IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 April 2023, accepted 2 June 2023, date of publication 12 June 2023, date of current version 21 June 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3285476

RESEARCH ARTICLE

ZMAD: Lightweight Model-Based Anomaly Detection for the Structured Z-Wave Protocol

CARLOS KAYEMBE NKUBA[1], (Member, IEEE), SEUNGHOON WOO[1], HEEJO LEE[1], (Member, IEEE), AND SVEN DIETRICH[2,3], (Senior Member, IEEE)

[1]Department of Computer Science and Engineering, Korea University, Seoul 02841, Republic of Korea
[2]Department of Computer Science, Hunter College, City University of New York (CUNY), New York, NY 10065, USA
[3]Department of Computer Science, The Graduate Center, City University of New York (CUNY), New York, NY 10016, USA

Corresponding author: Heejo Lee (heejo@korea.ac.kr)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korea Government Ministry of Science and ICT (MSIT) (Development of SBOM Technologies for Securing Software Supply Chains) under Grant 2022-0-00277, in part by the IITP Grant funded by the Korea Government MSIT (Convergence Security Core Talent Training Business) under Grant 2022-0-01198, in part by the IITP Grant funded by the Korea Government MSIT (ICT Creative Consilience Program) under Grant IITP2023-2020-0-01819, and in part by the Research Foundation City University of New York (RFCUNY).

ABSTRACT Smart home automation is part of the Internet of Things that enables house remote control via the use of smart devices, sensors, and actuators. Despite its convenience, vulnerabilities in smart home devices provide attackers with an opportunity to break into the smart home infrastructure without permission. In fact, millions of Z-Wave smart home *legacy devices* are vulnerable to wireless injection attacks due to the lack of encryption support and the lack of firmware updates. Worse yet, recent Z-Wave *secure S2 devices* with built-in encryption are also vulnerable to specific targeted attacks, *i.e.*, attacking S2 devices is possible via vulnerable legacy devices or injecting malicious unencrypted packets that alter S2 devices normal operations. In this paper, we present ZMAD, a *lightweight* anomaly-based intrusion detection system (IDS) for monitoring and detecting wireless attacks on Z-Wave smart home devices. ZMAD uses a technique called *packet formalization* to address heterogeneous packets coming from various Z-Wave devices. ZMAD also uses a *centralized learning approach* to profile normal communication patterns of devices to increase Z-Wave Command Class coverage. By constructing a lightweight artificial neural network built from scratch in consideration of packet formalization and centralized learning, ZMAD can effectively detect abnormal behaviors in Z-Wave networks and runs on an external device to avoid network overhead. We applied ZMAD to an evaluation testbed constructed using 17 top-rated real-world Z-Wave smart home devices. From our experiments, we confirmed that ZMAD could effectively discover wireless injected packets with an accuracy of 98% for its artificial neural network. Our further analysis demonstrated that ZMAD is more effective than existing approaches, increasing the coverage of Z-Wave Command Classes by 663% while reducing five to 47 times the size of the trained model (23.1 KB) compared to existing deep learning architectures.

INDEX TERMS Internet of Things, smart home security, Z-Wave, intrusion detection systems, artificial neural network.

I. INTRODUCTION

Recently, Z-Wave [1] has become one of the popular protocols used in smart home automation systems (HAS) and multi-dwelling units (MDU) due to its advantages such as low power usage and backward compatibility with legacy devices [2]. However, because of its high demand and proprietary nature, manufacturers tend to focus more on device functionality rather than device security [3]. Consequently, various security vulnerabilities have emerged in Z-Wave

The associate editor coordinating the review of this manuscript and approving it for publication was Cong Pu.

smart home devices (*e.g.*, [3]). By exploiting vulnerable devices, attackers can remotely control them, deny their service, or deplete their batteries [10].

