COMPUTERS & SECURITY 64 (2017) 44–58

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

Misuse-based detection of Z-Wave network attacks

Jonathan D. Fuller [a,*], *Benjamin W. Ramsey* [b], *Mason J. Rice* [b], *John M. Pecarina* [b]

[a] *United States Army Information Systems Engineering Command Fort Huachuca, AZ 85613, USA*
[b] *Air Force Institute of Technology, Department of Electrical and Computer Engineering, Wright-Patterson Air Force Base, OH 45433, USA*

ARTICLE INFO

Article history:
Received 17 March 2016
Received in revised form 25 August 2016
Accepted 16 October 2016
Available online 21 October 2016

Keywords:
Z-Wave
Wireless sensor networks
Vulnerability analysis
Wireless security
Intrusion detection

ABSTRACT

Wireless Sensor Networks (WSNs) are becoming ubiquitous, providing low-cost, low-power, and low-complexity systems in which communication and control are tightly integrated. Although much security research into WSNs has been accomplished, researchers struggle to conduct thorough analyses of closed-source proprietary protocols. Of the numerous available and underanalyzed proprietary protocols, those based on the ITU-T G.9959 recommendation specifying narrow-band sub-GHz communications have recently experienced significant growth. 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. Given recently demonstrated attacks against Z-Wave networks, defensive countermeasures are needed. This work extends an existing implementation of a Z-Wave Misuse-Based Intrusion Detection System (MBIDS). A side-by-side comparison is performed through experimentation to measure misuse detection accuracy of the baseline and extended MBIDS implementations. Experiment results determine the extended MBIDS achieves a mean misuse detection rate of 99%, significantly improving the security posture in MBIDS-monitored Z-Wave networks.

Published by Elsevier Ltd.

1. Introduction

Wireless Sensor Networks (WSNs) are technologies where computing, communications, and control are tightly coupled (Fuller and Ramsey, 2015a). WSNs consist of multiple sensor nodes that collect information, respond to commands, and report updates (Hormann et al., 2011). These agile networks are used in military surveillance, health care, environmental science, smart metering, and home automation (Ramsey and Mullins, 2013; Reaves and Morris, 2012; Stankovic et al., 2011). The use of WSNs is increasing because they extend communications ranges at low-cost, low-power, and with low-complexity (Patel et al., 2014).

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. ITU-T G.9959 specifies short range, narrow-band sub-GHz communications (International Telecommunication Union – Telecommunication Standardization Sector G.9959, 2015). The most common implementation of the ITU-T G.9959 recommendation is commercially known as Z-Wave. However, security evaluations of the Z-Wave protocol are difficult because developers are required to sign nondisclosure

* *Corresponding author.*
E-mail address: jonathan.d.fuller1@gmail.com (J.D. Fuller).
http://dx.doi.org/10.1016/j.cose.2016.10.003
0167-4048/Published by Elsevier Ltd.

agreements (NDAs) preventing them from legally disclosing any proprietary information limiting the availability of open-source tools to perform open source research. New and emerging network protocols are often assumed to be secure, but are not vetted until tools exist for security research (Goodspeed et al., 2012).

Vulnerabilities in the Z-Wave protocol and vendor implementations have been discovered and are discussed in Section 3. Although the discovery of vulnerabilities and exploitations is necessary to establish the security implications, proper security countermeasures are needed in order to safely employ Z-Wave networks.

A Misuse-Based Intrusion Detection System (MBIDS) is proposed in Fuller et al. (2016) that allows an investigator or system administrator to monitor Z-Wave traffic in real-time for any transmissions intended to disrupt normal network operations; however, there are limitations. The evaluated detection accuracy for maliciously injected packets with invalid payloads only resulted in a mean detection rate of 92% and lacks the ability to detect valid packets injected by an attacker attempting to disrupt normal network operation.

The focus herein is to enhance the MBIDS. This research significantly improves the system to increase the overall mean detection rate by providing additional enhancements and also reduces the cost of employment providing security researchers a low-cost extensible framework.

This work is organized as follows. Section 2 discusses the details of the Z-Wave protocol. Z-Wave protocol vulnerabilities are presented in Section 3 and demonstrate the need for security countermeasures. The methodology is presented in Section 4, explaining the system under test and experiment designs. Section 5 discusses the results and analysis of the experiments. Lastly, the conclusion and future work are highlighted in Section 6.

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 (International Telecommunication Union – Telecommunication Standardization Sector G.9959, 2015). This ensures interoperability between vendor devices, but differ at the Application Layer (Fig. 1). ITU-T G.9959-based networks operate in the industrial, scientific, and medical (ISM) bands. The available frequencies are regionally dependent and vary within regions subject to the transmission data rates being used. Available frequencies and data rates are further discussed in Section 2.1.

There are two types of nodes in Z-Wave networks: control nodes send commands and slave nodes respond to commands. As a meshed topology network, slave nodes also forward commands to other nodes that are not directly reachable by the control node. Message forwarding has a hop limit of four nodes and a maximum of 232 nodes are allowed in a Z-Wave network. For network overlap, each Z-Wave network has a unique 4B `HomeID` specified by the controller. The Z-Wave protocol consists of five layers where the Adaptive Layer is a set of three layers: SAR, Network (NWK), and Encryption (ENC) as illustrated in Fig. 1.

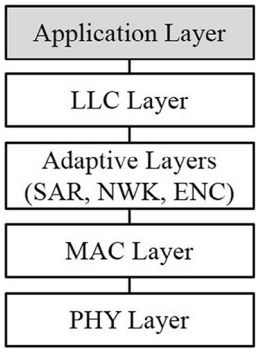


Fig. 1 – Z-Wave protocol reference model.

2.1. *PHY and MAC layers*

The PHY layer is responsible for the transfer of data between nodes in the network (International Telecommunication Union – Telecommunication Standardization Sector G.9959, 2015). The protocol supports three data rates: 9.6 kbps (R1), 40 kbps (R2), and 100 kbps (R3) (International Telecommunication Union – Telecommunication Standardization Sector G.9959, 2015). Fig. 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.

Furthermore, the rates generally determine what transmission frequency will be used. The available frequencies range from 868.40 MHz to 926.30 MHz depending on the geographic region. Specifically, devices operating in the European Union, Taiwan, or South Africa transmit either at the 869.40 MHz frequency at R3 or the 868.40 MHz at R1 and R2. However, if transmitting in the United States of America, Canada, Chile, or Mexico, frequencies used are 916.00 MHz at R3 and 908.40 MHz at R1 and R2 (ZWave Alliance Recommendation, 2014). There are approximately 130 countries with Z-Wave transmission frequencies (ZWave Frequency Ccoverage, 2016) and more to be established as its use continues to expand.

The MAC layer uses carrier sense multiple access with collision avoidance to moderate access to the wireless medium. It is 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 Fig. 2. When a node transmits a 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. *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.

