norton™ Antivirus and McAfee ® Antivirus software.

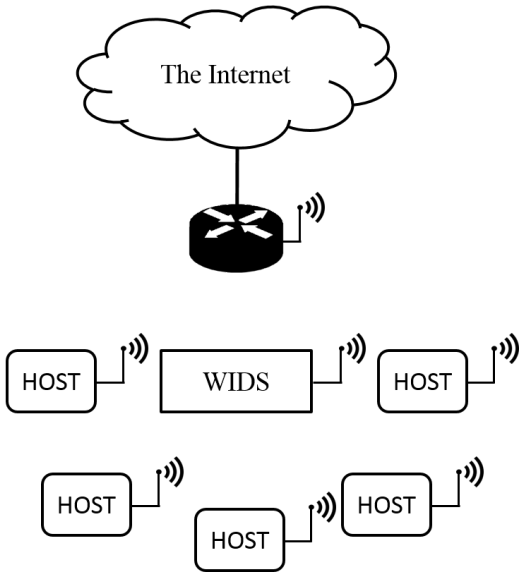


Figure 6. Logical topology of a WIDS

Whether the IDS is employed as a Network, Wireless, or Host, there are varying detection mechanisms that are used. Type of detection mechanisms include Signature-Based Detection, Anomaly-Based Detection, Stateful Protocol Analysis, and Misuse-Based Detection [NIS07]. The proceeding sections discuss each detection mechanisms including pros and cons for each.

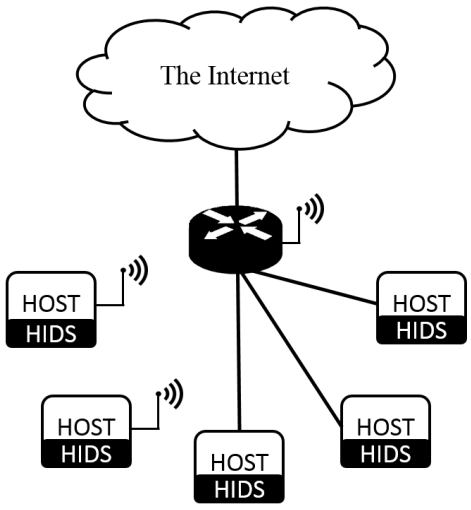


Figure 7. Logical topology of a HIDS

2.6.4 Signature-Based Detection.

Signature-Based detection mechanisms work by identifying signatures that correspond to known threats [NIS07]. Examples of signatures are:

- Any e-mail attachments with extension *.exe*, which can be a malicious executable
- Any wireless packet that contains a source ID that does not actually exist on the network

2.6.4.1 Advantages.

Signature-Based IDSs are dependent upon a set of signatures that must be established before the IDS is operational. Signature-Based IDSs are lightweight and if the database of signatures are thorough, the likelihood of improperly classifying a packet is low. This means that correct classification is high for malicious traffic.

2.6.4.2 Disadvantages.

Signature-Based IDSs cannot detect zero day attacks or new exploits, resulting in a relatively high false rejection rates. Packet analysis also provides a possible bottleneck to network traffic. In order to detect a packet matching the database of signatures, the IDS conducts exact string matching or approximate string matching. This can be time consuming depending on the quantity of signatures to evaluate captured packets against.

2.6.5 Anomaly-Based Detection.

Anomaly-Based detection compares definitions of normal activity with observed events to identify deviations [NIS07]. To determine normal activity, profiles are constructed that represent normal usage based on present system operation. The profiles

are then used to train the system with regard to legitimate behaviors. However, when a system is being trained, it is most vulnerable to experienced intruders aimed at training the system to point to intrusive behaviors that will seem normal.

2.6.5.1 Advantages.

The advantages of using Anomaly-Based detection is, unlike Signature-Based detection, there is a relatively high probability of detecting new types of attacks because they create instances that are out of sync with normal system operation. They are also able to detect the abuse of user privileges which are undetectable by a Signature-Based IDS.

2.6.5.2 Disadvantages.

Disadvantages of Anomaly-Based IDSs include their tendency to generate high improper classifications, specifically, classifying a malicious packet as normal. Because system usage is not generally monitored during the profile creation and training phase, normal user activity is not recorded. Therefore, false alarms are reported even though the behavior should be classified as normal.

2.6.6 Stateful Protocol Analysis.

Stateful protocol analysis works by comparing observed events with previously established profiles of generally accepted definitions of benign protocol activity for each protocol state [Kru04, NIS07]. Stateful means the IDS can understand and track the state of the protocols that have a notion of state. An example of stateful protocol analysis detection is monitoring a requests with its subsequent response. In this case, every request should have a certain response. Any response that does not conform to what is expected is classified as a violation.

2.6.6.1 Advantages.

The advantages of stateful protocol analysis is the ability to identify non-standard sequences of commands (e.g., issuing the same command over and over). Stateful protocol analysis is especially advantageous as an authenticator for suspicious activity given that the protocol requires authentication.

2.6.6.2 Disadvantages.

Disadvantages of stateful protocol analysis include high overhead because a state has to be recorded for later comparison. Stateful protocol analysis also cannot detect any attacks that do not violate generally accepted characteristics of protocol behavior. For proprietary protocols, details about the protocols are generally not available, making it difficult for to perform comprehensive, accurate analysis [NIS07].

2.6.7 Misuse-Based Detection.

There are varying definitions of Misuse-Based detection. Herein, Misuse-Based detection is defined as a process of comparing predetermined definitions or protocol states with observed transmissions. Any violation of the known-good status is classified as a misuse case. Signature-based detection is sometimes referred to as misuse detection [NIS07]. However, if they are synonymous, the resulting implications are misuse is only detected using signatures, which is not true. Misuse is detected using signatures and other methods like stateful protocol analysis. For this purpose of this work, an MBIDS includes signature-based detection and stateful protocol analysis.

2.7 Summary

This chapter presents relevant technical details of the Z-Wave protocol, HANs, and IDSs. Also presented are related research in vulnerability analysis and exploitation of Z-Wave networks.

An MBIDS is chosen over an Anomaly-Based IDS, solely Signature-Based IDS, and solely Stateful protocol analysis. Anomaly-Based systems tend to generate high improper classifications [NIS07]. Furthermore, the system is most vulnerable to packet injection attacks during the training phase recording malicious activity as normal. Although Signature-Based IDSs and Stateful protocol analysis can detect intrusions in Z-Wave networks, the combination of both (an MBIDS) provides the greatest possibility of monitoring Z-Wave networks. Based on the information provided herein, a wireless Misuse-Based detection mechanism is most suitable to identify attacks in Z-Wave networks.

III. Persistent Backdoor Attacks in Z-Wave Gateways

3.1 Introduction

Section 2.5 provides a discussion of previous works related to vulnerability analysis and exploitation in Z-Wave networks. These are categorized as gateway or RF approaches. This chapter presents two novel Z-Wave gateway attacks that create a persistent attack channel to the Z-Wave network with the second considered a hybrid (gateway/RF) approach. The first attack, discussed in Section 3.2, exploits Z-Wave gateway vulnerabilities that allow rogue device injection, specifically controllers. Section 3.3 presents a covert channel initiated Reverse SSH attack creating a open connection from the Z-Wave gateway to any Internet connected device the attacker chooses.

3.2 Rogue Controller Injection

There are three phases to this attack. First, is the initial reconnaissance of each device, where the default settings and modes of operations are identified. The second phase examines vulnerabilities in the device implementation that enables attackers to take control of the Z-Wave HAN. Phase three exploits a new vulnerability that allows an attacker to create a persistent attack channel by injecting a rogue controller into the Z-Wave HAN.

The equipment used to conduct vulnerability scanning and exploitation include those listed in Table 2, an Alfa AWUS036H Card, and an Aspire RF Wireless controller (Figure 8). Software tools used include *aircrack-ng* to defeat WLAN defense, *Zenmap* to scan the WLAN for Z-Wave gateways, *Burp Proxy* to capture and inject packets, and *Putty* to gain backend access via SSH.

Table 2. Hardware and software components

Classification	Type	Nomenclature
Hardware	Computer	HP EliteBook 8570w
	Processor	Intel Core i7-3720QM CPU
	RAM	16.0 GB
	System Type	64-bit
Software	Operating System	Ubuntu 14.04
	Software Development	LiClipse 2.2.0.2

Figure 8. Equipment used for vulnerability scanning and exploitation: (a) HP Elitebook 8570w, (b) Alfa AWUS036H Card, and (c) Aspire RF Wireless controller [FR15a].

3.2.1 Reconnaissance.

The three Z-Wave gateways under test are the VeraEdge Home Controller, RaZberry Pi, and the Almond+ (Figure 9). A real-world Z-Wave HAN is engineered using each gateway, each consisting of a smart switch, a light module, a door lock, and a water valve (Figure 10) that are dispersed throughout the network. It is apparent that each gateway device is configured with varying default settings (Table 3). Gate-

Figure 9. Gateways devices: (a) Gateway 1 - VeraEdge, (b) Gateway 2 - RaZberry Pi, and (c) Gateway 3 - Almond+ [FR15a].

ways 1 and 2 use a web UI, the only way to interact with either device. On the other hand, Gateway 3 has a touch screen that allows the user to configure all the settings without a web UI. However, in order for a user to access any HAN globally, a web UI is needed.

The Gateway 3 UI is enabled for this experiment to replicate how an actual Almond+ HAN with Internet accessibility would function. The UIs allow access to the gateways locally by navigating to the IP address of the device or globally through a mobile device application or another Internet connected computer.

As an attacker, the first step of reconnaissance is locating Z-Wave HANs. A likely approach is to use a Z-Wave sniffer [PLD14, BFH+15, HRRL16] to conduct Z-Wave warwalking similar to what is done in [RMW12, KP12]. Next, the attacker

Table 3. Z-Wave gateway default configurations [FR15a].

Gateway Device	Web UI Enabled	SSH Enabled	Web UI Authentication
RaZberry Pi	Yes	No	No
VeraEdge	Yes	Yes	No
Almond+	No	No	Yes

gains access to the WLAN by either associating with an unsecured network using one of the many freely available tools to penetrate the WLAN defense. As previously discussed, the likelihood of an unsecured or poorly secured WLAN is relatively high (Section 2.4). Furthermore, the likelihood of an attacker gaining access to the target WLAN is proportional to its insecurity.

After gaining access to the target network, network scanning is conducted using *Zenmap* to locate the IP address of the target gateway device. However, before scanning the network, fingerprints are needed to identify each device (Table 4). Using tools similar to EZ-Wave [HRRL16], an attacker can perform network reconnaissance to identify the make and model of the Z-Wave gateway and then conduct basic research to identify fingerprints. If an attacker prefers stealth over speed the fingerprints are used to reduce the search space, limiting the amount of traffic on the network. This tactic creates less "noise" and improves the chance that the attack goes unnoticed. After locating the device, it is now possible to navigate to the gateway UI. The RaZberry Pi Documentation, freely available on their website [RaZ15], lists the UI location at Uniform Resource Locator (URL) `http://x.x.x.x:8083` (`x.x.x.x` denotes the IP address of Gateway 2), which confirms the Zenmap scan results (Fig-

Table 4. Z-Wave gateway fingerprints [FR15a].

Gateway Device	Open Ports	Unique Feature
RaZberry Pi	22*, 443	**MAC address:**
	8083, 8084	(Raspberry Pi Foundation)
VeraEdge	80, 22, 53	**OS:** OpenWrt Barrier
	49451	Breaker
Almond+	80, 22*,	**OS:** OpenWrt Kamikaze
	8200**	Backfire

*Open if SSH is enabled.
**Open if UPnP is enabled.

Figure 10. Z-Wave devices that are connected to each gateway network: (a) smart switch, (b) light module (c) door lock, and (d) water valve [FR15a].

ure 11). The UIs for Gateways 1 and 3 are located at URL `http://y.y.y.y` or `http://z.z.z.z`, where `y.y.y.y` denotes the IP address of Gateway 1 and `z.z.z.z` denotes the IP address of Gateway 3.

After navigating to each UI, Z-Wave HAN access is achieved for each gateway. It is possible to actuate all Z-Wave devices from the UI. This approach seems somewhat naive. If an attacker has access to the UI simultaneously with the user, the attack will be discovered and the user will take immediate action. It is necessary to explore additional options to remain inconspicuous while compromising the Z-Wave HAN.

3.2.2 Gateway Vulnerabilities.

During reconnaissance, it is observed that all gateway controllers use HTTP `POST` and `GET` requests to send commands to their server, which in turn relays information to the embedded Z-Wave chip that transmits wireless packets. This allows for the capture and replay of HTTP request to the Z-Wave gateway using Burp Proxy. The packets are accepted as legitimate requests as if from the UI. Modifying the packets before retransmission allows actuation of all the devices on the network. For instance, capturing requests sent to the smart switch could be easily modified to trigger a motion detector or open a water valve. Similar attacks are demonstrated in [CBS13,

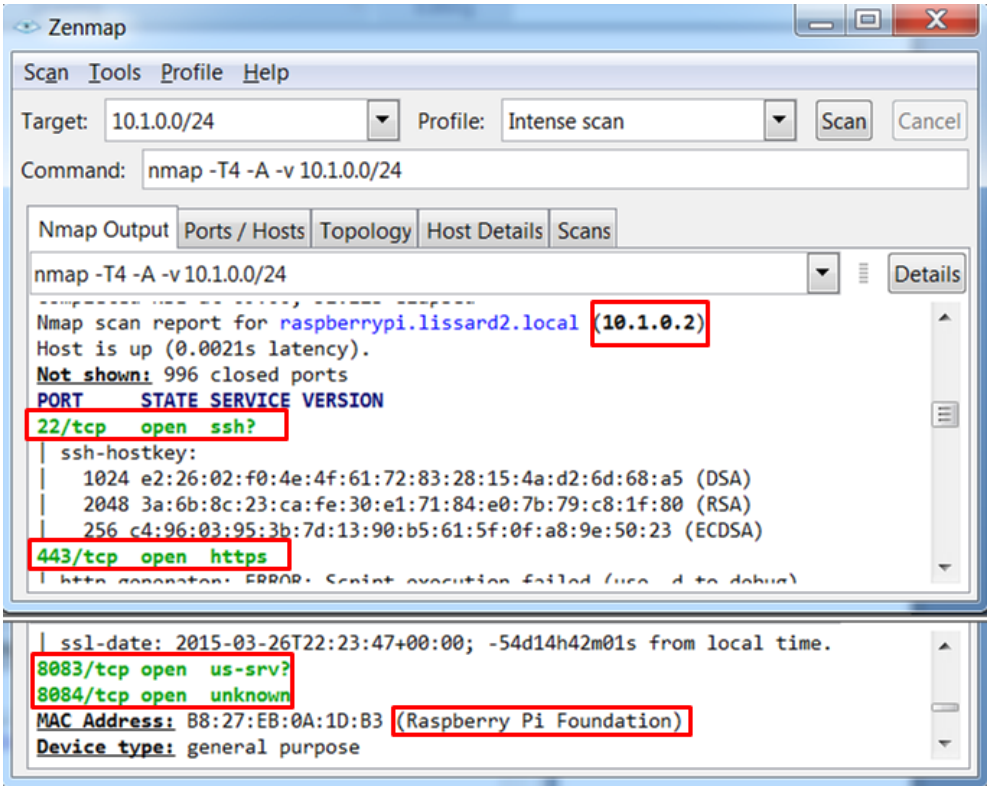


Figure 11. Zenmap scan showing Gateway 2 (RaZberry Pi) fingerprints and the IP address [FR15a].

OGPK13, BW15]. UI authentication credentials for the Almond+ are also captured because they are transported in the clear. Some vendors have taken action to fix discovered vulnerabilities [BW15], but multiple devices remain exploitable.

For the first time, exploits are demonstrated as found in [CBS13] on Gateway 1. Crowley et al. [CBS13] are able to successfully retrieve the VeraLite (same manufacturer of Gateway 1) backup file for the entire system. The backup file is used in case of a system malfunction. The user can reimage the system with the backup file to restore it to the original user settings. An attacker can use the backup file to retrieve passwords and sensitive system information that would aide their exploitation of the target network. After navigating to `http://y.y.y.y/cgi-bin/cmh/backup.sh?external=1`, the backup file is successfully downloaded from Gateway 1. Passwords are stored in the backup file, including the SSH password. Since SSH is enabled

(Table 4), *Putty* is used to login to the backend of Gateway 1. All other exploits demonstrated in [CBS13] are also successfully replicated on Gateway 1. This confirms the hypothesis that some manufactures either cannot or will not find solutions to existing vulnerabilities and that many of the same vulnerabilities exist in devices from the same manufacturer. This exploit is also applied to Gateway 2. By navigating to `http://x.x.x.x:8083/ZWaveAPI/Backup`, the attacker will download the backup files that contain Z-Wave device information. This is useful if the attacker wants to determine what types of devices are connected to the target Z-Wave HAN.

After testing exploits on all gateways under test that Crowley et al. [CBS13] demonstrate on the VeraLite, it is evident that an attacker will need to exercise some caution while manipulating the gateway UI. If the user becomes suspicious of an attack, they can check the gateway log file that exists on all devices tested. The log file keeps a record of all actions on the network. If an attacker attempts to perform an unauthorized firmware update [CBS13, BW15], exploit insufficient authentication checks on the gateway [CBS13], or other gateway attacks [OGPK13, BW15], all of the activity is recorded in the log file. Every time the attacker takes an action, the log file will capture the event. Once confirming their suspicion, the user takes action and possibly disconnects the gateway from the Internet or disables it completely preventing further access. Interestingly, this research finds that there is a way to gain access to a Z-Wave gateway and create a persistent connection to all Z-Wave devices while creating minimal log entries.

3.2.3 Adding a Rogue Controller to the Z-Wave Network.

In a Z-Wave HAN, if the user wants to add a new device to their network, the controller is placed in *inclusion* mode. Inclusion occurs when the primary controller (gateway device) in the Z-Wave HAN includes other devices in the network by as-

signing them its Home ID. When the network owner sets the device they intend to add in inclusion mode, the device accepts the Home ID of the controller and a new Node ID and joins the HAN. In Z-Wave HANs, the user must physically activate the inclusion mode on devices to add it to a network. However, there are two ways to put the controller into inclusion mode: UI inclusion and physical access inclusion. Rogue controller injection is demonstrated on the gateway devices using the UI inclusion mode.

3.2.3.1 Gateway 1: Rogue Controller Injection.

Gateway 1 allows a user to perform inclusion from the UI. UI inclusion allows a user to activate inclusion mode from the UI by the click of a button activating an HTTP request packet that activates inclusion mode on the Z-Wave chip. Once in inclusion mode, the gateway device is ready to accept the addition of any new devices. This feature allows an attacker to gain access to the WLAN once and make a copy of the Z-Wave HAN configuration, including Home ID and Node IDs. Doing so will allow stealthy persistent access and full control of devices even if the user decides to remove their gateway from the Internet.