Millions of Z-Wave *legacy* devices, based on 100, 200, 300, and some 500 series chipsets [11], [12], are still in use in many smart homes. These devices are vulnerable to the aforementioned attacks because they do not support data transport encryption. Worse yet, one-time-programmable (OTP) memory makes it infeasible to update the firmware on these devices to counter such threats, making it difficult to prevent or mitigate potential threats. In addition, recent Z-Wave higher-security devices called security 2 (S2) (some 500 and 700 series chipsets) with built-in AES-128 [13] encryption are still vulnerable to specific *unencrypted targeted attacks* that are exploited either through other vulnerable legacy devices (see Section III-A) or via *malformed* ***unencrypted*** *network operation traffic* [10]. Unfortunately, smart homeowners are less likely to change their well-functioning legacy devices due to the lack of information on their vulnerabilities, technical migration issues, and the higher cost of recent devices [14].

One effective way to circumvent these attacks is to leverage a lightweight intrusion detection system (IDS) for Z-Wave smart home IoT devices. Using such an IDS, the security of the smart home can be enhanced by monitoring packets transmitted within the smart home in real-time and then filtering out abnormal packets. However, precisely detecting abnormal behaviors is becoming challenging. Existing TCP/IP-based IDS (*e.g.*, Snort [15]) cannot be directly applied to the Z-Wave network, because Z-Wave smart home systems use a different communication protocol with its own specification. In addition, a Z-Wave network can have multiple devices that implement unique functionalities depending on their type (*e.g.*, controller, motion sensor, wall plug, and door lock). Therefore, packets transmitted within the Z-Wave smart home network are heterogeneous, which undermines the detection accuracy of an IDS. Moreover, to deploy a real-time IDS in a smart home network, a lightweight and high-performance IDS system is required; an IDS should have a low implementation and computation cost to seamlessly integrate into the smart home environment, including possible deployment on single-board computing devices.

A. LIMITATIONS OF EXISTING APPROACHES

Existing techniques for the Z-Wave protocol security (*e.g.*, [8], [16], [17]) hardly consider the entire characteristics of the Z-Wave specification, thereby resulting in limited intrusion detection accuracy. For example, MBIDS [8] attempted to detect Z-Wave attacks by leveraging manually the misuse-based detection technique. However, they could examine only 11 out of 100+ Z-Wave Command Classes (`CMDCLs`), which is a set of commands and responses related to certain functionality of a device. Hence, they may report false negatives due to the low `CMDCLs` coverage (see Section VI-C2) and lack the ability to detect unknown attacks. Other approaches [16], [17] that are fingerprinting-based can only be used to identify Z-Wave device model, make, and type. Therefore, to better defend Z-Wave networks from known and unknown wireless injection attacks, an effective IDS technique should be devised. Moreover, with the advancement of deep learning algorithms, they could be leveraged to detect new attacks in comparison to traditional signature-based IDSes.

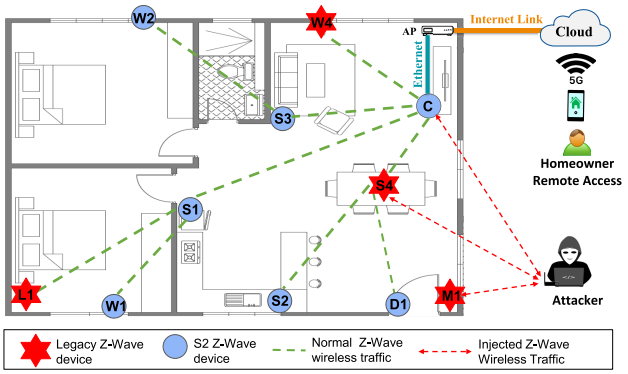


FIGURE 1. Wireless injection attacks on Z-Wave home network consisting of several smart devices: C (controller), D (door lock), L (LED Light), M (motion sensor), S (plug/outlet), and W (Windows contact sensor).

B. OUR APPROACH

We present **ZMAD** (**Z**-Wave **M**odel-based **A**nomaly **D**etection), a lightweight anomaly-based intrusion detection system for detecting external wireless attacks on Z-Wave smart home networks with selective countermeasures. ZMAD generates a new lightweight *artificial neural network* (ANN) in consideration of the Z-Wave protocol semantics and structure and runs on an external device to avoid the network overhead. ZMAD provides the total control of all implemented modules and procedures to IDS operators, with the possibility of rapid updates to new Z-Wave specifications in the future, while providing a clear understanding of the prediction process and a possibility of detecting new future attacks that deviate from the normal network profile.