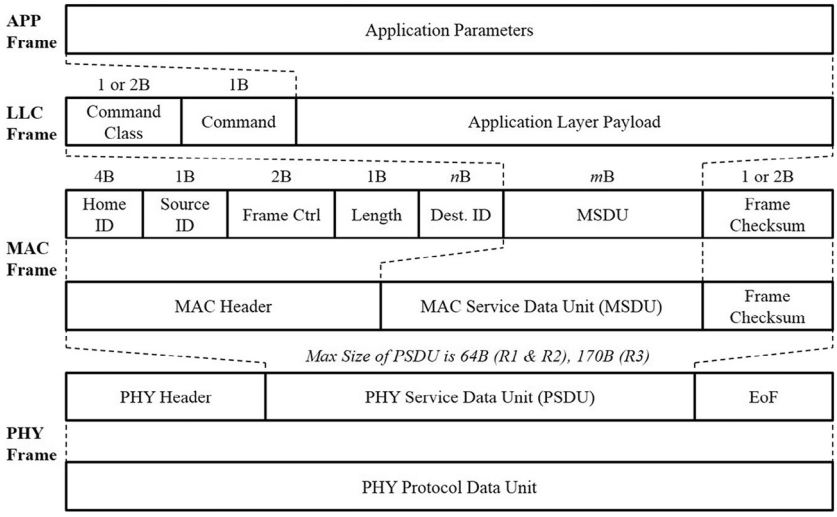


Fig. 2 – Z-Wave frame structure for APP, LLC, MAC, and PHY layers.

The sublayer is selected based on values set in the Frame Control field of the MAC Header or the Command Class (`CmdCl`) field of the LLC Layer (Fig. 2). If specific values in the Frame Control field are not set, the packet is forwarded to the following layer without action.

Notably, the Z-Wave protocol supports the Advanced Encryption Standard with 128-bit keys. Although supported, encryption is not required (Hall and Ramsey, 2016). In the low-rate, low-power WSNs, memory and power are limited, discouraging vendors from adding features unless necessary (Freescale Beestack: Application Development Guide, 2008). Hall and Ramsey found that only 9 of 32 Z-Wave end devices examined were even capable of supporting encryption (Hall and Ramsey, 2016). Surveys of similar protocols conclude the use of encryption is not universal (Ramsey et al., 2013).

2.3. *LLC layer and Application Layer*

The LLC layer consists of the `CmdCl`, Command (`Cmd`), and Application Layer Payload (Application Parameters) (Fig. 2). `CmdCls` represents the function of Z-Wave devices (e.g., a door lock supports different `CmdCls` than a power switch). An example is the *Binary Switch* `CmdCl`. The Binary Switch uses three Commands `Cmds`: *Set* to turn a device on or off, *Get* to request a device status, and *Report* to respond to the request.

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 `Cmd` to perform a specific function.

3. Z-Wave vulnerability analysis and exploitation

Recent works evaluate the Z-Wave protocol for vulnerabilities. These works are categorized as the Radio Frequency (RF) approach, Gateway approach, or Hybrid approach, exploiting PHY layer transmissions, device vulnerabilities, or both, respectively. The RF approach includes packet capture and injection attacks whereas the Gateway approach includes exploitation of Z-Wave gateway controllers.

3.1. *RF approach*

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 and injecting wireless information can inject forged packets into the network. Since Z-Wave developers are required to sign NDAs, researchers not willing to sign an NDA use publicly available tools to conduct packet capture and injection attacks against Z-Wave networks.

Fouladi and Ghanoun (2013) provide the first public vulnerability research of the Z-Wave protocol. They discover an implementation error used in a Z-Wave-compliant door lock that allows them to reset the established network key. They capture the Z-Wave packet containing the `HomeID` of the controller and Node ID of the device, reset the network key to a known value, and inject unauthorized commands into the Z-Wave network. As a result, they are able to open and close the lock without invoking the controller. Badenhop and Ramsey (2016) validate and extend the security layer analysis in Fouladi and Ghanoun (2013) and identify a hardware key extraction vulnerability. Exploiting this vulnerability allows them to inject authenticated and encrypted Z-Wave frames.

Another analysis of Z-Wave is the *Scapy-Radio* project (Picod et al., 2014). Scapy-Radio combines Scapy and gnuRadio software on a Software-Defined Radio (SDR) to capture traffic and replay packets back into the Z-Wave network. Picod et al. (2014) use Scapy-Radio to intercept an ON command sent from a Z-Wave controller to an alarm device and subsequently injects and OFF command to the alarm, effectively disabling it.

The AFIT Sniffer (Badenhop et al., 2015; Hall et al., 2016) is an extension of Scapy-Radio (Picod et al., 2014). The custom WireShark dissector used in Badenhop et al. (2015) allows them to passively discriminate between Z-Wave devices by functionality and vendor. Hall et al. (2016) extend the AFIT Sniffer

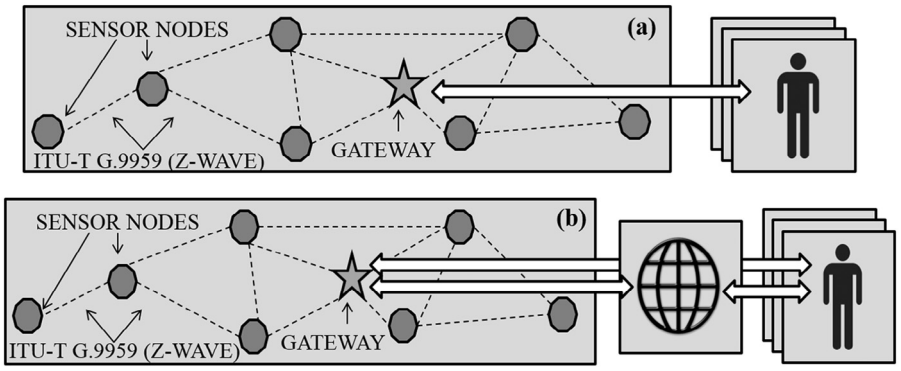


Fig. 3 – (a) Local gateway access: users manage their Z-Wave HAN via their gateway locally. All device control is performed from inside the home. (b) Global gateway access: users manage their Z-Wave HAN via their gateway locally or globally. Z-Wave devices can be controlled using any Internet connected device (Fuller and Ramsey, 2015a).

and demonstrate transceiver fingerprinting through preamble manipulation using two low-cost HackRF One SDRs. The result is an open source framework called *EZ-Wave*: a set of Z-Wave reconnaissance tools capable of network discovery and enumeration, device fingerprinting, and device status information gathering.

3.2. *Gateway approach*

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 (Fig. 3a). Gateway controllers provide Internet connectivity allowing the user to remotely access and mange the Z-Wave network (Fig. 3b). An example is the user unlocking their front door with a cellular phone. Thus, gateway controllers are the preferable option.

This global access point provides an attack vector. After gaining access, the inherent vulnerabilities in Z-Wave gateways allow an attacker to easily take control of the network (Fuller and Ramsey, 2015b). The resulting implications of the inherent vulnerabilities are seen in the following works.

Crowley et al. (2013) find several vulnerabilities that expose sensitive information from a Z-Wave gateway controller, allowing an attacker to fully control devices on the network. Crowley et al. (2013) also exploit a Lua code execution vulnerability and successfully create a backdoor account on the gateway.

Barcena and Wueest (2015) discuss multiple gateway insecurities. During their research, Barcena and Wueest (2015) poison the gateway Address Resolution Protocol to redirect gateway 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.

Recently, Fuller and Ramsey (2015a) demonstrate rogue controller injection in Z-Wave networks. After compromise of the gateway, `HTTP` packet capture and replay are used to activate device inclusion allowing an attacker to inject a rogue controller to the network. The rogue controller maintains persistent access to Z-Wave devices solely communicating in the sub-GHz spectrum. Thus, disabling the gateway controller does not affect rogue device to communication.

3.2.1. *Hybrid approach*

Fuller et al. (2016) illustrate a hybrid approach to attack Z-Wave networks. The *Hel* Attack is a covert Z-Wave channel-initiated Reverse-Secure Shell (R-SSH) attack, where a a Z-Wave RF transmission initiates malware on a compromised gateway controller.