To test this hypothesis, Gateway 1 is set to inclusion mode by injecting a previously captured HTTP packet creating one log entry. Once in inclusion mode, it is possible to use an Aspire RF Wireless controller to copy information from the gateway controller by setting it to *replicate* mode. The Aspire RF Wireless controller automatically connects to the gateway and, since the gateway is in inclusion mode, it transfers all of its information to the Aspire RF Wireless controller, creating a second log entry. The device settings on the Aspire RF Wireless controller now list many of the devices of the target network. After adding the rogue controller to the network, the UI displays the newly added device. The rogue controller is then deleted from the

UI so it is not visible to the target user. This creates a third log entry. Deleting the rogue controller from the HAN does not affect its access since Gateway 1 does not validate device deletion or exclusion. Full control and access is maintained to most devices while only creating three log entries.

3.2.3.2 Gateway 2: Rogue Controller Injection.

This exploit is attempted on Gateway 2. After injecting an inclusion packet to the gateway, an attempt to replicate the Z-Wave HAN configuration is made. It successfully connects to Gateway 2 and replicates the entire Z-Wave HAN configuration. The rogue controller is used to actuate devices on the target network. An attempt to remove the rogue controller from the UI is made but proves unsuccessful. Gateway 2, unlike the Gateway 1, validates that device deletion or exclusion occurs. Therefore, a device is only removed from the UI if it is actually removed from the Z-Wave HAN. Two portable Z-Wave controllers, a primary and secondary, are used to circumvent this Gateway 2 feature. An inclusion packet is sent to the target gateway. Once in inclusion mode, the secondary rogue controller is set to replicate mode, then immediately activate replicate mode on the primary rogue controller. While the secondary rogue controller is being included in the Z-Wave HAN, the primary rogue controller sniffs all traffic between the gateway device and the secondary rogue controller. Therefore, the primary rogue controller is never actually added to the Z-Wave HAN; it only captures all its network information. The secondary rogue controller is permanently excluded from Gateway 2 removing it from the UI. The primary rogue controller has full access to most devices on the Z-Wave HAN.

3.2.3.3 Gateway 3: Rogue Controller Injection.

Gateway 3 is unlike the previous devices. It requires the user to physically put the controller into inclusion mode. Therefore, an attacker without physical access to Gateway 3 is unable to inject a rogue device into the network.

The rogue controller injected in Gateways 1 and 2 only communicates in the sub-GHz spectrum and not via HTTP requests (Figure 12). Any communication between the new controller and Z-Wave devices goes undetected by both the HAN gateway and legitimate users. Unlike previous exploits, even if the gateway looses power, Internet connection, or the user willingly disconnects it from the WLAN, the rogue controller has persistent access to the Z-Wave devices. Using EZ-Wave [HRRL16],

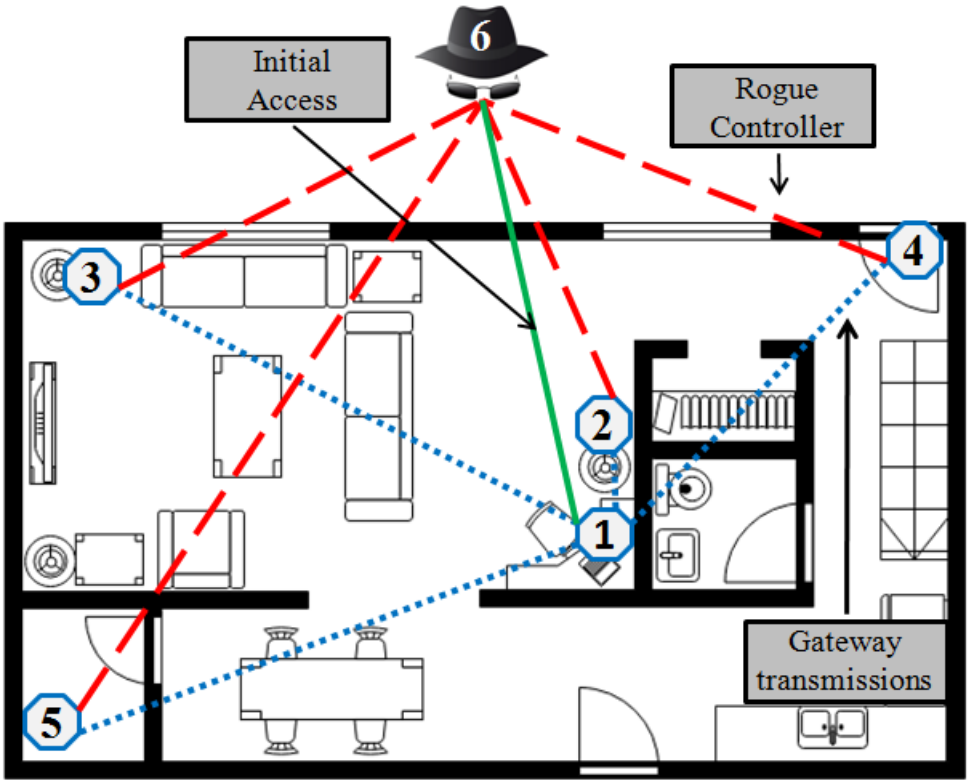


Figure 12. Actual test environment for rogue controller injection. (1) Represents the Z-Wave gateway, (2-5) represents Z-Wave devices, and (6) represents the attacker. Unlike previous gateway exploits, the rogue controller transmits commands directly to Z-Wave devices, bypassing the gateway [FR15a].

an attacker can accurately fingerprint Z-Wave devices that are vulnerable to rogue controller injection and exploit them.

3.3 Hel Attack

Now that access to the Z-Wave gateway is achieved, another gateway exploit is employed, the Hel Attack. Hel is coined because in old Norse mythology, Hel means hidden [Ore03]. Since the attack involves hidden malware and hidden information in a Z-Wave packet, Hel is a fitting title.

In order to conduct the Hel Attack in Z-Wave networks, it must be possible to hide information in Z-Wave packets. Motivated by previous work that successfully demonstrates this technique in IEEE 802.15.4 networks [MG10], this research further investigate the feasibility of this technique in ITU-T G.9959-based networks, specifically Z-Wave.

3.3.1 Information Hiding in the Z-Wave MAC Frame.

This work demonstrates the possibility of hiding data in the Z-Wave MAC frame similar to what was accomplished in IEEE 802.15.4-based systems by [MG10]. As discussed in Section 2.2, the MAC layer supports three frames. The maximum size of the payload in each frame differs depending on the data rates and devices used. Although newly released Gen5 Z-Wave devices can operate at rate R3, the focus is on identifying positions to hide information in the MAC frame at rates R1 and R2 for singlecast (acknowledgment messages are a subset of singlecast) and multicast messages (Figure 13). The maximum packet size at rates R1 and R2 is 64B. This evaluation represents the most challenging case for information hiding since the payload size available to hide information at R1 and R2 are less than half of R3.

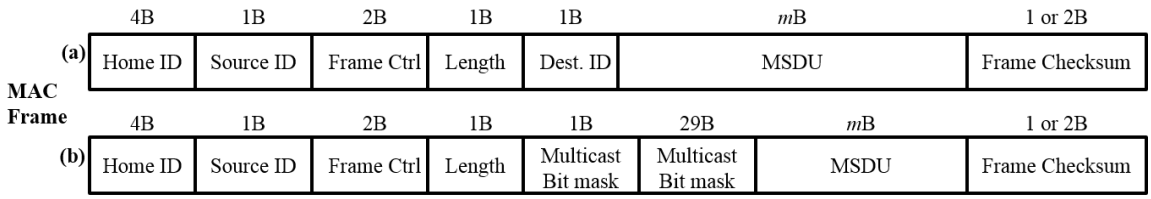


Figure 13. Z-Wave MAC frame structure: (a) singlecast frame and (b) multicast frame. [FRPR16].

Singlecast MAC frames includes the Home ID, Source ID, Frame Control, Length, and Destination ID totaling 9B (Figure 13). A one-byte non-correcting frame checksum is used to validate the MAC frame for data rates R1 and R2 (2B are used for R3). Given the channel configuration, the frame checksum of messages supported in this experimental Z-Wave network is one byte. Thus, there are 54B remaining for the MSDU (frame payload). A similar calculation is done to discover the maximum payload length of the multicast frame (Figure 13). There are 39B used in the multicast frame excluding the MSDU. Therefore, the MSDU is 25B.

3.3.2 Space Available to Hide Information.

The MSDU field has variable length and contains the frame payload information. The length depends on the command being sent. Commands are specified in Z-Wave Command Classes (Appendix A). The Command Class determines the function that needs to be performed by a device. One example is the *Basic* Command Class which is supported by all Z-Wave devices.

Since all devices, including the controller, will accept messages containing the Basic Command Class [RaZ15], the length required in the MSDU for the Basic Command Class portion of the payload can be determined. This will provide necessary information to calculate the available bytes remaining.

The representation of the Basic Command Class in the MAC frame is `0x20`. The Command following is dependent upon a `SET (0x01)`, a `GET (0x02)`, or a `REPORT`

(0x03). After the Command, the Payload parameters follow and are no more than one byte. In either case, the payload length needed to support the Basic Command Class is 3B. Since all Z-Wave devices support the Basic Command Class, an attacker can hide up to 51B of data in a singlecast MAC frame and up to 22B in a multicast MAC frame when injected a packet using the Basic Command Class. When the Z-Wave transceiver receives a packet, the header and EoF are stripped away as the packet moves up the protocol stack. Once at the LLC Layer, the Command Class function, Command, and Payload parameters are executed and hidden bytes are ignored.

3.3.3 Hel Attack.

This section presents the first information hiding attack in a Z-Wave network. The experiment setup includes the target gateway device, Gateway 2, on a WLAN backbone, hardware and software components listed in Table 2, and the AFIT Sniffer.

It was shown in Section 3.2 and [FR15b] that it is possible to gain access to the target Z-Wave gateway. After gaining access, the attacker uploads a Python script that is then executed but sleeps for a time determined by the attacker (in our experiment - ten minutes). When re-access to the Z-Wave network is needed, the attacker first executes SSH server on their machine. Using Scapy-Radio [PLD14, BFH+15], the attacker crafts a Z-Wave packet containing the hidden information and transmits the packet to the Z-Wave gateway using an SDR. To ensure the injected packet is accepted by the Z-Wave network, the attacker must first capture a Z-Wave packet using tools such as those presented in [PLD14, BFH+15, HRRL16] and extract the Home ID to construct a packet similar to the one in Figure 14. Lastly, the attacker follows the ITU-T G.9959 specification and calculates the checksum using the eight-bit checksum algorithm in (Algorithm 1).

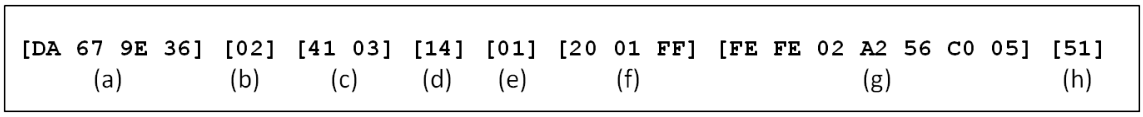


Figure 14. MAC frame consisting of (a) Home ID, (b) Source ID, (c) Frame Control, (e) Length, (e) Destination ID, (f) Basic Command Class, Command, and Payload, (g) hidden information in MSDU, and (h) Checksum. [FRPR16].

3.3.3.1 Hel Attack on Gateway 2.

This research demonstrates the Hel Attack on Gateway 2 (Figure 9). As a proof of concept, Gateway 2 provides the system architecture needed to execute the malware, written in Python 2.7. Although this attack is possible on other Z-Wave gateways, malware would need to be written specifically for each gateway design.

The key to this attack is locating Z-Wave packets on the gateway controller. Gateway 2 keeps a detailed log (Figure 15) that contains all actions on the Z-Wave network. Once awake, the Python script scans the log file for any injected packet containing the hidden information. The injected packet contains a marker `(FE FE)` in the MSDU (Figure 14). This marker is used because it is unlikely that a standard Z-Wave packet will contain consecutive byes `FE`. Upon identification, the Python script dissects the packet retrieving needed information including the attack type `(02)` and IP address `(A2-56-C0-05)`. Any attack type can be added depending on the needs of the attacker. Examples include Ping of Death to deny service or Reverse File Transfer to retrieve the `/etc/passwd` file or other important files.

Algorithm 1 One-byte frame check sequence algorithm [ITU15].

```
procedure BYTE GenerateCheckSum(BYTE *Data, BYTE Length)
    BYTE CheckSum = 0xFF
    for Length > 0; Length−− do
        Checksum = *Data++
    end for
    return Checksum
end procedure
```

```
[D] [zway] Job 0x13 (SwitchBinary Get): success                          (a)
[I] [zway] Removing job: SwitchBinary Get      (1)  (2)      (3)
[D] [zway] RECEIVED: ( 04 00 02 0A 20 01 FF FE FE 02 A2 56 C0 05 14)
[D] [zway] SENT ACK
[I] [zway] Job 0x13 (SwitchBinary Get): Delivered
```

```
[D] [zway] Job 0x13 (SwitchBinary Get): success                          (b)
[I] [zway] Removing job: SwitchBinary Get
[D] [zway] RECEIVED: ( 04 00 02 03 20 01 FF 2E )
[D] [zway] SENT ACK
[I] [zway] Job 0x13 (SwitchBinary Get): Delivered
```

Figure 15. Gateway 2 log file. (a) Log representation of manipulated Z-Wave packet containing information hidden in the MSDU. (1) Marker signifying that a packet has been injected, (2) attack type, and (3) IP address in hexadecimal representing 162.86.192.5. (b) Log representation of legitimate Z-Wave Packet [FRPR16].

After retrieving the hidden information, the Python script clears the log file and initiates a R-SSH client to the attack SSH server at IP address `162.86.192.5` (Figure 15). The attacker can now perform command line instructions as `root` user on the target gateway allowing access to Z-Wave devices and WLAN devices (Figure 16). Once complete, the attacker simply closes the connection which instructs the malware to sleep for the predetermined time. Given enough compromised Z-Wave gateways, an attacker can proceed with distributed attacks using the target gateways as botnets. Botnets take the form of globally distributed networks consisting of slave devices that act on the behest of the botmaster [SK14]. They are largely responsible for numerous financially motivated crimes, spamming, and distributed denial of service attacks.

Although this attack was conducted within 20 meters of the target location, more than enough distance for an attacker to remain inconspicuous, previous work demonstrates the exploitation of WSN devices from 64 km away [AP13]. Given improved tools, an attacker with this capability need not be in close proximity to the target location to inject packets into the network.

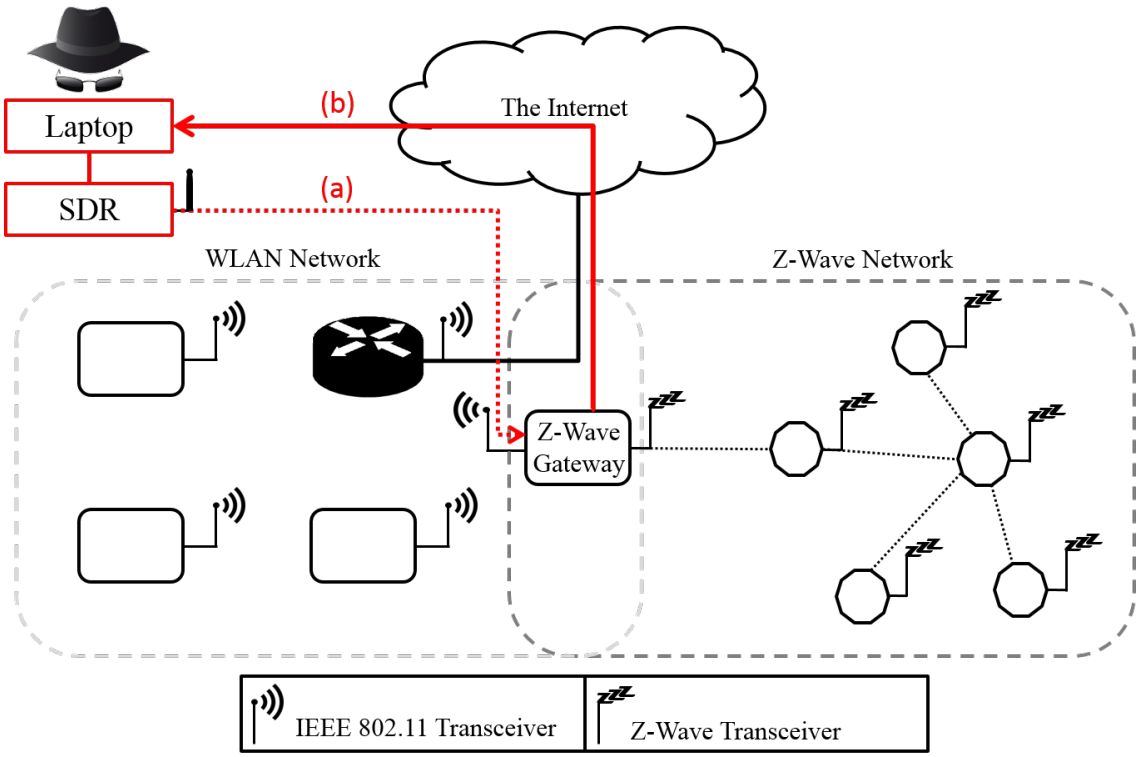


Figure 16. After the attacker executes a listening server, (a) a malicious Z-Wave packet is injected to the gateway. When the Python script on the infected gateway detects the injected frame, it retrieves the IP address and (b) opens a R-SSH. The attacker now has full control of the Z-Wave gateway and access to WLAN devices [FRPR16].

3.4 Summary

The attacks presented illustrate the effects of the numerous gateway insecurities and protocol vulnerabilities in Z-Wave networks. Both attacks provide control of the Z-Wave network with relatively minimal probability of detection. Given the new attacks illustrated herein: rogue controller injection and Hel attack, there is a need for defensive countermeasures. This research therefore develops an MBIDS capable of distinguishing between rogue devices and authorized devices, as well as manipulated packets and correctly formed packets allowing for the first monitored Z-Wave network.

IV. Methodology

4.1 Introduction

This chapter presents the methodology used to evaluate the MBIDS performance using three different experiments: detection of rogue device attacks, detection of manipulated packet injection attacks, and the increase misuse detection rate after enhancing the tool. First, the problem definition, goals and hypotheses, and approach are discussed in Section 4.2. Section 4.3 defines the system boundaries. The system and its services and outcomes are described in Section 4.4, followed by a detailed description of the parameters and factors in Section 4.5. Performance metrics are explained in Section 4.6. The evaluation technique is discussed in Section 4.7, followed by a description of the design of experiments in Section 4.8.