We aim to devise a system that effectively detects abnormal Z-Wave packets. The two main key techniques of ZMAD, which is significantly distinguishable from existing approaches, are *packet formalization* and *centralized learning*. To address the heterogeneity of Z-Wave devices, ZMAD formalizes packets obtained from various Z-Wave devices by considering only the core fields of the packets that were frequently used in Z-Wave network attacks. This allows ZMAD to process various types of Z-Wave packets, and also to focus on the important fields that are effective in detecting attacks. Thereafter, ZMAD trains its ANN using the normal Z-Wave packets collected from real-world devices and abnormal packets from known vulnerabilities and fuzz testing (see

Section V-A) and generates a lightweight centralized model for detection. The collected packets are preprocessed, filtered, and normalized in order to profile the normal behavior of Z-Wave devices.

We evaluated ZMAD using 17 real-world Z-Wave devices. In our experiments, we confirmed that ZMAD can distinguish between normal and abnormal packets with **98%** accuracy while increasing the Z-Wave Command Classes coverage from 11 to **73** `CMDCLs` compared to an existing MBIDS approach. When we compared ZMAD to various deep learning ANNs (*e.g.*, LSTM and MLP) on *not formalized* packet *raw* data, we observed that ZMAD could classify abnormal packets with higher detection accuracy (see Table 5, Figure 12) while generating a much smaller-sized learning model, demonstrating the effectiveness of ZMAD in the perspective of lightweight and accurate Z-Wave smart home IDS.

This paper makes the following contributions:

- We present ZMAD, which is an IDS framework that discovers wireless attacks on smart home devices. The core element of ZMAD is its new lightweight artificial neural network built from scratch for the Z-Wave protocol.
- ZMAD works as an effective lightweight IDS for Z-Wave smart homes by using packet formalization specific to the Z-Wave protocol and showing high anomaly detection accuracy.
- A systematic experimental analysis of ZMAD effectiveness on a smart home network, consisting of 17 real Z-Wave devices, shows that it is effective in detecting abnormal wireless attacks with an accuracy of 98%. ZMAD also showed higher anomaly detection accuracy than widely used ANN architectures on *raw data*.
- We provide our datasets of real-world Z-Wave device traffic that can be reused by other researchers to foster future research [18].

II. RELATED WORK

A. RESEARCH ON THE Z-WAVE PROTOCOL

There have been many studies to analyze the Z-Wave protocol (*e.g.*, [3], [4], [5], [6], [7], [8], [9], [16], [17], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28]); among these, some focused on the analysis of the Z-Wave traffic. MBIDS [8] analyzed 11 Z-Wave CMDCLs to find misuse cases. Z-IoT [16] performed a passive fingerprinting of the Z-Wave traffic to identify Z-Wave device types without analyzing any application payload. Others [17] sniffed Z-Wave traffic to retrieve the device make and model.

However, these approaches are not suitable for fully addressing our target problems. Owing to the low scope of `CMDCLs`, the study by Fuller et al. cannot detect unknown vulnerabilities that exploit uncovered CMDCLs. Moreover, attackers can exploit other Z-Wave packet fields *e.g.*, `P1`, `P2`, `SRC`, `DST` fields to conduct a routing attack on the Z-Wave network [3]. Therefore, our study formalizes the Z-Wave packets and covers not only more Z-Wave `CMDCLs` to detect Z-Wave attacks, but also other core packet fields that may be misuse by an attacker.

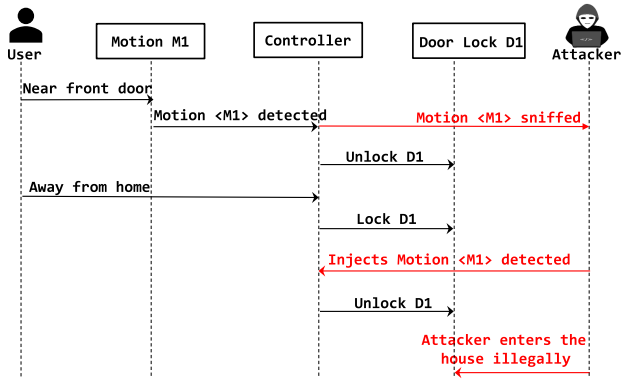


FIGURE 2. Attacker scenario.