This is possible because the Z-Wave protocol allows an attacker to pad extra bytes in a Z-Wave packet as long at it does not exceed the maximum MSDU size (Fuller et al., 2016). After gateway compromise, installed malware is triggered by hidden information (a covert channel) in a specially crafted Z-Wave packet. After initiation, the malware executes a R-SSH back to any Internet connected computer the attacker chooses.

These early studies illustrate the possible attacks resulting from wireless traffic capture and injection or compromise of the Z-Wave gateway. Therefore, previous works, including newly discovered vulnerabilities, motivate the need for countermeasures toward safe Z-Wave operation.

4. Methodology

This research begins with an enhancement of the MBIDS presented in Fuller et al. (2016). The MBIDS currently includes evaluations of Z-Wave `CmdCls`, `Cmds`, and Payload (`Pld`) parameters. This allows the detection of manipulated packet injection attacks, including the Hel Attack (Fuller et al., 2016). These enhancements are meant to increase the detection rate of maliciously injected packets, reinforcing the defensibility of Z-Wave networks. It is hypothesized that the system can be enhanced if captured packets are compared with gateway log file entries. This may allow the MBIDS to detect spoofed controllers even if all packet parameters are correct and the packet is classified as valid. It is also hypothesized that routed frames are improperly classified because the MBIDS misidentifies them as a standard Z-Wave packet. Therefore, reverse engineering the routing protocol allows for an in-depth dissection of routed packets providing the MBIDS the ability to properly evaluate them.

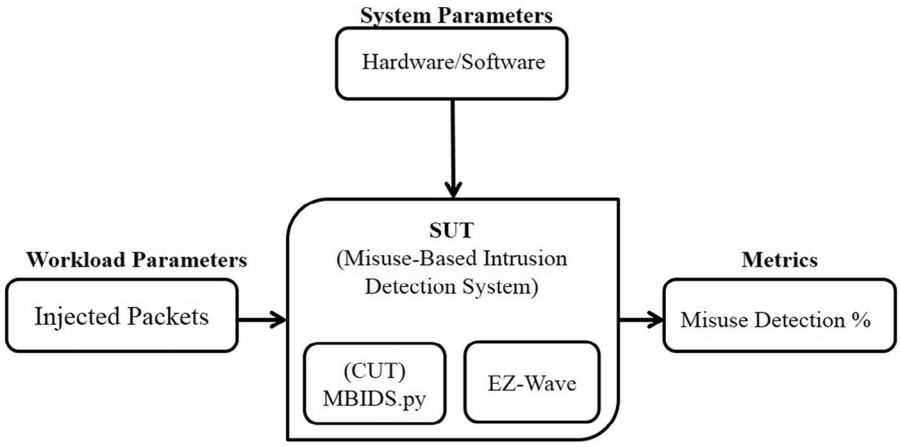


Fig. 4 – System under test diagram for the MBIDS.

Presently, the MBIDS supports the evaluation of 10 Z-Wave `CmdCls`. Adding the evaluation of another `CmdCl` provides broader support for the Z-Wave protocol. Furthermore, the high cost of the underlying SDR used in Fuller et al. (2016) makes employment relatively infeasible to many security researchers. Thus, this work evaluates the possibility of reducing the cost while maintaining accuracy.

Based on the hypotheses, this research answers the following questions:

1. What constitutes misuse cases in Z-Wave transmissions? *Specifically, for packets with the newly added* `CmdCl`.
2. Do the MBIDS enhancements increase attack detection accuracy?
3. Can the MBIDS cost of employment be reduced without performance degradation?

4.1. *System boundaries*

The System Under Test (SUT) is the MBIDS, a monitoring tool for Z-Wave networks capable of detecting previously discussed attacks (Section 3). A block diagram of the SUT is illustrated in Fig. 4. The SUT consists of the Component Under Test (CUT): *Enhanced_MBIDS.py* (packet evaluation software) and EZ-Wave (Hall et al., 2016). All parameters and components are discussed in detail below.

4.1.1. *Workload parameters*

The workload of the SUT consists of multiple factors as illustrated in Fig. 5. 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. Factors include the following.

4.1.1.1. *Packet length.* As seen in Fig. 2, the maximum Packet Length (`Len`) is 170B while operating at R3 and 64B while operating at R1 or R2. When this factor is varied, packets with MSDU different lengths are injected to the SUT. Since EZ-Wave is developed to only work at R2 (Hall et al., 2016), any MSDU length greater than 54B is considered invalid (i.e., 64B – (9B MAC Header + 1B checksum) = 54B).

4.1.1.2. *Source and destination ID.* The Z-Wave gateway keeps a record of all device Node IDs for each device included in the network. The boundaries of this factor are Node IDs `0x01`–`0xE8`, given 232 allowable nodes in any given Z-Wave network.

4.1.1.3. *Command class and command.* Each Z-Wave device supports specific `CmdCls`. There are 70 known `CmdCls` and associated `Cmds` derived from OpenZWave (2014), RaZberry Project (2015) and SmartThings API (2015). Thus, the `CmdCl` factor is varied within the 70 that are known. Following is the `Cmd` (Fig. 5). For example, if a packet contains a *Binary Switch* `CmdCl`, associated `Cmd` values are *Set*, *Get*, or *Report*. `Cmds` are varied in crafted packets.

4.1.1.4. *Payload.* Based on the size of the MSDU, the remaining `Pld` length can be calculated. Since the MBIDS evaluates packets at R2, the maximum MSDU length is 54B (Fig. 2). After the `CmdCl` and `Cmd` have been selected, 52B remain for the `Pld`. This variable factor consists of randomly generated byte lengths 0B–52B.

4.1.2. *System parameters*

The system parameters consist of the hardware and software components listed in Table 1. However, the primary

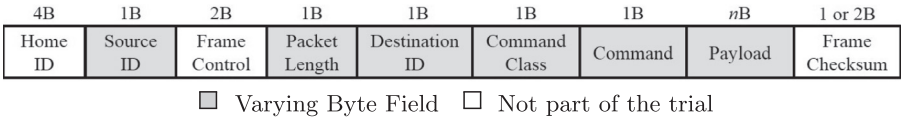


Fig. 5 – Workload parameters and varying byte fields.

Table 1 – SUT system parameters.

Component	Nomenclature	Specifications
Laptop	HP Elitebook 8570w	Intel Core i7-3720QM CPU 16 GB Ram and 64 bit
	Operating System	Ubuntu 14.04
	Software frameworks	LiClipse 2.2.0.2 (Python 2.7) EZ-Wave *MBIDS.py*
Software-defined radio	HackRF One	1 MHz–6 GHz
	VERT900 antenna	sub-GHz 3 dBi

components include EZ-Wave as a packet capture device and the IDS software tool for packet evaluation, *Enhanced_MBIDS.py*.

4.1.3. *Metrics*

The metrics used to evaluate the efficiency of the MBIDS are the proportion of packet capture of the underlying SDR and the packet classification accuracy of MBIDS packet monitoring tool (*Enhanced_MBIDS.py*). The overall misuse detection rate is a ratio of misuse packets detected versus packets classified by the MBIDS as known-good (whether improperly classified).

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.

The EZ-Wave framework (Hall, 2016a) is an SDR-based tool capable of packet capture and injection. In Fuller et al. (2016), an Ettus USRP N210 is the SDR of choice. However, a HackRF One reduces the cost of the MBIDS by at least 80% while maintaining a comparable packet capture accuracy. To evaluate its packet capture accuracy, we compare the HackRF One reception rate with the Z-Wave gateway controller (Raspbery Pi with RaZberry Pi Daughter Card henceforth referred to as RaZberry Pi). The RaZberry Pi is used in this work because of its low-cost and available documentation allowing for a clearer understanding of the software framework as well as supported Z-Wave `CmdCls`. Results of this comparison allow the determination of the best case misuse detection rate of the MBIDS given the underlying packet capture device.