4.2 Problem Definition

4.2.1 Goals and Hypotheses.

An objective of this research is to investigate vulnerabilities and to develop exploits to establish the security implications of Z-Wave. Furthermore, given the established exploits, countermeasures are needed. This research begins with the development of an MBIDS hypothesized to detect device attacks and manipulated packet injection attacks.

The research then answers the following questions:

- What are misuse cases in Z-Wave network transmissions?
- What is the effectiveness of the MBIDS at detecting rogue controller injection attacks?

- What is its effectiveness of the MBIDS at detecting packet manipulation attacks, including the Hel Attack?
- Can the MBIDS be enhanced to increase the misuse detection rate through packet state analysis and routed packet evaluations, respectively?

It is hypothesized that capturing Z-Wave packets and dissecting them for byte evaluation as well as using open source documentation allows the development of misuse cases after determining known-good cases. It is also hypothesized that because rogue Z-Wave devices have a node ID that is not actually part of the Z-Wave network, an IDS capable of gathering all valid node IDs is able to accurately validate group membership. In addition, the system can be expanded to include evaluations of Z-Wave Command Classes, Commands, and Payload parameters allowing the detection of manipulated packet injection attacks, including the Hel Attack. Furthermore, it is hypothesized that if all received packets with Source ID `0x01` are compared with the Z-Wave gateway log file, the MBIDS accurately distinguishes between an actual packet generated by the Z-Wave gateway and a maliciously injected packet. Lastly, since routed packets are generated similarly to standard singlecast packets, the MBIDS may improperly classify routed packets. It is therefore hypothesized that reverse engineering the routing protocol allows for an in-depth dissection of routed packets providing the MBIDS the ability to properly evaluate them.

4.2.2 Approach.

The MBIDS tool, *MBIDS.py*, evaluates packets captured by the AFIT Sniffer. The tool uses signatures and states that are derived from an in depth study of Z-Wave packet transmissions. After dissection, packets are compared with the ITU-T G.9959 specification in order to further understand the byte fields. This allows the development of the signature database and protocol states for packet comparison. The

authors also employ the *OpenZWave* framework, an open-source project that reverse engineered the closed-source serial application interface for communicating with the Z-Wave chip [Ope14]. OpenZWave also provides partial support for 53 Command Classes. Similarly, the *Z-Way Developer's Documentation* also provides information about 40 known Command Classes [RaZ15].

A realistic Z-Wave network is engineered with a gateway controller and multiple devices on a WLAN backbone. The accuracy of the MBIDS is then tested against two new attacks presented in Chapter III to evaluate the efficacy of this approach.

Figure 17 illustrates the overall functionality of the design. When the system receives a packet, the following occurs:

1. Packets are captured by the AFIT Sniffer for evaluation by the packet monitoring tool, *MBIDS.py*.

2. Upon receipt of the packet, the MBIDS dissects the packet and checks for a valid checksum. If the checksum is not valid, the packet is immediately discarded. If it is valid, further evaluation occurs.

   The Z-Wave protocol automatically discards packets with invalid checksums, so the Z-Wave gateway will not accept them. Therefore, the MBIDS does not need to evaluate them.

3. The next item to evaluate is packet length. As discussed in Section 2.2.1, the maximum payload length while operating at R3 is 170B and 64B if operation at R1 or R2. Therefore, if the packet is outside the normal bounds, it is a misuse case since it violates standard packet length. In this case, an attacker is attempting to inject significant amounts of data in one single packet. If the packet length is valid, further evaluation occurs.

4. Once the packet length is validated, group membership is then evaluated. Upon initialization of the MBIDS, the Z-Wave gateway is polled for information. When a device is included in the Z-Wave gateway, the gateway keeps a database

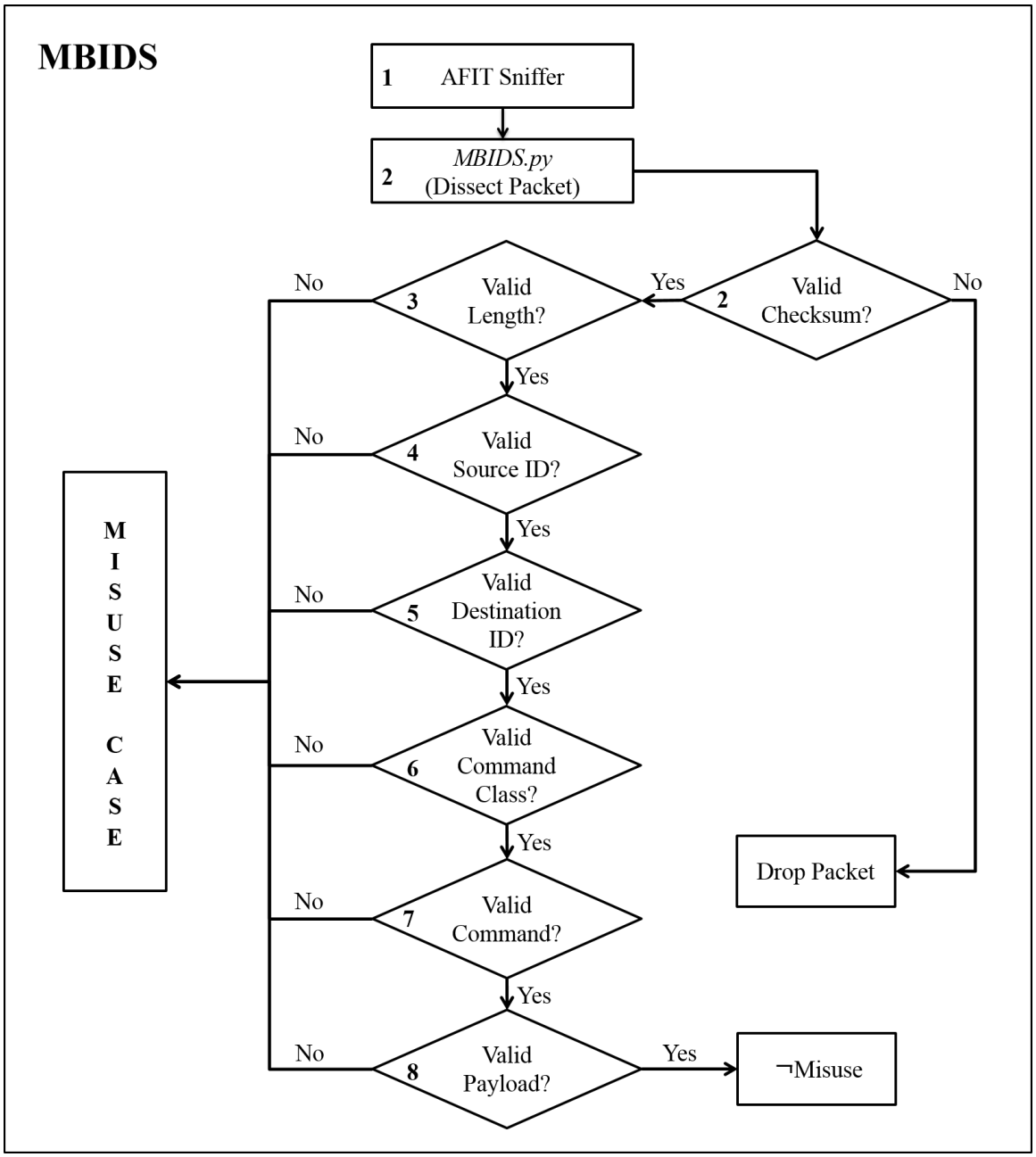


Figure 17. Packet data flow through the MBIDS. If the packet has an invalid checksum, it will immediately be dropped. If the packet violates any misuse case, it is logged. Otherwise, it is not a misuse case.

containing all Node IDs and all Command Classes that each device supports. Thus, the MBIDS can use this information to distinguish between a packet sent from a device that is part of the group of trusted Z-Wave devices or not. If the Source ID of the evaluated packet is identified as an existing node on the network, further packet evaluation occurs, else, the packet is considered a misuse case.

5. Next, the MBIDS evaluates the Destination ID field of the packet. If the Destination ID is identified as an existing node on the network, further packet evaluation occurs, else, the packet is considered a misuse case.

6. After group membership is confirmed, the MBIDS evaluates the Command Class sent by a device. Individual Z-Wave devices support different Command Classes. For example, a lamp module does not support the `ALARM` Command Class and an alarm does not support a `DOOR LOCK` Command Class. Therefore, if an attacker injects a packet with a valid Source ID and Destination ID, but a Command Class that is not supported by the Destination ID on the Z-Wave network, the packet is considered a misuse case. Otherwise, further evaluation occurs.

7. Once the Command Class is validated, the Command is checked. Known Commands are derived from OpenZWave and the *Z-Way Developer's Documentation*. For example, the *Basic* Command Class only supports Commands `SET`, `GET`, and `REPORT`. Any other Commands are invalid and considered a misuse case.

8. Lastly, the MSDU is evaluated. The Command Class and Command are part of the MSDU. Any byte field after the Command is referred to as the Payload. Although a Command may be valid and its associated payload triggers an action,

it is possible to pad extra bytes in the Payload that are ignored when received by a Z-Wave gateway. If the Payload is invalid, it is considered a misuse case. Otherwise, the packet is not ($\neg$) a misuse case.

The Z-Wave protocol supports over 70 known Command Classes. In order to develop an MBIDS, it is required that every Command Class is properly understood to detect any violation of signatures or states (Signature-Based and Stateful Protocol Analysis). As discussed, the *Basic* Command Class has three commands, `SET`, `GET`, and `REPORT`. There are five combinations of *Basic* commands supported by the Z-Wave protocol. As the most basic Command Class, *Basic* has the fewest commands available. At a modest estimation, over 70 known Command Classes would allow for over 2,000 possible combinations to account for in the MBIDS. Each Command Class has a subsequence Command between `0x00` to `0xFF` and Payload parameters 1B to 52B where each byte is a value `0x00` to `0xFF`. Therefore, this work provides a proof of concept with 11 of the most common Command Classes to illustrate the efficacy of this approach. To this end, deep packet inspection of normal payloads is conducted to develop known-good packet standards (signatures and states) that injected packets are compared against.

This research does not fully consider the two-byte Frame Control field. The Frame Control contains information defining the frame type and various flags. The Header flag defines the type of frame that is sent. Frame types include singlecast, multicast, and acknowledgment. There are also reserved bits the ITU-T G.9959 recommendation states should not be used. Given this information, identifying known-good Frame Control bytes seems trivial. However, some vendors disregard the reserved bits and modify their protocol implementation to include unsupported Header types. For example, the ITU-T G.9959 states that the Header type value `0x04 - 0x07` should not be used [ITU15]. It is observed that Gateways 1 and 2 includes the reserve Header

type `0x05`. This one example illustrates the infeasibility of accounting for all known-good Header types. Fully integrating Header types checks in the MBIDS increases improper packet classification reducing the overall effectiveness of the system.

As show in Figure 18, this research is divided into three experiments: (i) evaluating the effectiveness of the MBIDS to detect rogue devices including rogue controller injection presented in Section 3.2, (ii) evaluating the effectiveness of the MBIDS to detect manipulated packet injection attacks including the Hel attack as presented in Section 3.3, and (iii) evaluating the enhancement strategy for increasing the mean detection rate of the MBIDS. Each experiment consists of numerous tests that are discussed in detail in Section 4.8, but a brief description is given below.

4.2.2.1 Experiment 1: Rogue Device Detection.

The first experiment evaluates the detection rate of the MBIDS against rogue device attacks. This experiment is split into three parts. In each part, a series of packets are injected to the Z-Wave network. Concurrently, the MBIDS captures those packets and evaluates them for group membership violations (based on known signatures) signifying a rogue device. The detection rate is calculated based on the quantity of violations detected and logged versus the number of attack packets sent.

4.2.2.2 Experiment 2: Manipulated Packet Injection Detection.

The second experiment reveals whether the MBIDS is capable of detecting manipulated packet injection attacks against a Z-Wave network and then determines the detection rate of the MBIDS against the attacks. A series of attack packets are injected into the Z-Wave network. The MBIDS captures those packets and evaluates them for any violations signifying a maliciously injected packet. The detection rate is

calculated based on the quantity of violations detected and logged versus the number of attack packets sent.

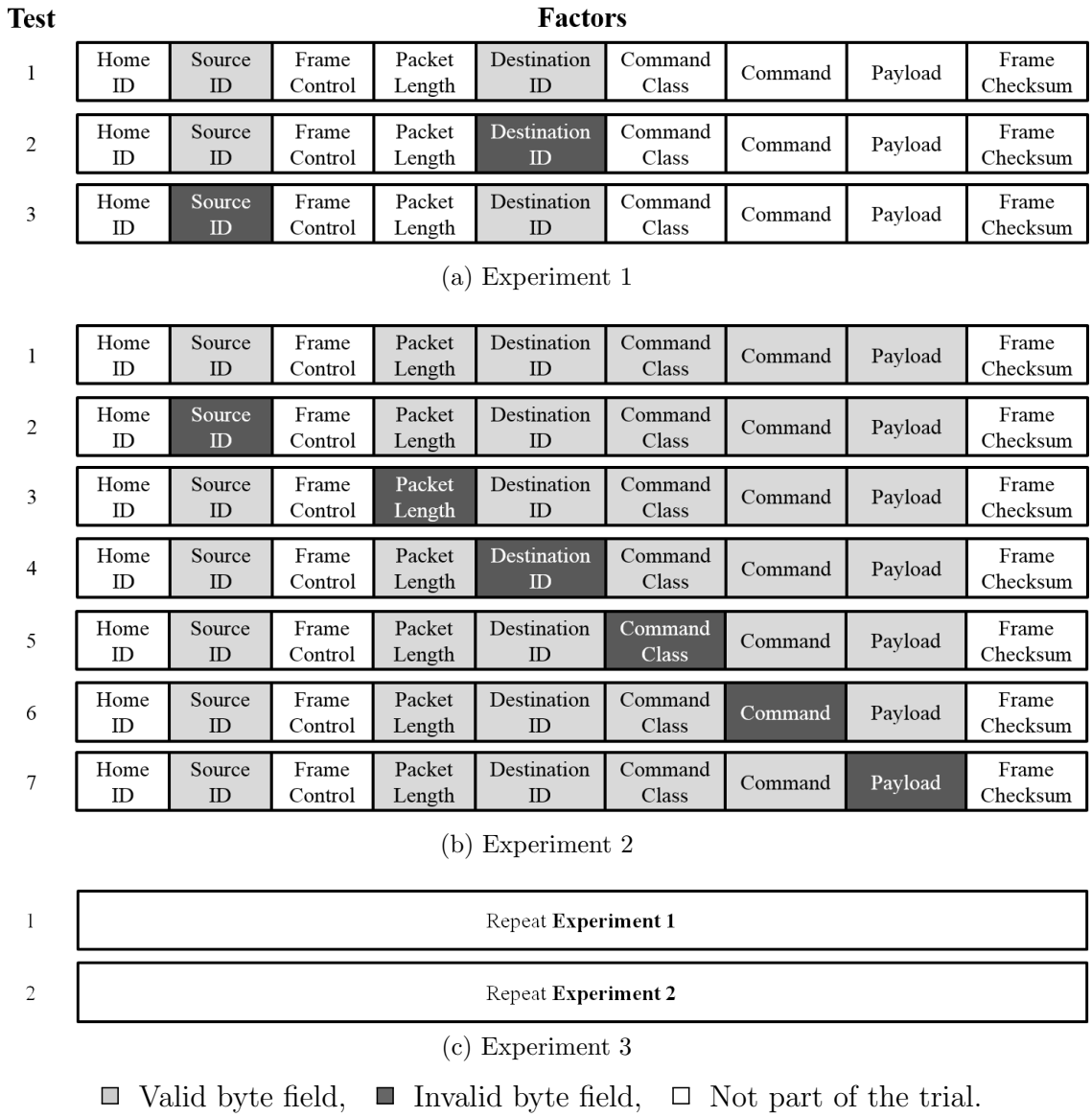


Figure 18. Experiments and tests used to achieve the research goals. (a) Experiment 1 consists of 3 tests. (b) Experiment 2 consists of 7 tests. (c) Experiment 3 repeats both previous experiments in order to compare results of the enhancement strategy with previous results.

4.2.2.3 Experiment 3: Detection Rate Post Enhancement.

Inevitably, with any IDS, there are packets that are falsely classified as normal. The goal of any IDS is to detect violations while maximizing accurate classifications and minimizing incorrect classifications. This is done by developing enhancement strategies to accurately evaluated routed packets and re-evaluate improperly classified packets for misuse.

As discussed in Section 2.6, Stateful Protocol Analysis can identify non-standard sequence of commands. Using this method, captured packets that appear to be from the controller (Source ID `0x01`) are re-evaluated to ensure that the controller actually sent them.

Although the MBIDS does not fully integrate Frame Control field checks, as a proof of concept, it evaluates singlecast frames (Header Type `0x01`). If a routed frame is received, the Header Type is also `0x01` but the routed flag is set. When this occurs, the MBIDS improperly classifies the packet as a singlecast frame with Command Class `0x00` (No Operation - Appendix A) instead of a routed frame. After reverse engineering the routing protocol, its evaluation is included in the MBIDS to increase the misuse detection rate.

Experiments 1 and 2 are retested as subset tests of Experiment 3. The results of Experiment 3 are compared with results from the first two experiments to determine if the overall performance of the MBIDS has increased.

4.3 System Boundaries

The System Under Test (SUT) is the MBIDS, a monitoring tool for Z-Wave networks capable of detecting attacks presented in Chapter III and other packet manipulation attacks in Z-Wave networks. A block diagram of the SUT is illustrated

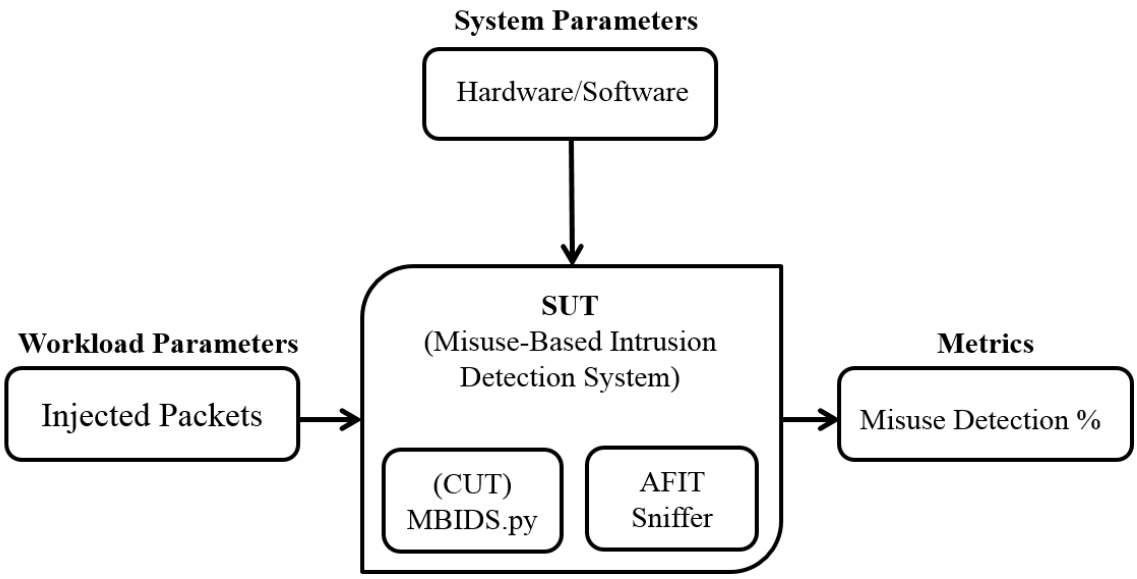


Figure 19. System under test diagram for the MBIDS [FRPR16].

in Figure 19. The SUT consists of the Component Under Test (CUT): *MBIDS.py* (packet evaluation software) and the AFIT Sniffer [BFH+15].

Workload parameters include crafted packets that are injected into the SUT. The system parameters consist of the hardware and software components. These parameters are discussed in detail in Section 4.5. The metrics used to evaluate the efficiency of the MBIDS is the probability of packet capture of the AFIT Sniffer and the misuse detection rate of *MBIDS.py*.

4.4 System Services and Outcomes

The service that the SUT provides is monitoring of Z-Wave networks where violations are detected and logged for further review by an administrator or investigator.

There are two outcomes for the SUT: success or failure. The service is successful when, after capturing and parsing a received packet, the packet is deemed a known-good packet or a misuse case. If it is a misuse case, the packet is logged for further review by an administrator or investigator. The service is a failure when a packet is

completely missed by the packet capture device or the captured packet is incorrectly classified.

4.5 Parameters and Factors

The parameters of the system are the properties which, if changed, impacts the performance of the system. These include both workload parameters, which characterize the workload and system parameters which characterize the system; parameters that vary are called *factors*. The system and workload parameters of the SUT are described below.

4.5.1 Workload Parameters.

The workload of the SUT consists of multiple factors. Factors include Packet Length, Source ID, Destination ID, Command Class, Command, and Payload. Each packet sent to the SUT is generated with varying factors in order to ensure the SUT is being tested from a representative pool of all possible generated packets. The factors are discussed in detail below.

4.5.1.1 Packet Length.

As discussed in Section 2.2, the Z-Wave MAC frame does not exceed 64B. Therefore, when the MBIDS receives packets, the length is checked for validity. If it exceeds 64B, the packet is deemed invalid and the misuse is logged. When this factor is varied, packets with different lengths are injected to the SUT.

4.5.1.2 Source ID and Destination ID.

The Z-Wave gateway under test keeps a record of all device Node IDs for each device include in the network. When a packet is received, the MBIDS requests this

information from the Z-Wave gateway and checks the Source ID and Destination ID in the Z-Wave MAC frame. If they are invalid, the misuse is logged. Otherwise, evaluation of the Command Class begins. The boundaries of this factor are Node IDs `0x01` to `0xE8` given 232 allowable nodes in any given Z-Wave HAN.

4.5.1.3 Command Class.

Each Z-Wave device supports specific command classes. For instance, a Z-Wave door lock supports Command Class `0x62` but not Command Class `0x33` (Switch Color Command Class). If the door lock receives a packet with Command Class `0x33` or any other unsupported Command Class, the packet is classified as a misuse case and logged. If the Command Class is supported, the Command parameter is then evaluated. This factor varies depending on a choice of one of the 70 known Command Classes (Appendix A).

4.5.1.4 Command.

The Command specifies one function within a specific Command Class. For example, the *Basic* Command Class (`0x20`) supports three functions: `SET` or `0x01`, `GET` or `0x02`, and `REPORT` or `0x03`. When the Command Class is validated, the Command is evaluated. Commands for all known Command Classes are listed in Appendix A.

4.5.1.5 Payload.

The next step in the evaluation of the Z-Wave MAC frames is Payload inspection. After the Command has been verified as supported, the payload must be checked for abnormalities. For example, if an attacker injects a packet into the network where the MSDU is as shown in Figure 20, the MBIDS will classify it is as the *Basic* Command Class `(0x20)` with a SET Command `(0x01)` and state on `(0xFF)`. This

requires an MSDU of length 3B but contains 10B. Therefore, the MBIDS will log the misuse since the Payload is abnormal given the specified command class. After the Command Class and Command have been selected, 52B remain in the MSDU (Section 2.2). This variable factor consists of Payloads 1B to 52B.

4.5.2 System Parameters.

The system parameters include hardware and software components that are discussed in detail in Section 4.7. However, the primary components include the AFIT Sniffer as a packet capture device and the IDS software tool for packet evaluation, *MBIDS.py*.

4.6 Performance Metrics

In order for the system to be effective, it must have a high probability of successfully parsing and evaluating received packets for misuse cases. By extension, in order for the system to successfully evaluate these packets, it must have the capability to capture all traffic in the Z-Wave network, which necessitates the requirement of using an effective packet capture device. Thus, the accuracy of the packet capture device used in this experiment is evaluated.

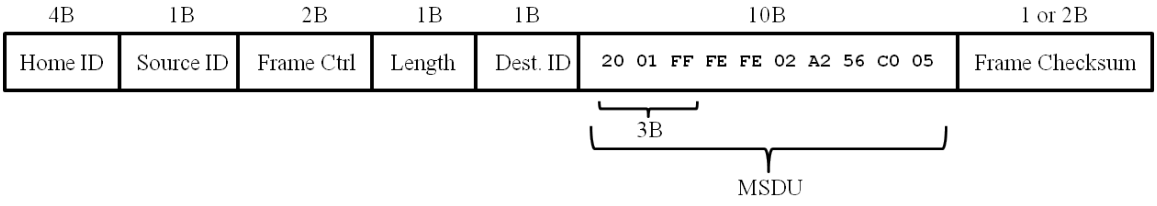


Figure 20. Example payload of injected packet

4.6.1 Packet Capture Device - AFIT Sniffer.

The AFIT Sniffer includes a USRP N210 and VERT900 3dBi antenna including an extension of the software framework proposed in [PLD14]. The efficiency of the AFIT Sniffer and Gateway 2 are tested to determine the best case misuse detection rate of the MBIDS given the underlying packet capture device.

Five hundred tests are conducted on each device wherein each test consists of sending one packet to Gateway 2. The mean packet reception rate for the Gateway 2 is 99.9% whereas the AFIT Sniffer maintains a mean packet reception rate of $\approx 94\%$ (Figure 21).

The RF front end on Gateway 2 is tuned per manufacturer specifications and is more effective than the SDR. Given the $\approx 6\%$ error rate of the AFIT Sniffer, it is know a priori that the MBIDS will not detect 100% of manipulated packets even in a perfect test. Therefore, in a realistic employment, the performance of the MBIDS is considered with respect to the packet capture accuracy of the AFIT Sniffer or any underlying SDR.

4.7 Evaluation Technique and Environment

This section discusses the experiment environment including software and hardware components used and evaluation techniques used to conduct the experiments and gather results for analysis.

4.7.1 Evaluation Techniques.

Overall, the results are evaluated by performing a statistical analysis of the performance metrics. Standard statistics such as the standard deviation, mean, and 99% confidence interval for all collected data are computed.

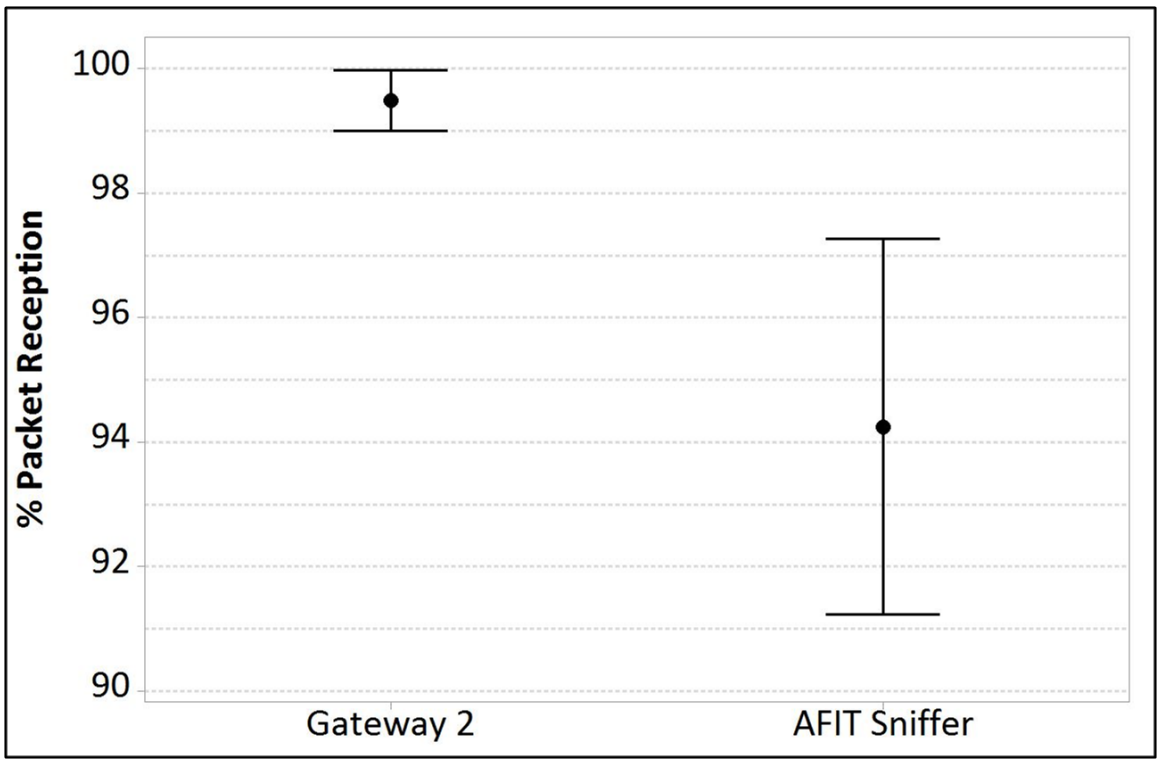


Figure 21. Gateway 2 and AFIT Sniffer packet reception rates (99% confidence interval for the mean) [FRPR16].

In all three experiments, collected data is binary, meaning that results are 1 for captured packets correctly classified as misuse cases and 0 for packets classified as known-good packets (whether properly or improperly classified). In Experiment 3, the data from Experiments 1 and 2 are re-evaluated using the enhancement strategy.

4.7.2 Experimental Environment.

To conduct the three experiments outlined in Section 4.2.2, the experiment setup is illustrated in Figure 22 and pictured in Figure 23. The setup consists of a target network with components listed Table 5(a) and the attacker system with components listed in Table 5(b).

4.8 Design of Experiments

The overall design for this study consists of three experiments. The design consists of a total of 6,000 trials, where each trial consists of a crafted packet sent to the Z-Wave network. Because the underlying AFIT Sniffer is not 100% accurate at packet capture, trials are conducted until the required number of packets are evaluated by the MBIDS. Details of the trial distribution between the experiments and tests are outlined below.

4.8.1 Experiment 1: Rogue Device Detection.

Experiment 1 consists of three tests totaling 900 trials. When a rogue device is injected in a Z-Wave network, it receives a unique Node ID. An attacker uses two portable rogue controllers and removes one from the network to cover their tracks (Section 3.2). Therefore, the remaining rogue controller has a Node ID that is not

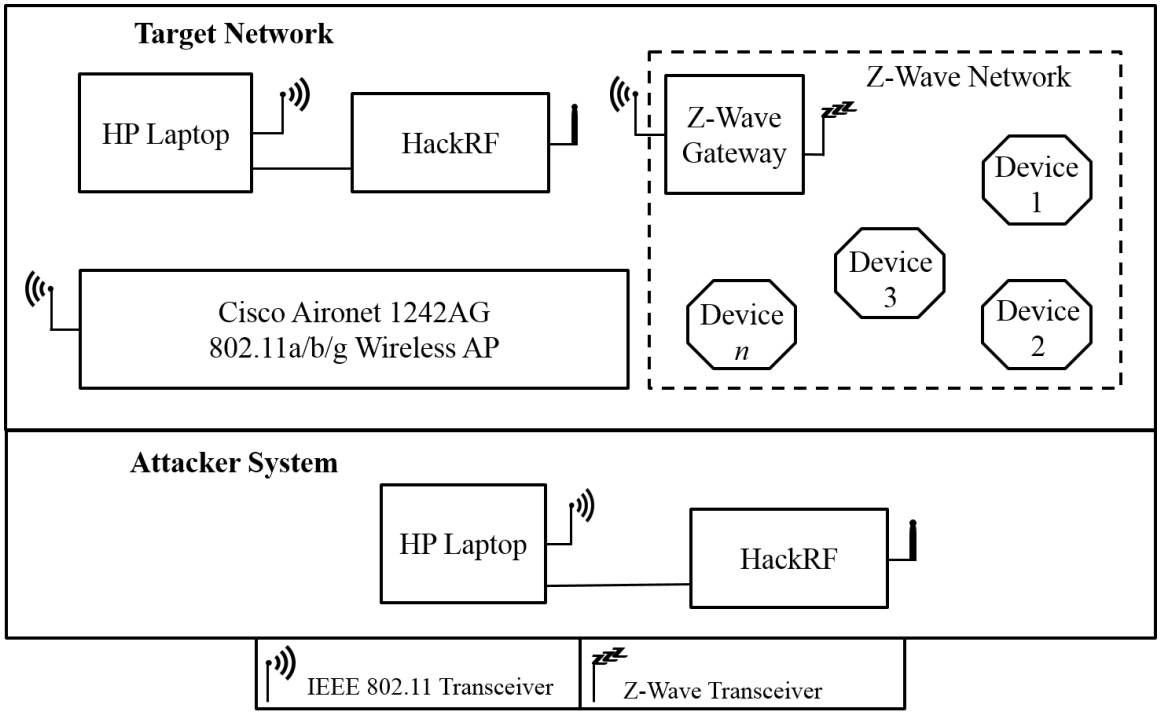


Figure 22. Block diagram of the experiment setup

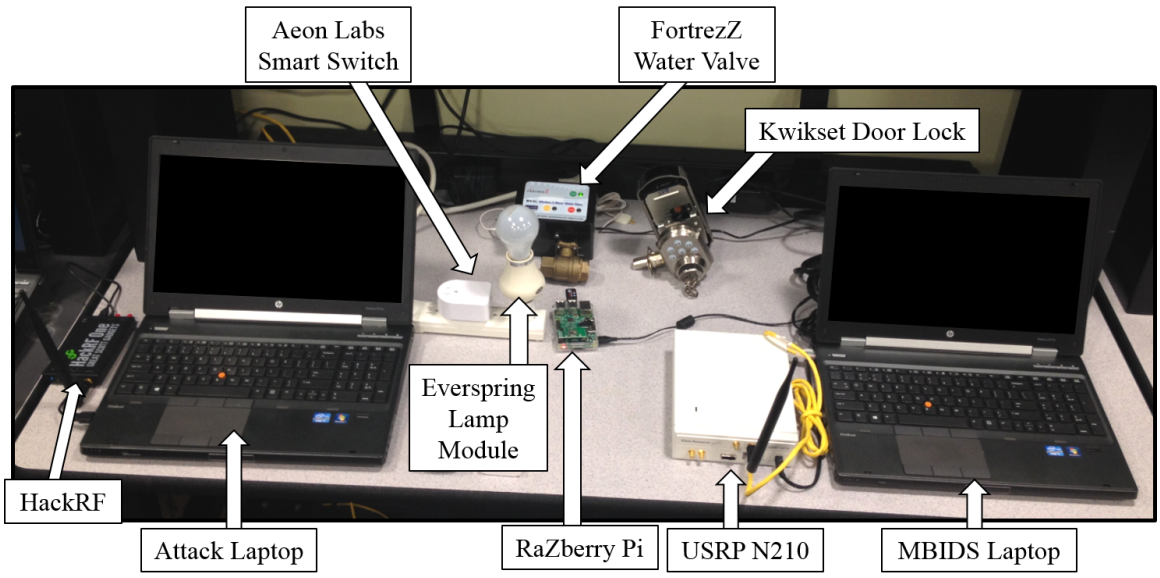


Figure 23. Actual experiment setup

represented on the Z-Wave gateway. The MBIDS verifies group membership by validating the Source ID and Destination ID in all received packets. If a packet is received with invalid membership IDs, it is considered a misuse case. These tests, therefore, ensure that candidates from all known possibilities are represented.

There are 232 nodes allowable in any Z-Wave network. An exhaustive test would require 232^2 trials to test every possible combination of Source ID and Destination ID. As a proof of concept, this work scopes the trials to include a representative set of all possible combinations. Since the MBIDS encoding uses conditional constructs, the set of all possible combinations does not include a case where more than one byte field is invalid. In a case where more than one byte field is invalid, the MBIDS logs a misuse case based on the first invalid byte field discovered. This allows a packet to be processed as quickly as possible. Combinations include:

- Test 1: Valid Source ID and Valid Destination ID
- Test 2: Valid Source ID and Invalid Destination ID
- Test 3: Invalid Source ID and Valid Destination ID

Component	Nomenclature	Specifications
Wireless AP	Cisco Aironet 1242AG	802.11a/b/g
Laptop	HP Elitebook 8570w	Intel Core i7-3720QM CPU
		16 GB Ram & 64 bit
	Operating System	Ubuntu 14.04
	Software frameworks	LiClipse 2.2.0.2 (Python 2.7)
		AFIT Sniffer
		MBIDS.py
Software-Defined Radio	USRP N210	DC - 6GHz
	VERT900 antenna	sub-GHz 3dBi
Z-Wave Controller	RaZberry Pi	Z-Way version 2.0.0
Z-Wave devices	Aeon Labs smart energy plug	-
	Kwikset door lock	-
	Fortrezz water valve	-
	Everspring lamp module	-

(a) Target network components

Component	Nomenclature	Specifications
Laptop	HP Elitebook 8570w	Intel Core i7-3720QM CPU
		16 GB Ram & 64 bit
	Operating System	Ubuntu 14.04
	Software frameworks	LiClipse 2.2.0.2 (Python 2.7)
		EZ-Wave
		BurpProxy
		Zenmap
Software-Defined Radio	HackRF One	1 MHz - 6 GHz
	VERT900 antenna	sub-GHz 3dBi

(b) Attacker system components

Table 5. Experiment components include (a) target network components and (b) attacker system components.