The RaZberry Pi and HackRF One are placed within 1 m of each other. A Z-Wave smart switch is placed 20 m away from the devices under test. Five hundred tests are conducted wherein each test consists of the smart switch sending one packet to the RaZberry Pi. The mean packet reception rate is then calculated for both devices.

The mean packet reception rate for the gateway is 99.9%, whereas the HackRF One maintains a mean packet reception rate of ≈92% (p-value < 0.01). Comparing the packet capture accuracy of the USRP N210 from Fuller et al. (2016) with the HackRF One shows no statistically significant difference in their mean reception rates (Fig. 6). The RaZberry Pi RF front end is tuned per manufacturer specifications and is more effective

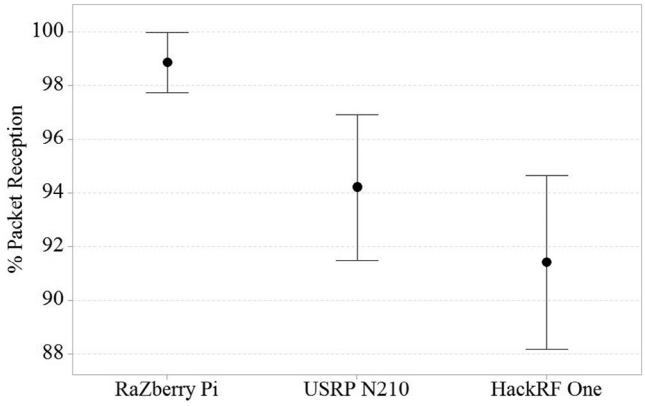


Fig. 6 – Packet capture accuracy comparison: RaZberry Pi, USRP N210, and HackRF One (n = 500, 99% CI).

than the SDR. Given the ≈8% error rate of the HackRF One, it is known, *a priori*, that the MBIDS does not detect 100% of manipulated packets even during a perfect test. Therefore, in a realistic employment, the performance of the MBIDS (ratio of identified misuse cases to total packets injected) is considered with respect to the packet capture accuracy of the HackRF One or any underlying SDR.

4.1.4. *Component under test*

The MBIDS software framework evaluates packets captured by EZ-Wave. 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 and open source documentation in order to further understand the byte fields. This allows the development of the signature database and packet states for comparison.

Fig. 7 illustrates the overall functionality of the design. A series of conditional constructs are used to detect misuse cases in captured packets.

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 containing all Node IDs and all `CmdCls` 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.

1. Packets are captured by EZ-Wave for evaluation by the packet monitoring tool.
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.
3. The next item to evaluate is packet length. 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.
4. Next, group membership is then evaluated. If the `SourceID` 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.

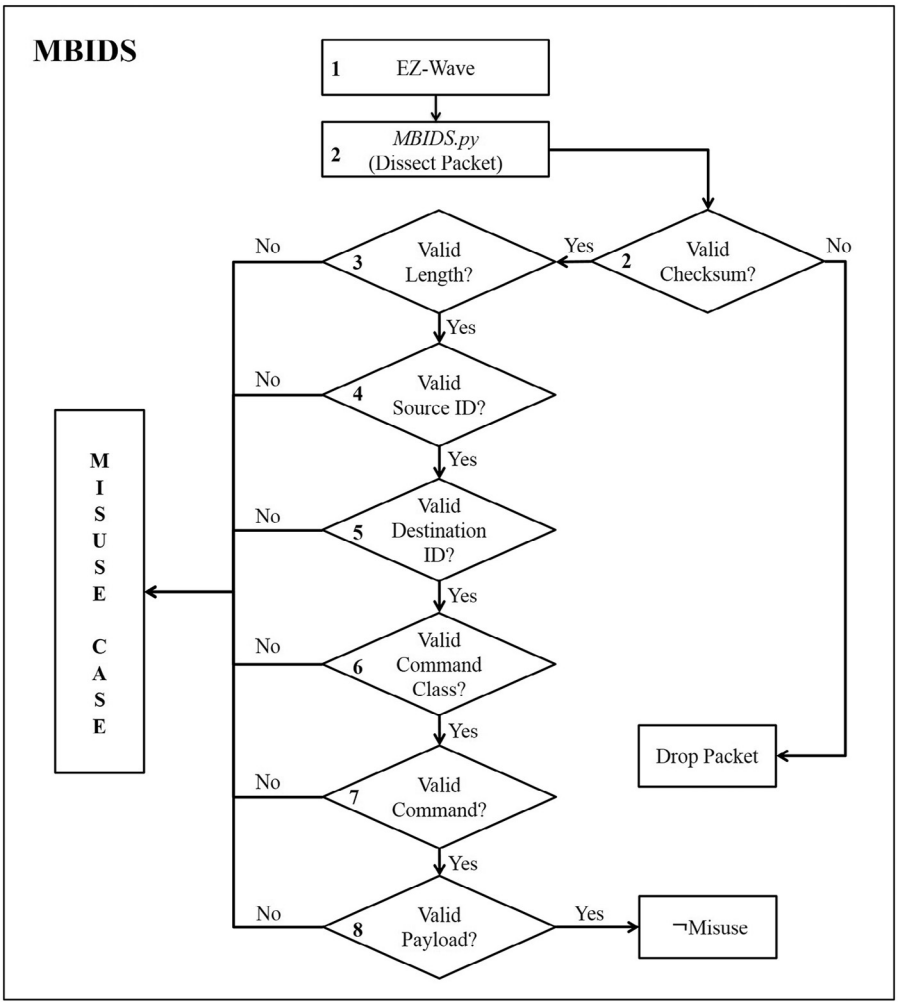


Fig. 7 – Packet data flow through the MBIDS. If the packet has an invalid checksum, it will immediately be dropped. If the packet violates any misuse case, it is logged. Otherwise, it is not a misuse case.

5. Next, the MBIDS evaluates the `DestinationID` field of the packet. If the `DestinationID` is identified as an existing node on the network, further packet evaluation occurs. Otherwise, the packet is considered a misuse case.
6. After group membership is confirmed, the MBIDS evaluates the `CmdCl` sent by a device. Individual Z-Wave devices support different `CmdCls`. For example, a lamp module does not support the *Alarm* `CmdCl` and an alarm does not support a *Door Lock* `CmdCl`. Therefore, if an attacker injects a packet with a valid `SourceID` and `DestinationID`, but a `CmdCl` that is not supported by the `DestinationID` on the Z-Wave network, the packet is considered a misuse case. Otherwise, further evaluation occurs.
7. Once the `CmdCl` is validated, the `Cmd` is checked. Known `Cmds` are derived from OpenZWave (2014), RaZberry Project (2015) and SmartThings API (2015). For example, the *Basic* `CmdCl` only supports `Cmds` *Set*, *Get*, and *Report*. Any other `Cmds` are invalid and considered a misuse case.
8. Lastly, the MSDU payload is evaluated. The `CmdCl` and `Cmd` are part of the MSDU. Any byte field after the `Cmd` is referred to as the `Pld`. Although a `Cmd` may be valid and its associated payload triggers an action, it is possible to pad extra bytes in the `Pld` that are ignored when received by a Z-Wave gateway (Fuller et al., 2016). If the `Pld` is invalid, it is considered a misuse case. Otherwise, the packet is not (¬) a misuse case.