In Tests 1 and 2, the Source ID is randomly chosen from the valid Node IDs present on the Z-Wave network (Figure 18(a)). In Test 1, the Destination ID is randomly

generated from within the pool of known valid Node IDs. Conversely, in Test 2, the Destination ID is randomly generated from a pool of all possible invalid Node IDs.

During Test 3, the Destination ID is randomly chosen from the list of valid Node IDs, while the Source ID is randomly generated from all possible invalid Node IDs (Figure 18(a)).

Three hundred trials are conducted for each test, where a trial consists of one generated packet sent to the Z-Wave network and captured by the MBIDS. Although it is possible to use a rouge controller to generate every packet, an SDR is used to simulate rogue controller transmissions by crafting packets identical to rogue controller packets and simply enumerate through the Source IDs and Destination IDs. The results of this experiment provide the mean detection accuracy of the MBIDS.

When a rogue controller is added to the network, it must be removed from the UI in order to evade user detection. As presented in Section 3.2, Gateway 1 allows for UI device removal without actual exclusion. Therefore, the rogue controller can be easily deleted from the UI on Gateway 1 while still maintaining access to the network. To increase the difficulty of rogue controller injection, it is recommended that gateway devices require device exclusion toward device removal from the UI. Without this security feature, rogue controller injection becomes evident to the user. With this feature, rogue devices are difficult to detect but the MBIDS is fully capable of detection.

4.8.2 Experiment 2: Manipulated Packet Injection Detection.

Experiment 2 consists of seven tests totaling 2,100 trials (Figure 18(b)). Unlike Experiment 1, not only does every possible combination of Source IDs and Destination IDs have to be accounted for, but also Packet Lengths, Command Classes, Commands,

and Payload parameters. The tests, therefore, ensure that candidates from all known possibilities are represented. Possibilities include:

- Test 1: All valid byte fields.
- Test 2: Invalid Packet Length and all other fields valid.
- Test 3: Invalid Source ID and all other fields valid.
- Test 4: Invalid Destination ID and all other fields valid.
- Test 5: Invalid Command Class and all other fields valid.
- Test 6: Invalid Command and all other fields valid.
- Test 7: Invalid Payload and all other fields valid.

Given the MBIDS software framework, a packet with invalid Source ID and Command Class with all other parameters valid is not a valid combination. After an invalid Source ID is checked, packet evaluation ceases and the misuse logs only records an invalid Source ID.

All valid fields are selected from the known-good signatures collected for the MBIDS. Invalid fields are randomly generated within accepted parameters for the respective field. Although a Command Class is represented by a specific byte (e.g., `0x32`) and is any byte from `0x00` to `0xFF`, if an attacker has requisite knowledge of the Z-Wave protocol before attacking, worse case, they will use a proper Z-Wave Command Class (Appendix A) to craft their packet. Therefore, Command Classes are randomly selected from the list of known Command Classes. Three hundred trials are conducted for each test, where a trial consists of one generated packet sent to the Z-Wave network and captured by the MBIDS. The results of this experiment provide the mean detection accuracy of the MBIDS.

4.8.3 Experiment 3: Increased Detection Rate Post Enhancement.

Before conducting Experiment 3, the MBIDS is enhanced as discussed in Section 4.2. Enhancement files include *Enhanced_MBIDS.py* and *Z-Way-Server_Comparator.py*. There are two enhancements. The first enhancement checks if a transmitted packet should have a previous state. If it should, the current state of the transmitted packet is compared with its required previous state. The second enhancement implements an evaluation framework for routed frames.

4.8.3.1 Enhancement 1: Stateful Protocol Analysis.

In the Z-Wave gateway, when a packet is sent by the controller (State 1), a log entry is created (State 2). To reduce incorrect packet classification, if a packet is received by the MBIDS where the Source ID is `0x01`, the packet is compared to the gateway log file to check if it was actually sent by the controller. Hence, if State 2 exists, State 1 must precede it. Since all primary gateway controllers have a default node ID `0x01`, if a packet is captured with Source ID `0x01`, it is from the controller. If the gateway log file does not contain a message sent by the controller that was captured by the MBIDS, the packet is malicious.

There is a slight delay from packet transmission to packet log; since Gateway 2 is used for this experiment, a ≈ 10 second delay is observed. The MBIDS logs all malicious packets for further review but also logs packets it receives and classifies as normal. If any packets that are classified as normal contain Source ID `0x01`, this enhancement strategy re-evaluates the packets after a < 10 second delay. This ensures that if the packets are in fact sent by the controller, every State 2 has a corresponding State 1. Conversely, if the packets are not sent by the controller, there are missing States 1 classifying the packets as misuse cases.

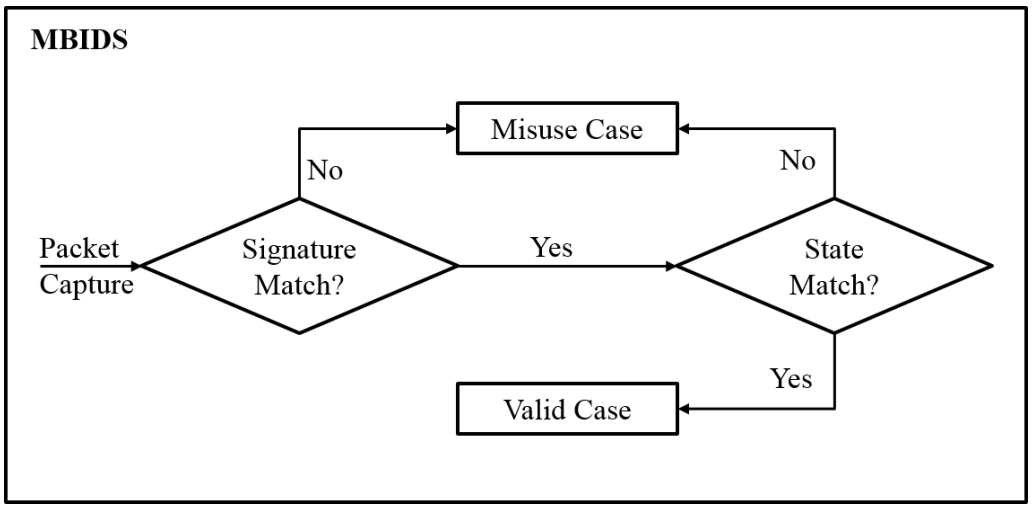


Figure 24. Signature-Based intrusion detection and stateful protocol analysis are subsets of an MBIDS. If any packet received is classified as a valid packet, the packet is re-evaluated using stateful protocol analysis to ensure validity.

Secondly, if a packet contains Source ID and Destination ID that are equal, the packet is a misuse case. In Local Area Network Denial (LAND) attack, the Source and Destination information in the TCP segment are the same. The target machine crashes or freezes because the packet is continually processed by the TCP stack. Although similar to a LAND attack, there are no effects of equal Source ID and Destination ID in Z-Wave networks. If a device with Node ID `0x05` receives a packet with Source ID `0x05`, it accepts the packet and executes the commands in the MSDU. To detect this attack, if a packet is classified as a valid packet, but contains equal Source ID and Destination ID, it is reclassified as a misuse case.

Figure 24 illustrates the logical topology of the MBIDS with the enhancement strategy. Valid packets are re-evaluated based on their state. If the state is invalid, the packet is reclassified as a misuse case.

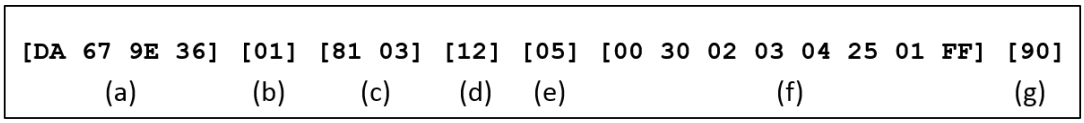
4.8.3.2 Enhancement 2: Routing Frame Evaluation.

When a routed Z-Wave frame is received by the MBIDS pre-enhancement, it is classified incorrectly. Before the routed frame evaluation is added to the MBIDS,

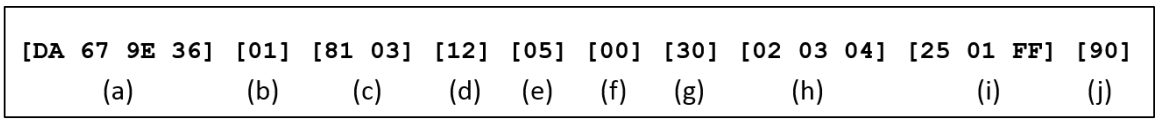
the routed frame is evaluated as seen in Sub-Figure 25(i). The (a) Home ID, (b) Source ID, (c) Frame Control, (d) Length, (e) Destination ID, and (g) Checksum are correctly evaluated. The Payload (f) is improperly evaluated. The first byte of the Payload `0x00` is considered the No Operation Command Class followed by variable parameters. However, `0x00` is actually the header for a routed payload.

Sub-Figure 25(ii) illustrates a proper routed frame evaluation provided by the enhancement strategy. As before, (a) through (e) and (j) are classified correctly. However, (f) is the routed frame header, (g) is split into two nibbles where nibble 1 is the hop count (3) and nibble 2 is the quantity of hops already completed (0). The next byte fields (h) consists of the nodes in the route that the packet goes through. When the packet gets to its final destination Node ID `0x05`, the hop count is 3, and (i) is executed as a Binary Switch Command Class `0x25`, with Command SET (`0x01`) and Payload Parameter ON (`0xFF`). The example mesh network is illustrated in Figure 26. The Z-Wave gateway (Node ID `0x01`) sends a command to destination Node ID `0x05` through nodes `0x02`, `0x03`, and `0x04`.

Given this new evaluation, the MBIDS can ensure the nodes in the route actually exist on the Z-Wave network by verifying group membership. The MBIDS also ensures that the destination node supports the Command Class sent.



(i) Pre-Enhanement Frame Classification



(ii) Post-Enhanement Frame Classification

Figure 25. Routed Frame pre and post enhancement classification.

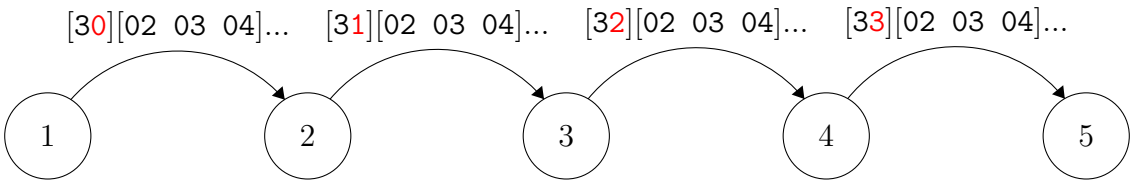


Figure 26. Z-Wave mesh topology routing. Node `0x01` sends a routed packet to Node `0x05`. The hop counter is steadily incremented until the final destination is reached.

After enhancement, the data collected from Experiments 1 and 2 are re-evaluated (Test 1 and Test 2, respectively) totaling 3,000 trials. Results of Tests 1 and 2 are compared with the results from Experiments 1 and 2 to calculate the post enhancement differences in detection rates.

4.9 Summary

The chapter discusses the methodology used to test the efficiency of the MBIDS under various workloads. Performance is evaluated using a real-world experimental design and is based on our performance metric: mean misuse detection rate. Three experiments are performed to measure the detection rate of the MBIDS and the impact of the enhancement on the overall system.

Experiments 1 and 2 use the *MBIDS.py* script in order to receive captured packets from the AFIT Sniffer and evaluate them for any malicious content. Malicious content includes invalid Packet Length, Source ID, Destination ID, Command Class, or Payload. When detected, the malicious content is logged for review by an investigator or system administrator.

Experiment 3 uses *Enhanced_MBIDS.py* to receive captured packets from the AFIT sniffer and evaluate them for any misuse cases. Any packets that are not misuse cases are re-evaluated using the *Z-Way-Server_Comparator.py*.

After the trials are complete for Experiments 1 and 2, the mean detection rate is calculated with a 99% confidence interval. After system enhancement, the data

from Experiments 1 and 2 are re-evaluated as Experiment 3. The third experiment measures the hypothesized increase in detection rate.

V. Results and Analysis

5.1 Introduction

This chapter presents and analyzes the results of the experiments. The results and analysis of Experiments 1, 2, and 3 are presented in Sections 5.2, 5.3, and 5.4, respectively. Lastly proposed in Section 5.5 is a list of defense-in-depth strategies enabling a user to harden the security of their insecure Z-Wave network.

5.2 Results and Analysis of Experiment 1

5.2.1 Test 1: Valid Source ID and Destination ID.

The first test performed on the system is used to determine the detection accuracy of the MBIDS at capturing rogue device packets. These packets include both valid Source IDs an Destination IDs (Figure 27). This test is considered the control/baseline test and establishes that the MBIDS is powered, responsive, and accepts valid packets, such as authorized network transmissions. Although a rogue device does not have a valid Source ID, as discussed in Section 3.2, packets with a valid Source ID are still injected to the Z-Wave network to ensure the MBIDS responds accurately.

Results of Test 1 are as anticipated. After 300 trials, the MBIDS detects zero violations in packets with valid Source ID and Destination ID. Since it is established the MBIDS responds correctly to packets containing both valid Source ID and Destination ID, subsequent tests are accomplished to evaluate remaining combinations.

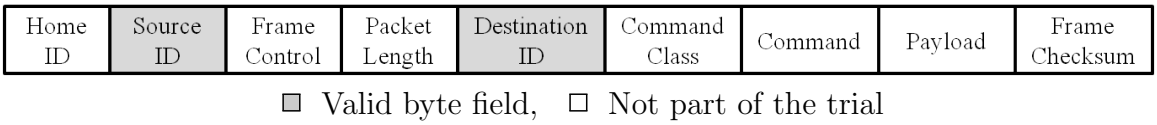


Figure 27. Experiment 1 - Test 1 packet structure.

5.2.2 Test 2: Valid Source ID and invalid Destination ID.

Test 2 provides the mean detection rate for injected packets containing valid Source ID and invalid Destination ID (Figure 28). When a rogue controller is added to a Z-Wave HAN, it receives a unique Node ID from the gateway and is capable of communicating with networked devices. This is possible because the rogue controller replicates the network receiving a list of Node IDs. Therefore, any commands sent from the rogue controller are sent to a valid node on the network, meaning the transmitted packet has a valid Destination ID. As a result, no packets generated by the rogue controller contain an invalid Destination ID. Although this packet with valid Source ID and invalid Destination ID is not likely concerning rogue controller packets, it represents an invalid group membership possibility that must be accounted for.

Results of Test 2 are also as anticipated. After 300 trials, the mean detection rate is 100% for all received packets with invalid Destination IDs.

5.2.3 Test 3: Invalid Source ID and valid Destination ID.

When two rogue controllers are used to conduct rogue controller injection, only one is actually added to the network and is excluded while the other rogue controller maintains access to the Z-Wave devices. Because of exclusion, the remaining rogue controller will have an invalid Source ID but still capable of communicating with nodes on the network because it has the correct Home ID. Therefore, every packet generated by the rogue controller will have an invalid Source ID and valid Destination ID (Figure 29).

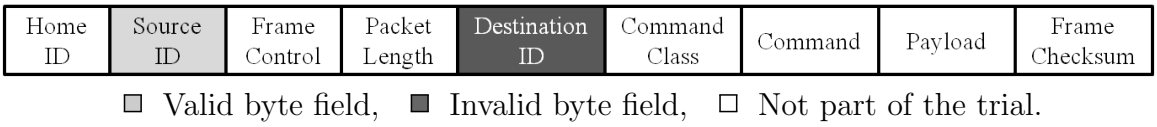


Figure 28. Experiment 1 - Test 2 packet structure.

After 300 trials, there is a mean detection rate of 100% for packets with invalid Source ID and valid Destination ID. This is the most realistic test for detecting rogue devices in Z-Wave networks. Given the results, an attacker is unable to use a rogue device without detection. Table 6 lists the total results for Experiment 1.

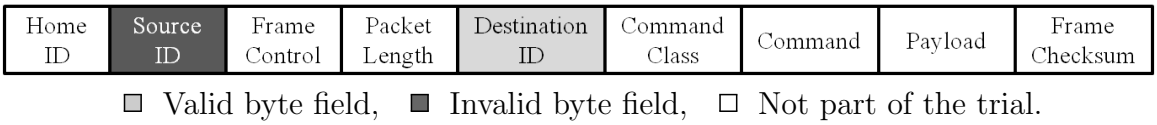


Figure 29. Experiment 1 - Test 3 packet structure.

Table 6. Results of Experiment 1. The detection rule corresponds to the byte fields tested.

Test	Trials	Detection Rule[1]	Detection Rate ($\bar{x}$)
1	300	`if (Source ID & Destination ID) == Valid:` `¬ misuse-case`	N/A[2]
2	300	`if Destination ID != Valid:` `LogFile += misuse-case`	100%
3	300	`if Source ID != Valid:` `LogFile += misuse-case`	100%
Overall $\bar{x}$	900	-	100%

[1]Conditional Constructs.
[2]Packets with valid fields are do not violate any misuse cases.

5.3 Results and Analysis of Experiment 2

5.3.1 Test 1: All Valid Byte Fields.

This test is similar to Experiment 1 Test 1. In this test, all six fields are valid in order to ensure the MBIDS does not classify the packet as a misuse case (Figure 30). In the attack scenario presented in Section 3.3 or similar attacks, an attacker injects a packet with valid parameters but also attempts to hide information. Since a manipulated injected packet will contain some misuse casec, this test does not demonstrate detection against a covert channel attack. However, it is necessary to evaluate the MBIDS for correctness. This test is therefore the first step in validation and verification of the system to ensure correct engineering.

After 300 trials, the results of this test demonstrate that the MBIDS evaluates all fields in the injected packet and determines them not to violate any misuse cases. Further validation and verification can occur to ensure that malicious packets are detected.

5.3.2 Test 2: Invalid Packet Length.

In Test 2, the injected packet contains an invalid Packet Length (Figure 31). As previously mentioned, in this case, an attacker is likely attempting to inject a packet padded with large amounts of data. The evaluation of packets sent at data rates R1 and R2 restricts the length to 64B. All injected packets are >64B length to test the detection accuracy of the MBIDS against packets that are too large.

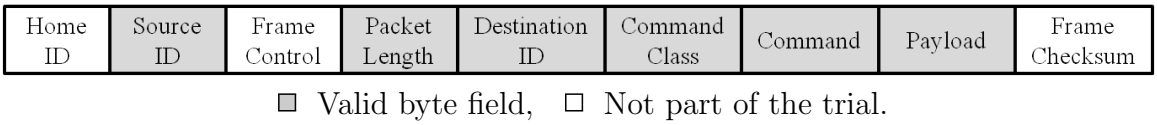


Figure 30. Experiment 2 - Test 1 packet structure.

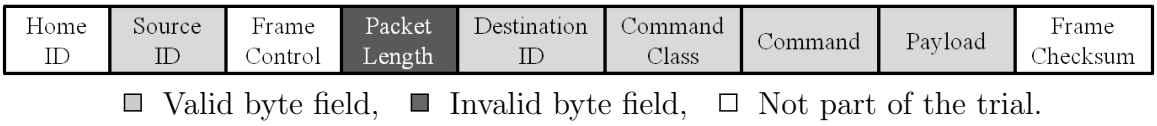
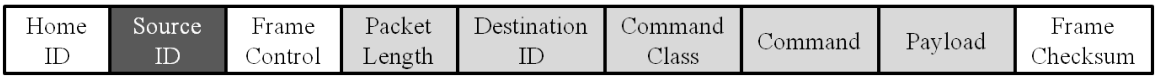


Figure 31. Experiment 2 - Test 2 packet structure.

Test 2 provides a mean detection rate of 100% for 300 injected packets containing an invalid Packet Length. When a packet is received, it is parsed and the length is extracted and compared to the maximum allowable length. The attacker can attempt to obviate this measure by changing the length in a crafted packet to an allowable value. However, when a Z-Wave transceiver receives a packet, it checks the length and reads the quantity of bytes corresponding to the length. If the length is changed to a value allowable but does not represent the actual length of the packet, the Z-Wave transceiver does not read the entire packet and the last byte is considered the checksum. Since it is not the checksum, it is considered an invalid checksum and the packet is dropped. The length of any injected packet must represent the actual length of the packet in order to be accepted by a Z-Wave transceiver, but if it is an accurate length and greater than the maximum allowable length, it is classified as a misuse case by the MBIDS.

5.3.3 Test 3 and 4: Invalid Source ID or Invalid Destination ID.

Tests 3 and 4 provide the mean detection rate for injected packets containing an invalid Source ID or Destination ID (Figure 32). Similar to Tests 2 and 3 in Experiment 1, 300 packets are injected for each invalid field. Results for each test are as expected. The MBIDS achieves a 100% mean detection rate for each test.



(a) Test 3

(b) Test 4

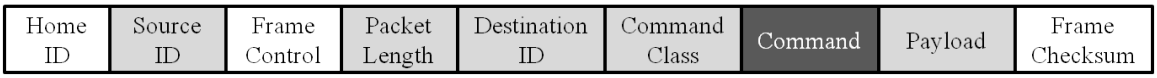
□ Valid byte field, ■ Invalid byte field, □ Not part of the trial.

Figure 32. Experiment 2 - Test 3 and 4 packet structure.

5.3.4 Test 5 and 6: Invalid Command Class or Invalid Command.

After an attacker gains access to a Z-Wave HAN, as illustrated in Section 3.2, they may discover valid Node IDs on the network. However, without knowledge of the exact types of devices, an attacker is unaware of the types of Command Classes that each device supports. Without this knowledge, an attacker crafts a packet with an invalid Command Class (Figure 33a). If the packet is sent to a Node ID that represents a slave node on the network, the device accepts the packet, does not perform the Command Class function, and responds with an acknowledgment. However, if an attacker sends the packet to the gateway, the gateway responds similarly, but also stores the communication messages in its log file. This is precisely how the Hel Attack is accomplished. Therefore, the Command Class is checked for validity.

(a) Test 5



(b) Test 6

□ Valid byte field, ■ Invalid byte field, □ Not part of the trial.

Figure 33. Experiment 2 - Test 5 and 6 packet structure.

The same applies to a Command (Figure 33(b)). Without prior knowledge of a specific device on the Z-Wave HAN, not only will the attacker not know the Command Class but also the corresponding Command.

After 300 trials, the MBIDS has a mean detection rate of 100% at detecting injected packets with unsupported Command Classes. The MBIDS also has a mean detection rate of 100% at detecting 300 injected packets with invalid Commands.

5.3.5 Test 7: Invalid Payload.

The maximum size of the MSDU is 54B (Section 2.2). After the Command Class and Command are chosen, 52B remain. The Payload is randomly generated between length 1B to 52B and concatenated to the Command Class and Command before packet injection. This ensures a valid Packet Length but the Payload contains random invalid byte manipulations (Figure 34).

After 300 trials, the MBIDS results in a 91.7% mean detection rate for packets with manipulated payloads. When packets are received and evaluated, the misuses are logged but the packets that are not considered misuse cases are also logged. This allows a for review and analytics of why some manipulated injected packets are not flagged as a misuse case.

Further analysis revealed that of the 11 implemented Command Classes: Configuration, Association, Version, and Security (Appendix A) Command Classes have too much variability to account for all possible combinations. Therefore, in this proof of concept, it is demonstrated that although the MBIDS cannot detect all packet

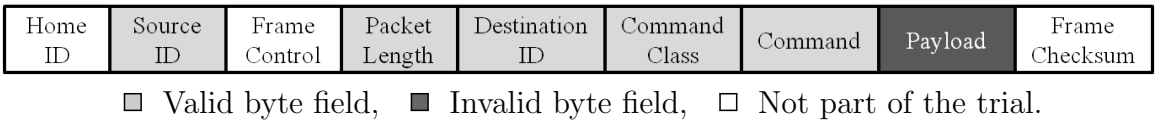


Figure 34. Experiment 2 - Test 7 packet structure.

manipulation attacks with 100% accuracy given the variability of payload parameters, given detailed proprietary information, the MBIDS can be fully implemented to achieve 100% accuracy.

Table 7 lists a 100% mean detection rate of invalid Packet Length, Source ID, Destination ID, Command Class, and Command. All validation checks in the MBIDS are deterministic and are sure to detect violations. As an analogy, a door lock manufacturer engineers a door lock to only accept one cut of key. Any other cuts used should not work. The manufacturer is sure of the design but still tests nonstandard possibilities to ensure the door lock operates as intended. Similarly, unsupported values are tested knowing the outcome but need to ensure *MBIDS.py* is engineered correctly.

Table 7. Results of Experiment 2. The detection rule corresponds to the byte fields tested.

Test	Trials	Detection Rule[1]	Detection Rate ($\bar{x}$)
1	300	`if (All Byte Fields) == Valid:` ¬ `misuse-case`	N/A[2]
2	300	`if Packet Length != Valid:` `LogFile += misuse-case`	100%
3	300	`if Source ID != Valid:` `LogFile += misuse-case`	100%
4	300	`if Destination ID != Valid:` `LogFile += misuse-case`	100%
5	300	`if Command Class != Valid:` `LogFile += misuse-case`	100%
6	300	`if Command != Valid:` `LogFile += misuse-case`	100%
7	300	`if Payload != Valid:` `LogFile += misuse-case`	$[88.5 - 94.8\%]$
Overall $\bar{x}$	2,100	-	$[98.3 - 99.2\%]$

[1]Conditional Constructs.
[2]Packets with valid fields are do not violate any misuse cases.

If the attacker is not aware of the Source ID, Destination ID, valid Packet Length, Command Class, or Command, they are likely to inject a packet with nonstandard values and will certainly be detected and logged as a misuse case.

Of the six workload parameters tested, only one does not result in 100% detection (Table 7). Packets with invalid Payloads are detected with a mean of 91.7%. Conducting 2,100 tests of independent packet transmissions with invalid Packet Length, Source ID, Destination ID, Command Class, Command, and Payload results in an overall accuracy of 98.8% mean misuse detection rate.

5.4 Results and Analysis of Experiment 3

5.4.1 Test 1: Re-Evaluation of Experiment 1.

Both enhancement strategies are targeted at re-evaluating packets that are improperly classified as having all valid byte fields. However, given the mean rogue device detection rate is 100% pre-enhancement, Experiment 1 is only re-evaluated to ensure the enhancement strategies proposed in *Enhanced_MBIDS.py* and *Z-Way-Server_Comparator.py* are engineered correctly and do not affect previous results.

Results are consistent with those in Table 6, confirming the MBIDS is capable of detecting rogue device attacks in Z-Wave networks with 100% mean detection rate.

5.4.2 Test 2: Re-Evaluation of Experiment 2.

Similarly to Test 1, the enhancement strategies target previously evaluated packets that are incorrectly classified as normal, specifically the Payload. Therefore, all results after re-evaluation are consistent with those in Table 7 with the exception of the mean misuse detection rate for invalid Payloads.

After re-evaluation, the mean invalid Payload detection rate increases to 95.7%, a 4% improvement. Of the 300 trials, *Enhanced_MBIDS.py* evaluates 25 packets as

valid even though all are invalid. If any of the 25 packets have Source ID `0x01` or Source ID equal to Destination ID, the packet is logged in a secondary logfile for re-evaluation. The secondary logfile is scanned by *Z-Way-Server_Comparator.py*. Of the 12 packets that are logged in the secondary logfile, *Z-Way-Server_Comparator.py* classifies all of them as having either equal Source ID and Destination ID or a packet with Source ID `0x01` that was actually sent by the Z-Wave gateway controller. The improved results of Experiment 3 Test 2 are listed in Table 8.

5.5 Defense-in-depth Strategies

An MBIDS is a effective means of detecting attacks in Z-Wave networks. However, because of the insecurity of many gateway devices, users must consider defense in depth to secure any means by which gateways are accessed. As discussed in Section 2.4, there are three ways to gain access to a gateway device. Of the three ways, compromise of the WLAN backbone is the most common, since an attacker can remain inconspicuous while attacking the target network. As the most viable means of entry, the WLAN provides a single point of access. Therefore, end users must ensure the WLAN backbone is secure. This sections proposes defense-in-depth strategies that aides users in securing their Z-Wave HANs.

5.5.1 Hide WLAN SSID.

The first step in exploiting the HAN is gaining access to the target WLAN. Hiding the service set identifier (SSID) slows attackers using passive network scans to locate the WLAN. This option can be configured in the wireless access point settings. If an attacker conducts Z-Wave warwalking and locates a Z-Wave network, but is not able to identify the SSID, the attacker is hindered from authenticating to the WLAN and attack the globally accessible Z-Wave gateway.

5.5.2 WLAN security with a robust password.

Some WLANs are left unsecured, granting attackers uninhibited access. Secondly, some WLANs use WEP to secure their network because they require backward com-

Table 8. Results of Experiment 3 - Test 2. The detection rule corresponds to byte fields tested.

Test	Trials	Detection Rule[1]	Detection Rate ($\bar{x}$)
1	300	`if (All Byte Fields) == Valid:` `¬ misuse-case`	N/A[2]
2	300	`if Packet Length != Valid:` `LogFile += misuse-case`	100%
3	300	`if Source ID != Valid:` `LogFile += misuse-case`	100%
4	300	`if Destination ID != Valid:` `LogFile += misuse-case`	100%
5	300	`if Command Class != Valid:` `LogFile += misuse-case`	100%
6	300	`if Command != Valid:` `LogFile += misuse-case`	100%

7	300	*Enhanced_MBIDS.py*
		`if Payload != Valid:`
		`LogFile += misuse-case`
		`else:`
		`SecondaryLog += valid-case`
		Z-Way-Server_Comparator.py
		`if Source ID == Destination:`
		`LogFile += misuse-case`
		`else if Source ID == 0x01`
		`& Packet ∉ GatewayLog:`
		`LogFile += misuse-case`
			$[93.3 - 97.9\%]$

Overall $\bar{x}$	2,100	-	$[99.0 - 99.7\%]$

[1]Conditional Constructs.
[2]Packets with valid fields are do not violate any misuse cases.

patibility with legacy devices or the user is not aware of existing vulnerabilities. If possible, WPA2 should be used since WEP is easily exploitable. A 128-bit WEP key can be cracked in just three minutes with freely available tools such as *aircrack-ng* [RSRR+10]. Weak passwords should be avoided because numerous free dictionaries containing precomputed passwords can be found on the Internet. An attacker armed with an Alfa AWUS036H Card, *aircrack-ng*, and a large password dictionary can crack weak WPA2 passwords. Although longer and more complex passwords are breakable, they take significantly longer to crack than weak passwords. A simple calculation can be used to estimate the maximum time required to brute force crack passwords (3).

$$timetaken(seconds) = \frac{combinations}{guesses/second} \tag{1}$$

Using case sensitive passwords consisting of 94 available characters, Table 9 lists the maximum cracking times, assuming the computer calculates a modest 5,000,000 guesses per second. Password dictionaries can significantly reduce the time taken to crack passwords, so users should choose passwords that are as long and complex as practicable.

5.5.3 MAC address filtering.

MAC address filtering allows the administrator to select specific devices that are trusted on the WLAN. Although the attacker can spoof MAC addresses, this is an additional barrier for WLAN defense in depth. This can also be configured in the wireless access point settings. If MAC address filtering is enabled, an attacker without knowledge of spoofing MAC addresses will not be able to authenticate to the WLAN, even if armed with the WLAN password.

5.5.4 Enable UI authentication on the Z-Wave gateway.

Not all Z-Wave gateways have UI authentication. However, if UI authentication is available, the user should enable it and create a strong password. Many of the vulnerabilities exploited in [CBS13] do not work on the recently released Almond+. With mandatory UI authentication, an attacker would have to intercept HTTP packets while the user is logging in to capture credentials since some gateways send passwords in the clear. UI authentication is clearly not enough, but it encumbers an attacker. Even though it is possible to capture the Almond+ UI authentication credentials, the exploits available are limited because the Almond+ architecture provides extra security. Not being able to put the Almond+ into inclusion mode from the UI prevents the injection of a rogue controller into the network.

5.5.5 Use a Reverse Proxy Server.

A Reverse Proxy Server (RPS) is an intermediary application between the user and the gateway. It provides an added layer of security by intercepting packets sent to the gateway and performs additional authentication before allowing the user to access

Table 9. Brute Force Password Cracking [FR15a].

Characters	Combinations	Time taken @ 5×10^6 guesses/sec
1	94	0.00002 seconds
2	8,836	0.00177 seconds
	830,534	0.16612 seconds
4	78,074,896	15.6145 seconds
5	7.3×10^9	25 minutes
6	6.9×10^{11}	38 hours
7	6.4×10^{13}	150 days
8	6×10^{15}	38 years

the gateway. This is a effective option for gateways that do not have UI authentication and provides increased UI security for those that already have authentication. RPS tools such as *NGINX* are free and available for download.

5.5.6 Disable unused network services.

Gateway administrators should also ensure that any unused network services are disabled. If there is not a need to have continual access to the gateway backend, SSH should be disabled when not in use. SSH is enabled by default on the VeraEdge Home Controller. As mentioned in Section 3.2.2, after downloading the backup file and locating the WLAN password, the attacker successfully logs into the backend of the VeraEdge using *Putty* with username *root* and password <*wlan_password*>. This vulnerability is avoided if SSH is disabled.

5.5.7 Enable end-device encryption.

When a rogue controller is added to the network, it can control most devices except those that use encryption. When a device that uses encryption is included into the Z-Wave network, the device sends its encryption key to the gateway. A chain of trust is then established between those two devices. When a rogue controller is added, the chain of trust does not extend to the rogue controller because the end device is not aware of its inclusion. Some Z-Wave devices, including Gen5 models, come with the option to enable encryption with the press of a button. Enabling end device encryption may mitigate rogue controller attacks.

5.5.8 Inspect log files.

Although it is possible for an attacker to clear log files to cover their tracks after compromising the Z-Wave HAN, it is still worth inspecting the log files for unusual

activity. Any actuation of devices at abnormal hours and failed login attempts is recorded in the log. If this is found, the user should reset their gateway and change their passwords.

Applying these mitigation strategies will not only deter an attacker but decrease the attack surface, lessening the chance of a successful Z-Wave HAN exploitation.

5.6 Summary

This chapter presents and analyzes the data collected from each of the experiments. A statistical analysis of the performance metrics in each experiment is performed. Then, a list of defense-in-depth strategies to aide in defending a Z-Wave network is provided. The results show that the enhancement strategy increases the mean misuse detection rate of packets with invalid Payloads to 95.7%. Therefore, the mean detecting accuracy of the MBIDS after enhancement is 99.4% for all byte fields tested.

VI. Conclusion

6.1 Introduction

This chapter presents the overall conclusions drawn from the research. Section 6.2 compares each research goal with the experiment results and determines if the research objectives have been met. The significance of this research is discussed in Section 6.3. Section 6.4 provides several recommendations for future work. Finally, Section 6.5 provides the source code used in this research.

6.2 Conclusions of Research

6.2.1 Goal 1: Misuse Case Identification.

The first goal of this research is to identify misuse cases in order to develop an MBIDS. A Z-Wave-capable SDR is used to capture packets. The packets are dissected and each byte field is evaluated using the ITU-T G.9959 recommendation [ITU15] and open source documentation including OpenZWave [Ope14] and the RaZberry Pi documentation [RaZ15]. As a proprietary protocol, discovering all misuse cases is possible given proprietary information. Therefore, misuse cases are determined for 11 of the known Z-Wave Command Classes and their corresponding Payload parameters. This allows the engineering of conditional constructs used for packet evaluation. The first goal is accomplished toward achievement of subsequent goals.

6.2.2 Goal 2: Rogue Device Detection.

The second goal of this research is to engineer a system, an MBIDS, that captures and analyzes Z-Wave transmissions against a lists of signatures and states determining whether the captured packets are valid. The MBIDS is then evaluated based on the

ability to detect, and efficiency of detecting rogue device attacks in Z-Wave networks. To accomplish this goal, the AFIT Sniffer [BFH+15] is used to capture packets. The packets are read in by the MBIDS and evaluated for violation. By dissecting the packets and comparing the Source ID and Destination ID with a list of known Node IDs retrieved from the Z-Wave gateway, the MBIDS is able to detect rogue device attacks with a 100% mean detection rate accomplishing the second research goal.

6.2.3 Goal 3: Manipulated Injected Packet Detection.

The third goal of this research is to determine the efficiency of the MBIDS at detecting manipulated injected Z-Wave packets. Using an SDR as an attacker, numerous manipulated packets are injected into the Z-Wave network containing invalid Packet Length, Source ID, Destination ID, Command Class, Command, or Payload. Of the six byte fields modified to be invalid, all but one results in a mean detection rate of 100%. The packets with invalid Payloads are detected on average 91.7%. The overall efficiency of the MBIDS given the mean detection rate of 100% for five of the evaluated fields and a 91.7% mean detection rate for one evaluated field results in an overall mean detection rate of 98.8% The results confirm the hypothesis and accomplishing the third research goal.