The Z-Wave protocol supports over 70 known `CmdCls` (OpenZWave, 2014; RaZberry Project, 2015; SmartThings API, 2015). In order to develop an MBIDS, it is required that every `CmdCl` is properly understood to detect any violation of signatures or states (Signature-Based and Stateful Protocol Analysis). As discussed, the *Basic* `CmdCl` has three commands, *Set*, *Get*, and *Report*. There are five combinations of *Basic* commands supported by the Z-Wave protocol. As the most basic `CmdCl`, *Basic* has the fewest commands available. At a modest estimation, over 70 known `CmdCls` would allow for over 2000 possible combinations to account for in the MBIDS. Each `CmdCl` has a subsequence `Cmd` between `0x00–0xFF` and `Pld` parameters 1B–52B where each byte is a value `0x00–0xFF`. Therefore, this work provides a proof of concept system with and added `CmdCl` evaluation totaling 11 of the most common `CmdCls`. To

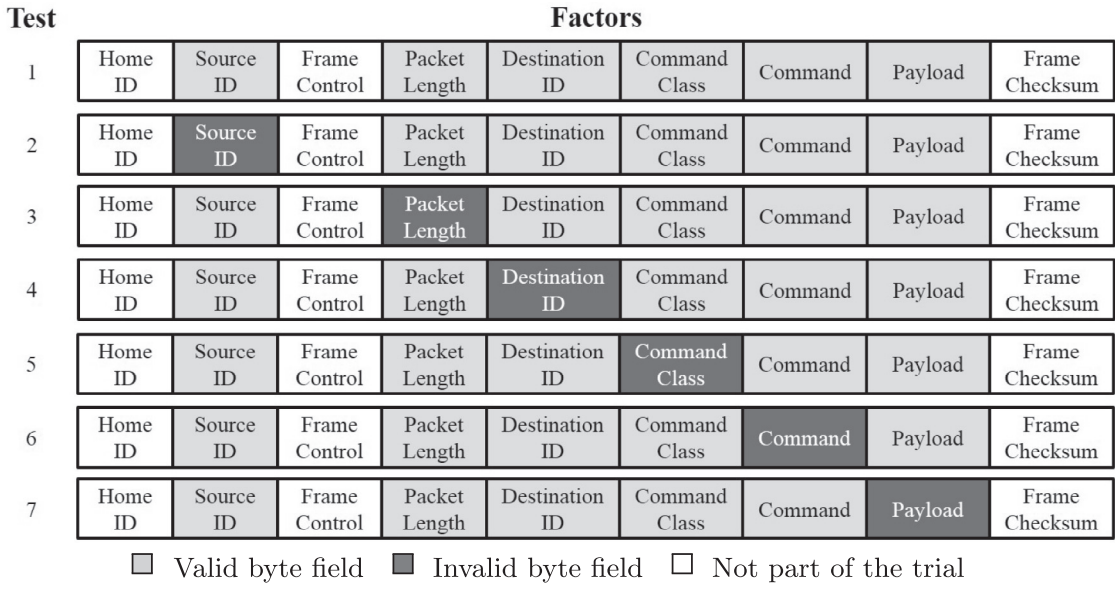


Fig. 8 – Experiments and tests used to achieve the research goals. Experiments 1 and 2 are identical and each consists of 7 tests.

this end, deep packet inspection of normal payloads is conducted to develop known-good packet standards that injected packets are compared against.

This research does not fully consider the two-byte Frame Control field (Fig. 5). 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 (International Telecommunication Union – Telecommunication Standardization Sector G.9959, 2015). It is observed that the RaZberry Pi includes the reserve Header type `0x05` (Hall, 2016b). This one example illustrates the infeasibility of accounting for all known-good Header types. Fully integrating Header type checks in the MBIDS increase improper packet classification reducing the overall effectiveness of the system.

4.2. *Design of experiments*

The overall design for this study consists of two experiments. The design consists of a total of 4200 trials, where each trial consists of a crafted packet wirelessly transmitted to the Z-Wave network. Because the underlying HackRF One is not 100% accurate at packet capture, trials are conducted until a required number of packets are evaluated by the MBIDS. Details of the trial distribution between the experiments and tests are outlined below.

4.2.1. *Experiment 1: manipulated packet injection detection*

Experiment 1 repeats the experiment design in Fuller et al. (2016) and consists of seven tests, each having 300 trials, totaling 2100 trials (Fig. 8). Since this work aims to enhance the system, Experiment 1 is conducted in order to allow a comparison of pre and post enhancement results.

There are up to 232 nodes allowed in any Z-Wave network (International Telecommunication Union – Telecommunication Standardization Sector G.9959, 2015). An exhaustive test would require 232^2 trials to test every possible combination of `SourceID` and `DestinationID`. 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. Not only does every possible combination of `SourceID` and `DestinationID` have to be accounted for, but also `Len`, `CmdCl`, `Cmd`, and `Pld` parameters. Thus, all possible combinations are significantly greater than 232^2. The tests ensure that candidates from all known possibilities are represented. Possibilities include:

- Test 1: All valid byte fields.
- Test 2: Invalid `Len` and all other fields valid.
- Test 3: Invalid `SourceID` and all other fields valid.
- Test 4: Invalid `DestinationID` and all other fields valid.
- Test 5: Invalid `CmdCl` and all other fields valid.
- Test 6: Invalid `Cmd` and all other fields valid.
- Test 7: Invalid `Pld` and all other fields valid.

Given the MBIDS software framework, a packet with invalid `SourceID` and `CmdCl` with all other parameters valid is not a valid combination. After an invalid `SourceID` is checked, packet evaluation ceases and the misuse logs only record an invalid `SourceID`.

All valid fields are selected from the known-good signatures collected for the MBIDS. Invalid fields are randomly generated within accepted invalid parameters for the respective field. Al-

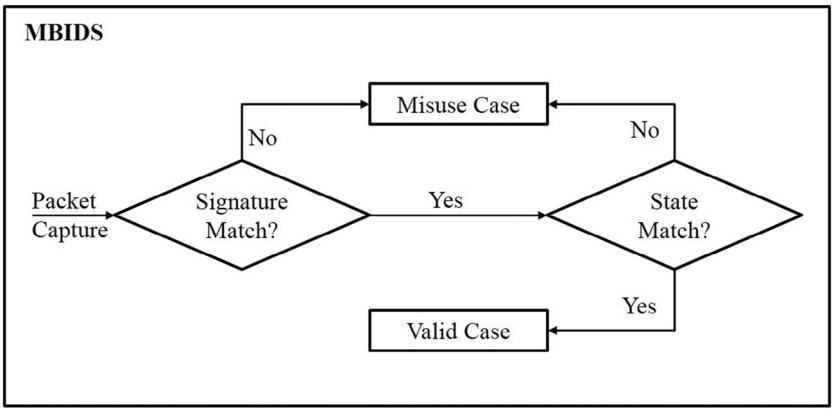


Fig. 9 – If any packet received is classified as a valid packet, the packet is re-evaluated to ensure it is valid.

though a `CmdCl` is represented by a specific byte (e.g., `0x32`) and any byte ranges from `0x00` to `0xFF`, we will assume the attacker will use a proper Z-Wave `CmdCl` to craft their packet. Therefore, `CmdCls` are randomly selected from the list of known `CmdCls`. 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.

4.2.2. *Experiment 2: increased detection rate post enhancement*

After results are gathered from Experiment 1, the system is enhanced and re-evaluated. The first enhancement checks if a transmitted packet should have a previous state. If it is the case, the current state of the transmitted packet is compared with its required previous state. The second enhancement implements an evaluation framework for routed frames.

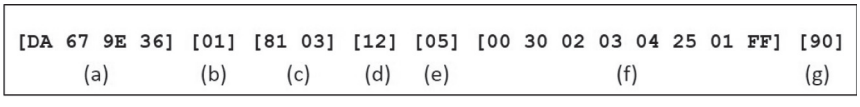
4.2.2.1. *Enhancement 1: packet state analysis.* In the Z-Wave gateway, when a packet is sent by the controller (State 1), a log entry is created (State 2). To reduce incorrect packet classification, if a packet is received by the MBIDS where the `SourceID` is `0x01`, the packet is compared to the gateway 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 `SourceID 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; for the RaZberry Pi used for this experiment, an ≈10 s 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 `SourceID 0x01`, this enhancement strategy re-evaluates the packets after a <10 s 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.

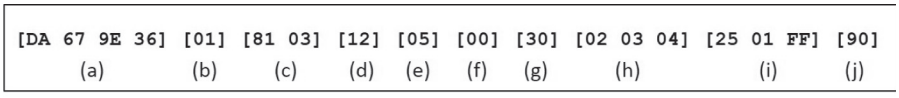
Secondly, if a packet contains `SourceID` and `DestinationID` that are equal, the packet is a misuse case. There are no effects of equal `SourceID` and `DestinationID` in Z-Wave networks. If a device with Node ID `0x05` receives a packet with `SourceID 0x05`, it accepts the packet and executes the commands in the MSDU. To detect this attack, if a packet is classified as a valid packet, but contains equal `SourceID` and `DestinationID`, it is reclassified as a misuse case.

Fig. 9 illustrates the logical topology of the MBIDS with the enhancement strategy. Valid packets are re-evaluated by *Z-Way-Server_Comparator.py* based on their state. If the state is invalid, the packet is reclassified as a misuse case.

4.2.2.2. *Enhancement 2: routing frame evaluation.* When a routed Z-Wave frame is received by the MBIDS pre-enhancement, it



(i) Pre-Enhanement Frame Classification



(ii) Post-Enhanement Frame Classification

Fig. 10 – Routed frame pre and post enhancement classification.

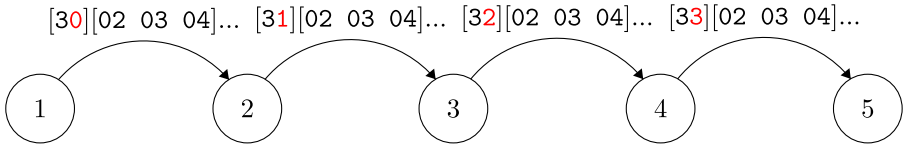


Fig. 11 – Z-Wave mesh topology routing. Node `0x01` sends a routed packet to Node `0x05`. The hop counter is steadily incremented until the final destination is reached.

is classified incorrectly. Before the routed frame evaluation is added to the MBIDS, it is evaluated as seen in Fig. 10(i). The (a) `HomeID`, (b) `SourceID`, (c) Frame Control, (d) Length, (e) `DestinationID`, and (g) Checksum are correctly evaluated. The `Pld` (f) is improperly evaluated. The first byte of the `Pld` `0x00` is considered the No Operation `CmdCl` followed by variable parameters. However, `0x00` is actually the header for a routed payload.

Fig. 10(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 (Stankovic et al., 2011) and nibble 2 is the quantity of hops already completed [0]. The next byte fields (h) consist of the nodes in the route that the packet goes through. When the packet gets to its final destination of Node ID `0x05`, the hop count is 3, and (i) is executed as a Binary Switch `CmdCl` `0x25`, with `Cmd` *Set* (`0x01`) and `Pld` Parameter ON (0xFF). The example mesh network is illustrated in Fig. 11. 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 `CmdCl` sent.

After enhancement, the system is re-evaluated totaling 2100 trials to calculate the post enhancement differences in detection rates.

4.3. *Evaluation technique and environment*

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. In all three experiments, collected data are 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). Experiment 1 confirms the findings in Fuller et al. (2016). In Experiment 2, the system is re-evaluated using the enhancement strategy.

To conduct experiments, a realistic Z-Wave network is engineered with a RaZberry Pi and multiple devices on a LAN backbone. The accuracy of the MBIDS is then tested against two attacks presented in Section 3 to evaluate the efficacy of this approach. Fig. 12 diagrams the experiment setup.

5. Results and analysis

5.1. *Results and analysis of Experiment 1*

The following sections provide the results and analysis of the seven tests conducted in Experiment 1.

5.1.1. *Test 1: all valid byte fields*

In this test, all six fields are valid in order to ensure the MBIDS does not classify the packet as a misuse case. In the attack sce-

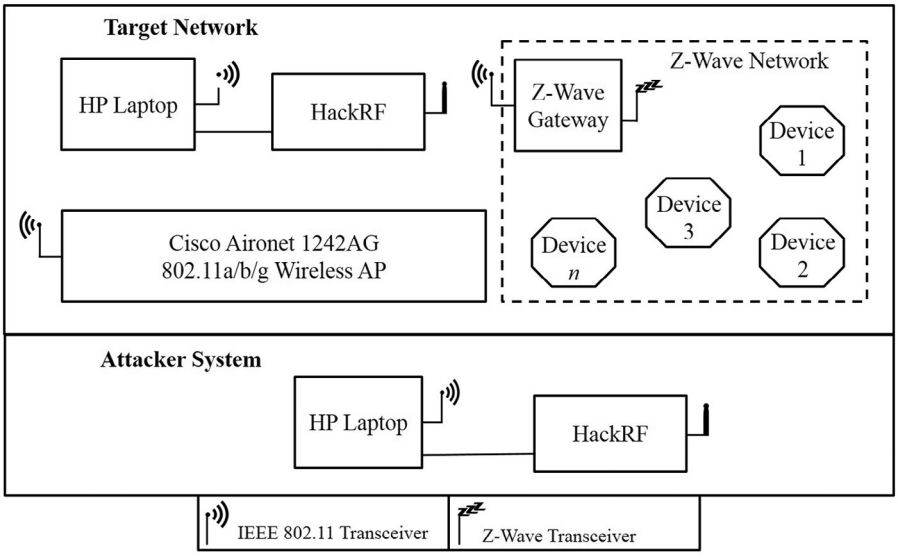


Fig. 12 – Block diagram of the experiment setup.

nario presented in Fuller et al. (2016) or similar attacks, an attacker injects a packet with valid parameters but also attempts to hide information. Since a manipulated injected packet contains some misuse cases, this test does not demonstrate detection against a covert channel attack. However, it is necessary to evaluate the MBIDS for correctness. 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. 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.

5.1.2. *Test 2: invalid packet length*

In Test 2, the injected packet contains an invalid `Len`. 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 have an MSDU that is >54B length to test the detection accuracy of the MBIDS against packets that are too large.

Test 2 provides a mean detection rate of 100% for 300 injected packets containing an invalid `Len`. 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.

5.1.3. *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 `SourceID` or `DestinationID`. After 300 trials for each test, the results are as expected. The MBIDS achieves a 100% mean detection rate for each test.

In Fuller and Ramsey (2015a), when a rogue controller is injected into a Z-Wave network, the controller can communicate with other Z-Wave devices while remaining undetected. However, because an attacker covers their tracks, the controller has an invalid `SourceID`. 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.

5.1.4. *Tests 5 and 6: invalid command class or invalid command*

After an attacker gains access to a Z-Wave network (Fuller and Ramsey, 2015a, 2015b), 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 `CmdCls` that each device supports. Without this knowledge, an attacker crafts a packet with an invalid `CmdCl`. 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 `CmdCl` 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 `CmdCl` is checked for validity.