6.2.4 Goal 4: Enhanced MBIDS.

The final goal of this research is to enhance the MBIDS to increase the misuse detection rate. The mesh routing protocol is reverse engineered, allowing the MBIDS to properly evaluate routed frames. Secondly, the Z-Wave protocol allows for devices to accept packets with an equal Source ID and Destination ID, although this is only present during packet injection attacks. Therefore, the MBIDS is enhanced to check for this condition and classify it as a misuse case. Lastly, when a packet that is

a misuse case is improperly classified, if the Source ID is `0x01`, the packet is re-evaluated. If it is not in the Z-Wave gateway logfile, it is reclassified as a misuse case.

Results of Experiment 1 cannot be improved, so the enhancement strategies are solely tested against rogue device attacks. Results of rogue device attack detection using the enhancement strategies are consistent with the results pre-enhancement.

To evaluate the enhancement strategies at detecting manipulated injected packet attacks the six byte fields are modified to be invalid. Injected packets with invalid byte fields result in a mean detection rate of 100% for five of the byte fields (Packet Length, Source ID, Destination ID, Command Class, and Command) and 95.7% mean detection rate for packets with invalid Payloads. Therefore, the overall efficiency of the MBIDS is a mean detection rate of 99% accomplishing the final research goal.

6.3 Significance of Research

The significance of this research can be approached in two ways. First, a discussion of the novel attacks presented herein. The second is to discuss the intrinsic value of the MBIDS as a monitoring tool for Z-Wave networks. When fully extended to include support for all Command Classes, the MBIDS is a tool fully capable of monitoring Z-Wave networks.

6.3.1 Z-Wave Network Attacks.

The newly discovered attacks presented herein illustrate the numerous insecurities in Z-Wave gateways and the underlying protocol. The ability to inject rogue devices into a Z-Wave network, especially controller devices, poses a serious threat to Z-Wave network users. Given that there are no publicly available sub-GHz traffic analyzers

for Z-Wave networks, an attacker using a rogue device in a Z-Wave network cannot be detected while communicating in the Z-Wave RF spectrum.

Based on the ITU-T G.9959 specification, the Z-Wave protocol ensures packets are no more than 64B while operating at 9.6 kbps or 40 kpbs data rates and 170B while operating at data rate 100 kbps. However, an attacker can craft a packet that is smaller than the allowable bounds and pad extra bytes until the threshold is reached. This provides a covert channel in Z-Wave packets to hide information. Using this covert channel and Z-Wave gateways insecurities, the Hel Attack is created whereby the attacker injects a packet in the Z-Wave network with hidden information that triggers a R-SSH to any Internet connected computer the attacker chooses. This backdoor into the gateway not only provides Z-Wave network control, but allows allows access to other WLAN connected devices. Motivated by these attacks, the MIBDS is developed to detect rogue device attacks, covert channel attacks, and other packet injection attack in Z-Wave networks.

6.3.2 MBIDS Employed.

This research provides the DoD and other government agencies with an effective method to detect Z-Wave network attacks. This system is the first of its kind and demonstrates a proof of concept approach to detecting rogue device attacks and manipulated packet injection attacks. Enhancements to the system include detection of known-good packets from a spoofed controller. By designing the system to operate on any computer that supports Python 2.7, it can be easily implemented using any SDR configured for Z-Wave packet capture. The determinism of the MBIDS enables it to run at high speeds, ensuring a high probability of successful packet classification even when monitoring a heavily utilized Z-Wave network. Because the MBIDS works

in parallel with the Z-Wave network (Figure 35), any failure has no negative effects on network performance.

The MBIDS supports 11 of the known command classes. When fully implemented, the MBIDS is an effective monitoring and detection tool against Z-Wave network attacks. By validating group membership and evaluating byte fields in Z-Wave frames based on signatures and states, the MBIDS can determine if packets are malicious.

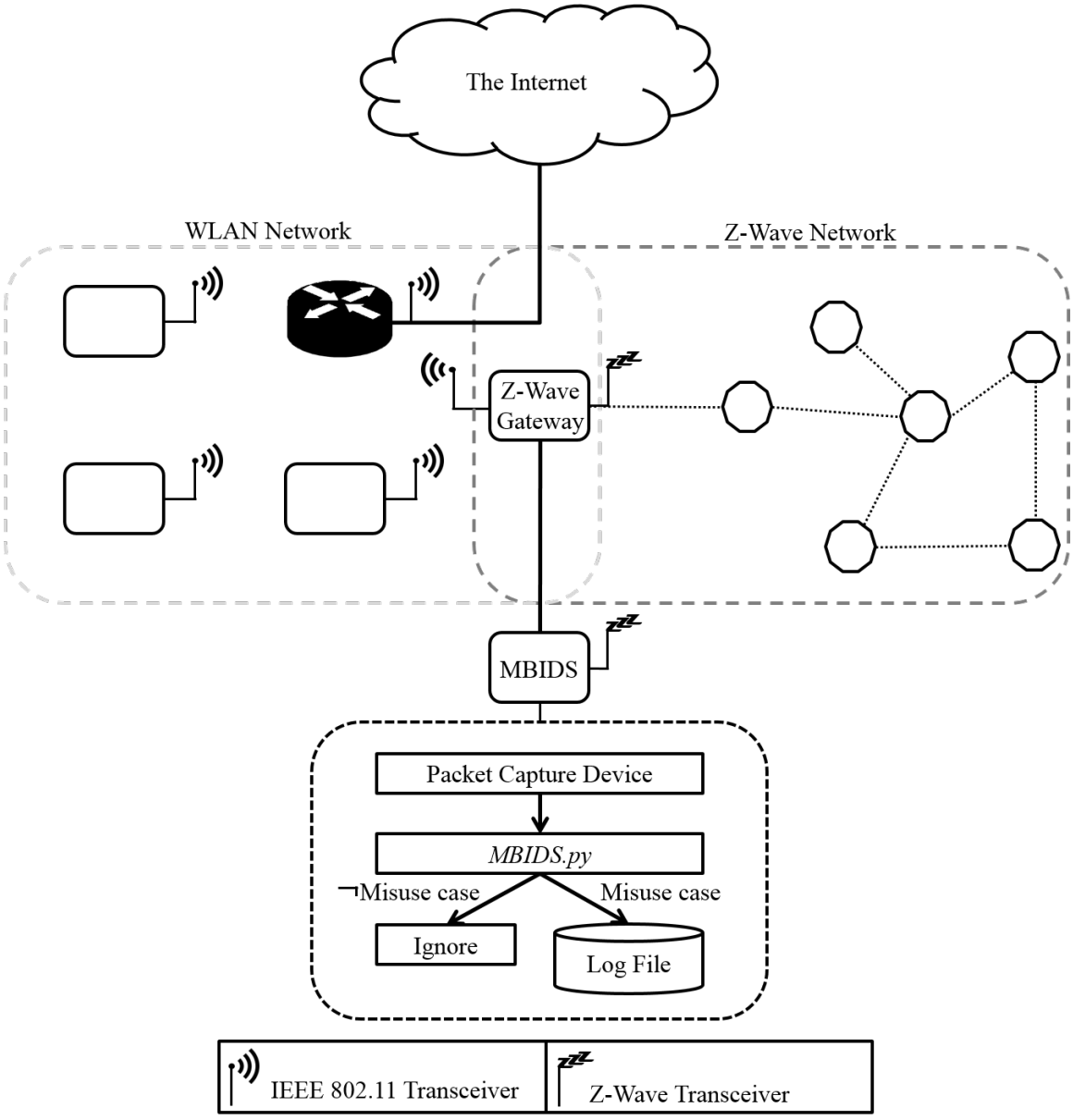


Figure 35. Z-Wave Misuse-Based Intrusion Detection System

With this binary evaluation, 1 for misuse case and 0 for valid case, an investigator or system administrator can determine malicious packets and the intent of an attacker by evaluating logged frames.

Furthermore, the MBIDS can be modified to detect packet injection attacks discussed in Section 2.5. To perform the attack illustrated in [FG13], an attacker needs detailed knowledge of the key exchange protocol. Then, using Z-Force, the attacker can force the target Z-Wave door lock to overwrite the current encryption key. The attacker can now inject messages to the door lock to modify its status. However, if the attacker does not use a valid Source ID or uses Source ID `0x01`, the attack is detected. The same applies to the Scapy-Radio tool [PLD14]. If the Source ID violates the MBIDS group membership policy, it is classified as a misuse case and detected.

The MBIDS provides security in the lacking ITU-T G.9959-based networks. The lack of security and indefensibility motivates the need for a system capable of monitoring network activity. As an illustration, a vignette is provided in Appendix B discussing the use of WSNs in a deployed combat environment and the security implications of its use.

6.4 Recommendations for Future Research

The next logical step for this research is to include the evaluation of remaining Command Classes and their corresponding Payload parameters. However, since Z-Wave is a proprietary protocol, proprietary information is needed toward implementation of all Command Classes.

Secondly, tuning the SDR to receive packets with varying preamble lengths can increase the MBIDS detection accuracy of other types of packet injection attacks. The EZ-Wave tool provides an attacker with more capabilities than Z-Force or Scapy-

Radio [HRRL16]. Device fingerprinting through preamble manipulation allows an attacker to perform network reconnaissance and enumeration. With this information, the attacker can target specific devices on the network. However, the MBIDS packet capture device can be tuned to accept packets with varying preamble lengths. Specifically, [HRRL16] shows that Z-Wave devices will respond to packet with 2B or greater preambles. If the MBIDS SDR is tuned to receive packets with preamble lengths of at least 2B and any field in the packets used for device fingerprinting is invalid, EZ-Wave attacks will be detected.

Another area of future research is using the MBIDS as a framework. Given the feasibility of an MBIDS, vendors that are partnered with Sigma Designs and receive Z-Wave SDKs have knowledge of the proprietary protocol and are free to add their interoperable modifications. Vendors develop devices and know what `CmdCls`, `Cmds`, and `Pld` (parameters) each device supports. Using the MBIDS deterministic byte field evaluations, vendors can integrate a detection and prevention system in each device to allow or reject frames based on the byte fields. This will not only provide a detection mechanism, but given the ability to evaluate received packets, devices will be able to discard misuse cases allowing for a Z-Wave network capable of detecting and preventing attacks.

Future work should focus on evaluating the effectiveness of a hardware-based misuse detection system. A Z-Wave device can be implemented using an SDR and computer peripheral to mimic normal device operation. Integrating the MBIDS on the hardware peripheral as a proof of concept further illustrates the applicability of this approach at defending Z-Wave networks.

Finally, this research examined the Z-Wave protocol alone. The impact that the attacks presented herein and the detection method warrants further study on other

ITU-T G.9959-based protocols and the efficacy of this approach can be extended to other WSN protocols.

6.5 Supporting Documentation

All source code for this research is written in Python 2.7. The Hel Attack (Section 3.3) involves triggering malware on an infected Z-Wave gateway to open a R-SSH to any Internet Connected computer the attacker chooses. Therefore, a Python server script and client script are needed for backdoor communication, namely *Target_SSH.py* and *Server_SSH.py*, respectively. Experiments 1 and 2 use *MBIDS.py* and its associated custom imported files. Experiment 3 uses *Enhanced_MBIDS.py* and *Z-Way-Server_Comparator.py* and their associated custom imported files.

All source files can be retrieved from `https://github.com/AFITWiSec/ZIDS`.

Appendix A. Z-Wave Command Classes

This comprehensive list of known Command Classes is compiled using the Open-ZWave Library [Ope14], RaZberry Pi Z-Way Developer's Documentation [RaZ15], and SmartThings API [Sma15] (account required). A * denotes a Command Class that is implemented in the MBIDS.

Table 10. Z-Wave Command Classes

Name	Hex	Commands
NoOperation*	0x00	N/A
CallNIF*	0x01	Get (0x02)
Basic*	0x20	Set (0x01)
		Get (0x02)
		Report (0x03)
ControllerReplication	0x21	TransferGroup (0x31)
		TransferGroupName (0x32)
		TransferScene (0x33)
		TransferSceneName (0x34)
ApplicationStatus	0x22	Busy (0x01)
		RejectedRequest (0x02)
Zip	0x23	Packet (Unk)
SwitchBinary*	0x25	Set (0x01)
		Get (0x02)
		Report (0x03)
SwitchMultilevel	0x26	Set (0x01)
		Get (0x02)
		Report (0x03)
		StartLevelChange (0x04)
		StopLevelChange (0x05)
		SupportedGet (0x06)
		SupportedReport (0x07)
SwitchAll	0x27	Set (0x01)
		Get (0x02)
		Report (0x03)
		On (0x04)
		Off (0x05)

SwitchToggleBinary	0x28	Set (0x01)
		Get (0x02)
		Report (0x03)
SwitchToggleMultilevel	0x29	Set (0x01)
		Get (0x02)
		Report (0x03)
		StartLevelChange (0x04)
		StopLevelChange (0x05)
SceneActivation	0x2B	Set (0x01)
SceneActuatorConf*	0x2C	Set (0x01)
		Get (0x02)
		Report (0x03)
SceneControllerConf	0x2D	Set (Unk)
		Get (Unk)
		Report (Unk)
SensorBinary	0x30	Get (0x02)
		Report (0x03)
SensorMultilevel	0x31	SupportedGet (0x01)
		SupportedReport (0x02)
		Get (0x04)
		Report (0x05)
Meter	0x32	Get (0x01)
		Report (0x02)
		SupportedGet (0x03)
		SupportedReport (0x04)
		Reset (0x05)
Color	0x33	CapabilityGet (0x01)
		CapabilityReport (0x02)
		Get (0x03)
		Report (0x04)
		Set (0x05)
		StartLevelCapabilityChange (0x06)
		StopStateChange (0x07)
MeterPulse	0x35	Get (0x04)
		Report (0x05)

MeterTableMonitor	0x3D	GetAdminId (Unk)
		GetID (Unk)
		StatusDepthGet (Unk)
		StatusDateGet (Unk)
		CurrentDataGet (Unk)
		HistoricalDataGet (Unk)
ThermostatMode	0x40	Set (0x01)
		Get (0x02)
		Report (0x03)
		SupportedGet (0x04)
		SupportedReport (0x05)
ThermostatOperatingState	0x42	Get (0x02)
		Report (0x03)
ThermostatSetPoint	0x43	Set (0x01)
		Get (0x02)
		Report (0x03)
		SupportedGet (0x04)
		SupportedReport (0x05)
ThermostatFanMode	0x44	Set (0x01)
		Get (0x02)
		Report (0x03)
		SupportedGet (0x04)
		SupportedReport (0x05)
ThermostatFanState	0x45	Get (0x02)
		Report (0x03)
ClimateControlSchedule	0x46	Set (0x01)
		Get (0x02)
		Report (0x03)
		ChangedGet (0x04)
		ChangedReport (0x05)
		OverrideSet (0x06)
		OverrideGet (0x07)
		OverrideReport (0x08)

DoorLockLogging	0x4C	RecordSupportedGet (0x01)
		RecordSupportedReport (0x02)
		RecordGet (0x03)
		RecordReport (0x04)
ScheduleEntryLock	0x4E	Enable (Unk)
		WeekdayGet (Unk)
		WeekdatReport (Unk)
		YearGet (Unk)
		YearReport (Unk)
6lowpan	0x4F	FirstFragment (Unk)
		SubsequentFragment (Unk)
BasicWindowCovering	0x50	StartLevelChange (0x01)
		StopLevelChange (0x02)
TransportService	0x55	FirstFragment (Unk)
		FragmentComplete (Unk)
		FragmentRequest (Unk)
		FragmentWait (Unk)
		SubsequentFragment (Unk)
CRC16Encap*	0x56	Encap (0x01)
DeviceResetLocally	0x5A	Notification (0x01)
CentralScene	0x5B	CapabilityGet (0x01)
		CapabilityReport (0x02)
		Set (0x03)
ZWavePlusInfo	0x5E	Get (0x01)
		Report (0x02)
MultiChannel	0x60	CapabilityGet (Unk)
		CapabilityReport (Unk)
		CmdEncap (Unk)
		EndPointFind (Unk)
		EndPointFindReport (Unk)
		EndPointGet (Unk)
		EndPointReport (Unk)

DoorLock	0x62	Set (0x01)
		Get (0x02)
		Report (0x03)
		ConfigurationSet (0x04)
		ConfigurationGet (0x05)
		ConfigurationReport (0x06)
UserCode	0x63	Set (0x01)
		Get (0x02)
		Report (0x03)
		UserNumberGet (0x04)
		UserNumberReport (0x05)
Configuration*	0x70	Set (0x04)
		Get (0x05)
		Report (0x06)
Alarm	0x71	Get (0x04)
		Report (0x05)
		SupportedGet (0x07)
		SupportedReport (0x08)
ManufacturerSpecific	0x72	Get (0x04)
		Report (0x05)
PowerLevel	0x73	Set (0x01)
		Get (0x02)
		Report (0x03)
		TestNodeSet (0x04)
		TestNodeGet (0x05)
		TestNodeReport (0x06)
Protection*	0x75	Set (0x01)
		Get (0x02)
		Report (0x03)
Lock	0x76	Set (0x01)
		Get (0x02)
		Report (0x03)

NodeNaming	0x77	Set (0x01)
		Get (0x02)
		Report (0x03)
		LocationSet (0x04)
		LocationGet (0x05)
		LocationReport (0x06)
FirmwareUpdateMd	0x7A	Get (Unk)
		Report (Unk)
		MdGet (Unk)
		MdReport (Unk)
		RequestGet (Unk)
		RequestReport (Unk)
		StatusReport (Unk)
GroupNaming	0x7B	Set (Unk)
		Get (Unk)
		Report (Unk)
RemoteAssociationActivate	0x7C	Activate (Unk)
RemoteAssociation	0x7D	Set (Unk)
		Get (Unk)
		Report (Unk)
Battery	0x80	Get (0x02)
		Report (0x03)
Clock	0x81	Set (0x04)
		Get (0x05)
		Report (0x06)
Hail	0x82	Hail (0x01)
WakeUp	0x84	IntervalSet (0x04)
		IntervalGet (0x05)
		IntervalReport (0x06)
		Notification (0x07)
		NoMoreInformation (0x08)
		CapabilitiesGet (0x09)
		CapabilitiesReport (0x0A)

Association*	0x85	Set (0x01)
		Get (0x02)
		Report (0x03)
		Remove (0x04)
		GroupingGet (0x05)
		GroupingsReport (0x06)
Version*	0x86	Get (0x11)
		Report (0x12)
		CommandClassGet (0x13)
		CommandClassReport (0x14)
Indicator	0x87	Set (0x01)
		Get (0x02)
		Report (0x03)
Propietary	0x88	Set (0x01)
		Get (0x02)
		Report (0x03)
Language	0x89	Set (0x01)
		Get (0x02)
		Report (0x03)
Time	0x8A	DateGet (Unk)
		DateReport (Unk)
		TimeGet (Unk)
		TimeReport (Unk)
		TimeOffsetGet (Unk)
		TimeOffsetReport (Unk)
		TimeOffsetSet (Unk)
TimeParameters	0x8B	Set (0x01)
		Get (0x02)
		Report (0x03)
GeographicLocation	0x8C	Set (0x01)
		Get (0x02)
		Report (0x03)

MultiChannelAssociation	0x8E	Set (0x01)
		Get (0x02)
		Report (0x03)
		Remove (0x04)
		GroupingGet (0x05)
		GroupingsReport (0x06)
MultiCmd	0x8F	Encap (Unk)
EnergyProduction	0x90	Get (0x02)
		Report (0x03)
ManufacturerProprietary	0x91	Get (Unk)
		Report (Unk)
ScreenMd	0x92	Get (Unk)
		Report (Unk)
ScreenAttibutes	0x93	Get (Unk)
		Report (Unk)
SimpleAvControl	0x94	Set (Unk)
		Get (Unk)
		Report (Unk)
		SupportedGet (Unk)
		SupportedReport (Unk)
Security*	0x98	SupportedGet (0x02)
		SupportedReport (0x03)
		SchemeGet (0x04)
		SchemeReport (0x05)
		NetworkKeySet (0x06)
		NetworkKeyVerify (0x07)
		SchemeInherit (0x08)
		NonceGet (0x40)
		NonceReport (0x80)
		MessageEncap (0x81)
		MessageEncapNonceGet (0xc1)

IpConfiguration	0x9A	Set (Unk)
		Get (Unk)
		Report (Unk)
		Release (Unk)
		Renew (Unk)
AssociationCommand-Configuration	0x9B	SupportedRecordsGet (0x01)
		SupportedRecordsReport (0x02)
		Set (0x03)
		Get (0x04)
		Report (0x05)
SensorAlarm	0x9C	Get (0x01)
		Report (0x02)
		SupportedGet (0x03)
		SupportedReport (0x04)
SilenceAlarm	0x9D	Set (Unk)
SensorConfiguration	0x9E	Set (Unk)
		Get (Unk)
		Report (Unk)

Appendix B. DoD Integration of WSNs

The vignette below illustrates the use of WSN in a deployed combat environment and the security implications inherent with its employment.