The same applies to a `Cmd`. Without prior knowledge of a specific device on the Z-Wave network, the attacker will not only know the `CmdCl` but also the corresponding `Cmd`.

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

5.1.5. *Test 7: invalid payload*

The maximum size of the MSDU is 54B. After the `CmdCl` and `Cmd` are chosen, 52B remain. The `Pld` is randomly generated between 0B and 52B and concatenated to the `CmdCl` and `Cmd` before packet injection. This ensures a valid `Len` but the `Pld` contains random invalid byte manipulations.

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 for a review and analysis of why some manipulated injected packets are not flagged as a misuse case.

Further analysis revealed that, of the 11 implemented `CmdCls`, Configuration, Association, Version, and Security `CmdCls` 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 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 2 lists a 100% mean detection rate of invalid `Len`, `SourceID`, `DestinationID`, `CmdCl`, and `Cmd`. 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. If the attacker is not aware of the `SourceID`, `DestinationID`, valid `Len`, `CmdCl`, or `Cmd`, 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 2). Packets with invalid `Pld`s are detected with a mean of 91.7%.

Table 2 – Results of Experiment 1. The detection rule corresponds to the byte fields tested.

Test	Trials	Detection rule[a]	Detection rate ($\bar{x}$)
1	300	`if (All Byte Fields) == Valid:` `¬ misuse-case`	N/A[b]
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 $\bar{x}$	2100	–	[98.3–99.2%]

[a]Conditional constructs.
[b]Packets with valid fields do not violate any misuse cases.

5.2. *Results and analysis of Experiment 2*

The following sections provide the results and analysis of Experiment 2. Experiment 2 is conducted to evaluate the enhancement strategies discussed in Section 4.2.2. The enhancement strategies target previously evaluated packets that are incorrectly classified as normal, specifically the `Pld`. Therefore, all results after re-evaluation are consistent with those in Table 3 with the exception of the mean misuse detection rate for invalid `Pld`s.

After re-evaluation, the mean invalid `Pld` detection rate (Test 7) increases to 95.7%, a 4% improvement of Experiment 1 Test 7. Of the 300 trials, *Enhanced_MBIDS.py* evaluates 25 packets as valid even though all are invalid. If any of the 25 packets have `SourceID 0x01` or `SourceID` equal to `DestinationID`, the packet is logged in a secondary logfile for re-evaluation. The

Table 3 – Results of Experiment 2. The detection rule corresponds to byte fields tested.

Test	Trials	Detection rule[a]	Detection rate ($\bar{x}$)
1	300	`if (All Byte Fields) == Valid:` `¬ misuse-case`	N/A[b]
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 $\bar{x}$	2100	–	[99.0–99.7%]

[a]Conditional constructs.
[b]Packets with valid fields do not violate any misuse cases.

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 `SourceID` and `DestinationID` or a packet with `SourceID` `0x01` that was actually sent by the Z-Wave gateway controller as misuse cases. The improved results of Experiment 2 are listed in Table 3.

6. Conclusion and future work

6.1. Conclusions

This section discusses how the hypotheses are supported and research goals are achieved.

6.1.1. 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 (International Telecommunication Union – Telecommunication Standardization Sector G.9959, 2015) and open source documentation including OpenZWave (OpenZWave, 2014) and the RaZberry Pi documentation (RaZberry Project, 2015). Misuse cases are determined for 11 of the known Z-Wave `CmdCls` and their corresponding `Pld` parameters. This allows the engineering of conditional constructs used for packet evaluation. The first goal is accomplished toward achievement of subsequent goals since misuse case identification is needed before system development and enhancement.

6.1.2. Goal 2: enhanced MBIDS

The second 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 `SourceID` and `DestinationID`, 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 `SourceID` is `0x01`, the packet is re-evaluated. If it is not in the Z-Wave gateway logfile, it is reclassified as a misuse case.

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 (`Len`, `SourceID`, `DestinationID`, `CmdCl`, and `Cmd`) and 95.7% mean detection rate for packets with invalid `Pld`s. Therefore, the overall efficiency of the MBIDS given the mean detection rate of 100% for five of the evaluated fields and a 95.7% mean detection rate for one evaluated field results in an overall mean detection rate of 99%. Because detection accuracy has increased with the extended MBIDS, the second research goal is achieved.

6.1.3. Goal 3: MBIDS cost reduction

The original MBIDS uses a USRP N210 as the underlying packet capture device. At approximately $2400 USD, the USRP N210 causes the MBIDS to be expensive to implement. Although the USRP N210 achieves a mean packet capture accuracy of 94%, a low-cost solution is more practical for MBIDS employment. The HackRF One not only maintains a similar packet capture accuracy of the USRP N210, but only cost $300 USD, resulting in a 87% savings while maintaining system accuracy. Thus, our final research goal is achieved.

6.2. Significance of research

This research demonstrates an effective method to detect Z-Wave network attacks. The enhancements extend and improve the system in Fuller et al. (2016) by increasing the mean detection accuracy. The enhancements also enable the MBIDS to detect spoofed controllers and rogue devices. By designing the system to operate on any computer that supports Python 2.7, it can be implemented using any SDR configured for Z-Wave packet capture. Using the HackRF One reduces implementation cost by up to 87% while maintaining relatively similar packet capture accuracy as the more expensive USRP N210. 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, any failure will have no negative effects on network performance. The extensible framework allows the MBIDS to work with any underlying SDR. This allows lowering the price by 87%, significantly reducing the cost of employment. All tools and source code are open source and available at https://github.com/AFITWiSec.

6.3. Future work

The next logical step for this research is to include the evaluation of remaining `CmdCls` and their corresponding `Pld` parameters.

Secondly, an advanced adversary capable of compromising a Z-Wave gateway intends to execute their attack covertly. The attacker may use the encrypted command class to evade detection. At this point in the MBIDS evolution, the attacker would successfully exploit the Z-Wave network and remain undetected. As demonstrated in Badenhop and Ramsey (2016) and Badenhop et al. (2016), an attacker can extract the Z-Wave encryption keys from physically-accessible devices. Using the encryption keys, the attacker can intercept and transmit encrypted traffic. To decipher encrypted attacks, the user can leverage the `decryptPCAPNG` tool to decrypt Z-Wave encapsulated frames (Badenhop and Ramsey, 2016) and accurately determine the intent of the encrypted packet. However, an attacker can use their own encryption key to inject encrypted traffic into the target network. Although the end user will not be able to decrypt the injected packets, the use of an unknown key will immediately produce a misuse case and alert the user to a potential malicious transmission. Similarly, if an attacker attempts to overwrite the encryption key to control a secured Z-Wave device (Fouladi and Ghanoun, 2013), the new encryption key is considered AN unknown key and will trigger a misuse case. The ability to detect AES encrypted covert channels using methods and tools

in Badenhop and Ramsey (2016) will create a more robust MBIDS.

Thirdly, the 87% in cost reduction while using the HackRF One is significant, but further reduction is possible using the YARD Stick One (developed by the same manufacturers as the HackRF One). Although not an SDR, the YARD Stick One is a computer controlled wireless transceiver. It understands many radio protocols by default and can be programmed to understand Z-Wave. RfCat comes standard with any YARD Stick One and allows a user to control the device from a Python Shell. An implementation of the YARD Stick One to capture Z-Wave traffic is provided in Advens (2015). This implementation can be extended to support the MBIDS.