Abbreviations

AASLT - Air Assault
ABN - Airborne
BDA - Battle Damage Assessment
CONOP - Concept of the Operation
DIV - Division
FARP - Forward Area Refueling Point
FSC - Forward Support Company
FOB - Forward Operating Base
GFC - Ground Force Commander
IBCT - Infantry Brigade Combat Team
ID - Infantry Division
ISO - In Support Of
JAF - Jalalabad Air Field
LLVI - Low Level Voice Intercept
NLT - No Later Than
OEF - Operation Enduring Freedom
OP - Outpost
RC-E - Regional Command East
RIP - Relief in Place
RTB - Return to Base
SOP - Standard Operating Procedures
TIC - Troops in Contact
TOC - Tactical Operations Center

B.1 Introduction

1st Squadron, 32nd Cavalry Regiment, 1st IBCT, 101st ABN DIV (AASLT) (hereinafter, 1-32 CAV) are deployed ISO of OEF IX. NLT 22 April 2010, 1-32 CAV conducts a RIP with 1-112th CAV, 2nd IBCT, 4th ID of FOB Bostick, RC-E and subordinate OPs.

FOB Bostick recently upgraded the FARP to include a Z-Wave fuel tank gauge sensor to monitor the fuel tank levels. The fuel sensor is frequently polled by the sensor network controller in the TOC to ensure fuel levels are known. In this way, the FSC can track fuel usage trends and also determine when the tanks need replenishing.

FARPs are crucial in order to keep rotary wing assets near during operations. When Blackhorse Troop, 1-32 CAV develops the CONOP for their upcoming presence patrol, intelligence reports suggests enemy activity in the area at H-hour. Therefore,

a request is made for AH-64D support providing overwatch and attack capabilities if needed.

All rotary wing assists are JAF-based and must travel to their requested location. Thankfully, FARPs allow air support longer station time by providing a refueling point near the supported unit.

B.2 Scenario 1

While on patrol, Blackhorse Troop reports TIC. Immediately, air support is online and ready to engage. The GFC provides authorization and the AH-64Ds begin to neutralize the targets. After 2 hours on target, the AH-64Ds report their intent to depart to the FARP in 10 minutes. Per SOP, Blackhorse Troop halts assaulting through their objective until the AH-64Ds return and fire superiority is regained.

Upon return, the AH-64D eliminates all targets and Blackhorse Troop proceeds with the BDA. Mission complete, RTB.

B.3 Scenario 2

While on patrol, Blackhorse Troop reports TIC. Immediately, air support is online and ready to engage. The GFC provides authorization and the AH-64Ds begin to neutralize the targets. After 2 hours on target, the AH-64Ds report their intent to depart to the FARP in 10 minutes. Per SOP, Blackhorse Troop halts assaulting through their objective until the AH-64Ds return and fire superiority is regained.

Unbeknownst to 1-32 CAV, at H-Hour-72, the enemy are aware that 1-32 CAV employed sensor networks in the TOC. Using wireless packet capture tools, the enemy conducts sensor network reconnaissance, network enumeration, and device fingerprinting and identifies the fuel sensor. After some basic Google-hacking, the enemy

determines the fuel sensor reports status updates to the controller with a fuel reading. After capturing packets sent by the device to the controller, the enemy crafts their own packets that reports fuel levels at 100% and persistently injects them into the network. Even when the fuel sensor legitimately reports a low fuel level, the enemy subsequently injects a 100% fuel level effectively nullifying the fuel sensor transmissions. As a result, the TOC believes the FARP to have sufficient fuel for upcoming operations.

Upon arrival at the FARP, refueling personnel realize fuel tanks are in fact empty although reports in the TOC suggest they are full. As a result, the AH-64Ds cannot refuel and the fuel that is left is not enough for the AH-64Ds to return to JAF. They cannot re-enter the fight and cannot RTB.

Blackhorse Troops GFC radios the TOC for AH-64D status. The TOC responds, "NoGo on air support. AH-64Ds are out of the fight. No fuel, no flying." Realizing that the enemy estimations suggest a 4:1 ratio in favor of the enemy, Blackhorse Troop has no other option but to retrograde and reorganize.

B.4 Conclusion

Some DoD organizations already employ similar fuel sensors. With the current down sizing of the force, it is necessary to optimize performance at lower costs. Therefore, the FARP fuel sensor scenario is a reasonable one. This research provides system integration engineers with possible attacks in WSN, countermeasures, and mitigation strategies.

Bibliography

[Ahm10] S. Ahmad. WPA Too! *DEFCON 18*, 2010.

[AP13] L. Apa and C. Penagos. Compromising Industrial Facilities from 40 Miles Away. *Black Hat Conference*, 2013.

[Bee08] Freescale Beestack: Application Development Guide. Retrieved on 10 December, 2015 from `http://cache.freescale.com/files/rf\_if/doc/user\_guide/BSADG.pdf?fsrch=1`, January 2008.

[BFH+15] C. Badenhop, J. Fuller, J. Hall, B. Ramsey, and M. Rice. Evaluating ITU-T G.9959 Based Wireless Systems Used in Critical Infrastructure Assets. In M. Rice and S. Shenoi, editors, *Critical Infrastructure Protection IX*, volume 466 of *IFIP Advances in Information and Communication Technology*, pages 209–227. Springer International Publishing, 2015.

[BW15] M. Barcena and C. Wueest. Insecurity in the Internet of Things. Retreived 6 October, 2015 from `https://www.symantec.com/content/en/us/enterprise/media/security\_response/whitepapers/insecurity-in-the-internet-of-things.pdf`, March 2015.

[CBS13] D. Crowley, D. Bryan, and J. Savage. Home Invasion V2.0 - Attacking Network-Controlled Hardware. *Black Hat Conference*, 2013.

[CV11] A. Cui and J. Voris. Print Me If You Dare: Firmware Modification Attakcs and the Rise of Printer Malware. *28th Chaos Communication Congress*, December 2011.

[DHS10] Privacy Impact Assessment for the DHS Sensor Web. Retrieved on 5 December, 2015 from, `http://www.dhs.gov/sites/default/files/publications/privacy_pia_st_sensorweb.pdf`, January 2010.

[FG13] B. Fouladi and S. Ghanoun. Security Evaluation of the Z-Wave Wireless Protocol. *Black Hat Conference*, 2013.

[FNS12] S. Fahmy, A. Nasir, and N. Shamsuddin. Wireless network attack: Raising the awareness of Kampung WiFi residents. In *2012 International Conference on Computer Information Science (ICCIS)*, volume 2, pages 736–740, June 2012.

[FR15a] J. Fuller and B. Ramsey. Rogue Z-Wave Controllers: A Persistent Attack Channel. In *Tenth IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp)*, pages 734–741, October 2015.

[FR15b] J. Fuller and B. Ramsey. Stealty and Persistent Access to Z-Wave Gateways. *DerbyCon 5.0 "Unity"*, 2015.

[FRPR16] J. Fuller, B. Ramsey, J. Pecarina, and M. Rice. Wireless Intrusion Detection of Covert Channel Attacks in ITU-T G.9959-based Networks. In *Eleventh International Conference on Cyber Warfare and Security*, March 2016.

[GBM+12] T. Goodspeed, S. Bratus, R. Melgares, R. Speers, and S. Smith. Apido: Tools for Exploring the Wireless Attack Surface in Smart Meters. In *45th Hawaii International Conference on System Science (HICSS)*, pages 2133–2140, January 2012.

[GSC+14] F. Greitzer, J. Strozer, S. Cohen, A. Moore, D. Mundie, and J. Cowley. Analysis of Unintentional Insider Threats Deriving from Social Engineering Exploits. In *2014 IEEE Security and Privacy Workshops (SPW)*, pages 236–250, May 2014.

[HGSW11] L. Hormann, P. Glatz, C. Steger, and R. Weiss. Measuring the state-of-charge - Analysis and impact on wireless sensor networks. In *IEEE 36th Conference on Local Computer Networks (LCN)*, pages 982–985, October 2011.

[HRRL16] J. Hall, B. Ramsey, M. Rice, and T. Lacey. Z-Wave Network Reconnaissance and Transceiver Fingerprinting Using Software-Defined Radios. In *Eleventh International Conference on Cyber Warfare and Security*, March 2016.

[ITU15] ITU-T G.9959 - Short Range Narrow-Band Digital Radiocommunication Transceivers - PHY, MAC, SAR and LLC layer specifications, January 2015.

[KP12] C. Kiraly and G. Picco. Where's the mote? Ask the MoteHunter! In *7th IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp)*, pages 982–990, October 2012.

[Kru04] C. Kruegel. *Intrusion Detection and Correlation: Challenges and Solutions*. Springer-Verlag TELOS, Santa Clara, CA, USA, 2004.

[Lan05] H. Lane. Security Vulnerabilities and Wireless LAN Technology. Technical report, SANS Institute InfoSec Reading Room, 2005.

[LK12] N. Langhammer and R. Kays. Performance Evaluation of Wireless Home Automation Networks in Indoor Scenarios. *IEEE Transactions on Smart Grid*, 3(4):2252–2261, December 2012.

[LKP10] M. Li, I. Koutsopoulos, and R. Poovendran. Optimal Jamming Attack Strategies and Network Defense Policies in Wireless Sensor Networks. *IEEE Transactions on Mobile Computing*, 9(8):1119–1133, August 2010.

[MG10] D. Martins and H. Guyennet. Steganography in MAC Layers of 802.15.4 Protocol for securing Wireless Sensor Networks. In *International Conference on Multimedia Information Networking and Security*, pages 824–836, November 2010.

[NIS07] NIST. *Guide to Intrusion Detection and Prevention Systems (IDPS)*. Number 800-94. February 2007.

[Nou12] G. Noubir. *Handbook on Securing Cyber-Physical Critical Infrastructure: Foundations and Challenges*, chapter Robust Wireless Infrastructure Against Jamming Attacks, pages 132–145. M. Kaufmann, 2012.

[OGPK13] T. Oluwafemi, S. Gupta, S. Patel, and T. Kohno. Experimental Security Analyses of Non-Networked Compact Flourescent Lamps: A Case Study of Home Automation Security. In *Learning from Authoritative Security Experiment Results (LASER)*, October 2013.

[Ope14] OpenZWave. Retrieved on 4 November, 2015 from `http://openzwave.com/`, 2014.

[Ore03] V. Orel. *A Handbook of Gemanic Etymology*. Brill Academic Publisher, October 2003.

[PIK11] K. Pelechrinis, M. Iliofotou, and S. Krishnamurthy. Denial of Service Attacks in Wireless Networks: The Case of Jammers. *Communications Surveys Tutorials, IEEE*, 13(2):245–257, February 2011.

[PLD14] J. Picod, A. Lebrun, and J. Demay. Bringing Software Defined Radio to the Penetration Testing Community. *Black Hat Conference*, 2014.

[PTBR14] H. Patel, M. Temple, R. Baldwin, and B. Ramsey. Application of Ensemble Decision Tree Classifiers to ZigBee Device Network Authentication Using RF-DNA Fingerprinting. In *Ninth International Conference on Cyber Warfare and Security (ICCWS)*, pages 176–186, March 2014.

[RaZ15] RaZberry Project. Z-Way Developer's Documentation. Retrieved on 13 November, 2015 from `http://razberry.z-wave.me`, November 2015.

[RM12] B. Reaves and T. Morris. Analysis and mitigation of vulnerabilities in short-range wireless communications for industrial control systems. *International Journal of Critical Infrastructure Protection*, 5(34):154 – 174, 2012.

[RM13] B. Ramsey and B. Mullins. Defensive Rekeying Strategies for Physical-Layer-Monitored Low-Rate Wireless Personal Area Networks. In J. Butts and S. Shenoi, editors, *Critical Infrastructure Protection VII*, volume 417 of *IFIP Advances in Information and Communication Technology*, pages 63–79. Springer Berlin Heidelberg, 2013.

[RMSB13] B. Ramsey, B. Mullins, R. Speers, and K. Batterton. Watching for Weakness in Wild WPANs. In *IEEE Military Communications Conference, MILCOM 2013*, pages 1404–1409, November 2013.

[RMTG15] B. Ramsey, B. Mullins, M. Temple, and M. Grimaila. Wireless Intrusion Detection and Device Fingerprinting through Preamble Manipulation. *IEEE Transactions on Dependable and Secure Computing*, 12(5):585–596, September 2015.

[RMW12] B. Ramsey, B. Mullins, and E. White. Improved Tools for Indoor ZigBee Warwalking. In *7th IEEE International Workshop on Practical Issues in Building Sensor Network Applications (SenseApp)*, pages 921–924, October 2012.

[RSRR+10] S. Reddy, K. Sai Ramani, K. Rijutha, S.M. Ali, and P. Reddy. Wireless hacking - a WiFi hack by cracking WEP. In *2010 2nd International Conference on Education Technology and Computer (ICETC)*, volume 1, pages V1–189–V1–193, June 2010.

[SDv10] M. Strasser, B. Danev, and S. Čapkun. Detection of Reactive Jamming in Sensor Networks. *ACM Transactions on Sensor Networks*, 7(2):16:1–16:29, 2010.

[SK14] M. Scanlon and T. Kechadi. The Case for a Collaborative Universal Peer-to-Peer Botnet Investigation Framework. In *Ninth International Conference on Cyber Warfare and Security*, pages 287–293, 2014.

[Sma15] SmartThings API. Retrieved on 10 December, 2015 from `https://graph.api.smartthings.com`, December 2015.

[Smi14] E. Smith. Global Broadband and WLAN (Wi-Fi) Networked Households Forecast 2009-2018. Retrieved on 10 December, 2015 from `https://www.strategyanalytics.com/access-services/devices/connected-home/consumer-electronics/reports/report-detail/global-broadband-and-wlan-(wi-fi)-networked-households-forecast-2009-2018`, October 2014.

[SWH11] J. Stankovic, A. Wood, and T. He. Realistic Applications for Wireless Sensor Networks. In Sotiris Nikoletseas and Jos D.P. Rolim, editors, *Theoretical Aspects of Distributed Computing in Sensor Networks*,

Monographs in Theoretical Computer Science. An EATCS Series, pages 835–863. Springer Berlin Heidelberg, 2011.

[Wig15] WiGLE: Wireless Network Mapping. Retrieved on 10 December, 2015 from `https://wigle.net/stats`, December 2015.

[XTZW05] W. Xu, W. Trappe, Y. Zhang, and T. Wood. The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '05, pages 46–57, New York, NY, USA, 2005. ACM.

[Yus08] S. Yusufovna. Integrating Intrusion Detection System and Data Mining. In *International Symposium on Ubiquitous Multimedia Computing (UMC)*, pages 256–259, October 2008.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
24–03–2016	Master's Thesis	Aug 2014 — Mar 2016

4. TITLE AND SUBTITLE

A Misuse-Based Intrusion Detection System for ITU-T G.9959 Wireless Networks

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Fuller, Jonathan D., Captain, USA

5d. PROJECT NUMBER

16G129

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-16-M-016

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL Information Directorate
POC: Hiren Patel,
Cyber ISR Program Manager
ATTN: Hiren Patel AFRL/RIGD
Mailstop: 525 Brooks Rd, Rome NY, 13441
Email: hiren.patel@us.af.mil, Phone: (315) 330-3415

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RIGD

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT Wireless Sensor Networks (WSNs) provide low-cost, low-power, and low-complexity systems tightly integrating control and communication. Protocols based on the ITU-T G.9959 recommendation specifying narrow-band sub-GHz communications have significant growth potential. The Z-Wave protocol is the most common implementation. Z-Wave developers are required to sign nondisclosure and confidentiality agreements, limiting the availability of tools to perform open source research. This work discovers vulnerabilities allowing the injection of rogue devices or hiding information in Z-Wave packets as a type of covert channel attack. Given existing vulnerabilities and exploitations, defensive countermeasures are needed. A Misuse-Based Intrusion Detection System (MBIDS) is engineered, capable of monitoring Z-Wave networks. Experiments are designed to test the detection accuracy of the system against attacks. Results from the experiments demonstrate the MBIDS accurately detects intrusions in a Z-Wave network with a mean misuse detection rate of 99%. Overall, this research contributes new Z-Wave exploitations and an MBIDS to detect rogue devices and packet injection attacks, enabling a more secure Z-Wave network.

15. SUBJECT TERMS

Z-Wave, wireless sensor networks, rogue device, covert channel, intrusion detection

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Maj Benjamin W. Ramsey, PhD
U	U	U	U	125	19b. TELEPHONE NUMBER *(include area code)* (937)255-3636x4603; benjamin.ramsey@afit.edu