The YARD Stick One costs 66% less than the HackRF One and is 25% smaller. After programming the YARD Stick One to understand Z-Wave, the test in Section 4.1.3 is repeated. The YARD Stick One achieves a mean packet reception rate of ≈94% [90.16–96.43] (n = 500, CI = 99%, p-value <0.01). Comparing its packet capture accuracy with previously tested SDRs shows no statistically significant difference in their mean reception rates (Fig. 6). Thus, implementing the MBIDS with the underlying YARD Stick One would achieve similar accuracy while lowering cost, based on these early studies.

The RTL2832U is another viable option to lower the cost of the MBIDS. This software defined radio scanner uses a DVB-T TV tuner dongle and cost $20. Although developed mainly for TV operations, it operates in the ISM bands and can be modified to capture and interpret Z-Wave packets.

Another area of future research is using the MBIDS as a framework. Given the feasibility of an MBIDS, vendors partnered with Sigma Designs with access to the Z-Wave SDK 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. Thus, 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 examines the Z-Wave application layer protocol. The potential impact of that the attacks presented herein and the detection method warrant further study into the efficacy of this approach for other WSN protocols.

Acknowledgments

This research is supported in part by the U.S. Department of Homeland Security ICS-CERT. The views expressed in this work are those of the authors and do not reflect official policy of the United States Army, United States Air Force, Department of Defense, or the U.S. Government.

REFERENCES

Advens. Z-Attack, <https://github.com/advens/Z-Attack>; 2015.
Badenhop C, Ramsey B. Carols of the Z-Wave security layer; or, robbing keys from peter to unlock Paul. In: Pocorgtfo `0x12`. 2016 <http://openwall.info/wiki/_media/people/solar/pocorgtfo12.pdf>.
Badenhop C, Fuller J, Hall J, Ramsey B, Rice M. Evaluating ITU-T G.9959 based wireless systems used in critical infrastructure assets. In: Rice M, Shenoi S, editors. Critical infrastructure protection IX, vol. 466. IFIP Advances in Information and Communication Technology. Springer International Publishing; 2015. p. 209–27.
Badenhop C, Ramsey B, Mullins B, Mailloux L. Extraction and analysis of non-volatile memory of the ZW0301 module, a Z-Wave transceiver. Digit Invest 2016;17:14–27.
Barcena M, Wueest C. Insecurity in the Internet of Things, <https://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/insecurity-in-the-internet-of-things.pdf>; 2015 [accessed 06.10.15].
Crowley D, Bryan D, Savage J. Home Invasion V2.0 - Attacking Network-Controlled Hardware, Black Hat Conference, July 2013.
Fouladi B, Ghanoun S. Security Evaluation of the Z-Wave Wireless Protocol, Black Hat Conference, July 2013.
Freescale Beestack: Application Development Guide, <http://cache.freescale.com/files/rf_if/doc/user_guide/BSADG.pdf?fsrch=1>; 2008 [accessed 10.12.15].
Fuller J, Ramsey B. Rogue Z-Wave controllers: a persistent attack channel. In: 10th IEEE international workshop on practical issues in building sensor network applications (SenseApp). 2015a. p. 734–41.
Fuller J, Ramsey B. Stealty and Persistent Access to Z-Wave Gateways, DerbyCon 5.0 “Unity”, September 2015b.
Fuller J, Ramsey B, Pecarina J, Rice M. Wireless intrusion detection of covert channel attacks in ITU-T G.9959-based networks. In: 11th international conference on cyber warfare and security (ICCWS). 2016. p. 137–45.
Goodspeed T, Bratus S, Melgares R, Speers R, Smith S. Api-do: tools for exploring the wireless attack surface in smart meters. In: 45th Hawaii international conference on system science (HICSS). 2012. p. 2133–40.
Hall J. EZ-Wave Software Framework, <https://github.com/AFITWiSec/EZ-Wave>; 2016a.
Hall J. A practical wireless exploitation framework for Z-Wave networks [Master’s thesis]. Air Force Institute of Technology; 2016b.
Hall J, Ramsey B. Breaking Bulbs Briskly by Bogus Broadcast, Shmoocon, January 2016.
Hall J, Ramsey B, Rice M, Lacey T. Z-Wave network reconnaissance and transceiver fingerprinting using software-defined radios. In: 11th international conference on cyber warfare and security (ICCWS). 2016. p. 163–71.
Hormann L, Glatz P, Steger C, Weiss R. Measuring the state-of-charge – Analysis and impact on wireless sensor networks. In: 36th IEEE conference on local computer networks (LCN). 2011. p. 982–5.
International Telecommunication Union – Telecommunication Standardization Sector G.9959. Short range narrow-band digital radiocommunication transceivers – PHY, MAC, SAR and LLC layer specifications, January 2015.
OpenZWave, <http://openzwave.com/>; 2014 [accessed 04.11.15].
Patel H, Temple M, Baldwin R, Ramsey B. Application of ensemble decision tree classifiers to ZigBee device network authentication using RF-DNA fingerprinting. In: 9th international conference on cyber warfare and security (ICCWS). 2014. p. 176–86.

Picod J, Lebrun A, Demay J. Bringing software defined radio to the penetration testing community, Black Hat conference, August 2014.
Ramsey B, Mullins B. Defensive rekeying strategies for physical-layer-monitored low-rate wireless personal area networks. In: Butts J, Shenoi S, editors. Critical infrastructure protection VII, vol. 417. IFIP Advances in Information and Communication Technology. Springer Berlin Heidelberg; 2013. p. 63–79.
Ramsey B, Mullins B, Speers R, Batterton K. Watching for weakness in wild WPANs. In: IEEE military communications conference (MILCOM). 2013. p. 1404–9.
RaZberry Project. Z-Way Developer's Documentation, <http://razberry.z-wave.me>; 2015 [accessed 13.11.15].
Reaves B, Morris T. Analysis and mitigation of vulnerabilities in short-range wireless communications for industrial control systems. Int J Crit Infr Prot 2012;5(34):154–74.
SmartThings API, <https://graph.api.smartthings.com>; 2015 [accessed 10.12.15].
Stankovic J, Wood A, He T. Realistic applications for wireless sensor networks. In: Nikoletseas S, Rolim JD, editors. Theoretical aspects of distributed computing in sensor networks, monographs in theoretical computer science. An EATCS series. Springer Berlin Heidelberg; 2011. p. 835–63.
ZWave Alliance Recommendation ZAD12837-1, <http://z-wavealliance.org>; 2014 [accessed 18.07.16].
ZWave Frequency Coverage, <http://z-wave.sigmadesigns.com>; 2016 [accessed 18.07.16].

Jonathan D. Fuller is a Network Systems Engineer in the United States Army. He earned a MS in computer science from the Air Force Institute of Technology in 2016. His interests include computer and network security, with an emphasis in vulnerability analysis and exploitation of wireless sensor networks.

Benjamin W. Ramsey is an Assistant Professor of Computer Science at the Air Force Institute of Technology. He earned a PhD in computer science in 2014 from the Air Force Institute of Technology. His research focuses on wireless computer network security and critical infrastructure protection. Benjamin teaches graduate courses in secure software design and reverse engineering.

Mason J. Rice is an Assistant Professor of Computer Science at the Air Force Institute of Technology. He earned a PhD in computer science from the University of Tulsa in 2011. His research interests include network and telecommunications security, cyber-physical systems security, and critical infrastructure protection. Mason teaches graduate courses on information warfare and computer network security.

John M. Pecarina is an Assistant Professor of Computer Science at the Air Force Institute of Technology. He earned a PhD in computer science from Texas A&M University in 2013. He specializes in pattern detection, process mining, and image retrieval. John teaches graduate courses on operating systems, distributed systems, and data security.