B. SECURITY RESEARCH ON IOT

Some previous approaches focused on finding smart home intrusions while examining devices that communicate using the TCP/IP protocol [29], [30], [31], [32], [33], [34], [35]. For example, Nguyen et al. [33] and IoTSentinel [35] assessed IoT devices IP network traffic for anomalies. These solutions are efficient to examine devices that communicate over the IP protocol. However, they cannot catch specialized Z-Wave wireless attacks, which directly target a vulnerable device, because they rely on TCP/IP traffic routed by the controller for analysis.

Some other approaches can be used to detect vulnerabilities in the firmware of IoT devices (*e.g.*, vulnerable code detection techniques [36], [37], [38], [39]). However, most Z-Wave devices are blackboxes, and abnormal behavior can attack beyond the device's own vulnerabilities; therefore, they are not capable of detecting anomalies in Z-Wave networks. Meanwhile, ZMAD can directly inspect Z-Wave traffic for anomalies in the air, and also it can be easily integrated into the smart home networks.

III. THREAT MODEL AND CHALLENGES

In this section, we describe the threat model associated with vulnerable smart home devices, present the challenges of our work, and then clarify the goal of this paper.

A. PROBLEM STATEMENT AND THREAT MODEL

1) PROBLEM STATEMENT

A vulnerability report by [10] found that Z-Wave *legacy* and *recent S2* devices are vulnerable to impersonation, replay, and denial of service (DoS) attacks. Referring to this, we clarify the threat model considered in this paper as follows:

1) We consider a smart home network consisting of several heterogeneous devices types (see Figure 1) from different manufacturers (*e.g.*, smart controller, door lock, and motion sensor) configured with scenes,

routines, and automation in an *If This Than That* (IFTTT) settings [40].

2) We assume that a typical Z-Wave smart home has a combination of legacy devices and new Z-Wave devices implementing recent S2 AES encryption.

3) Users are unaware of the security of legacy devices, as it is not stated in the device documentation.

Referring to the scenario depicted in Figure 2, a typical IFTTT automation could be structured as follows: if motion sensor M1 captures human presence near the entrance door, then the controller automatically sends a packet that opens the door lock D1 to allow the user to go outside. Here, we assume that D1 uses the latest secure S2 communication encryption and M1 is a simple *legacy* motion sensor device that does not implement any data encryption; however, M1 is involved in the controller automation to open D1.

2) ADVERSARY

An attacker could be located near the house (∼100m) and sniff the traffic of M1 (see Figure 2). Then, he can generate a fake non-encrypted packet (motion detection events) with a payload of M1 [3], [10]. Thereafter, the attacker can send this malicious unencrypted packet to the controller, which will validate its payload and unlock D1; therefore, allowing the attacker to access illegally the house during the daytime.

3) SPECIFIC TARGET ATTACK ON S2 DEVICES

Even if some recent S2 Z-Wave devices support encryption, attackers can manipulate them through *specific targeted attacks*. Let D1 be a Z-Wave S2 device that supports encryption. In this case, it is difficult for the attacker to directly attack D1, and thus it is difficult to manipulate the door. However, if the motion sensor M1 is a legacy device, an attacker can attack M1 as a substitute for D1. In other words, the attacker manipulates M1 to fool the controller that a human motion was detected near the door (inside the home), and thus manipulates the door D1 to automatically open. Additionally, an attacker can send malicious unencrypted transport packets causing a denial of service (DoS) [10], [41] to the main S2 controller, which will not be able to notify, via mobile app, the remote homeowner of any intrusion. Hence, this DoS attack on the controller allows the attacker to break into the house via any windows W1 and W2, or door D1. Moreover, additional Z-Wave specialized attacks on legacy and S2 devices are listed in Table 3.

4) ATTACK IMPACT

The lack of security in the HAS can lead to additional threats such as privacy breaches and safety violations of homeowners, device misuse, and office and residential burglary. Lack of security in smart home devices can also cause a safety threat on smart home residents; attackers can remotely control vulnerable smart devices and gain illegal access to their homes at will. Moreover, an attacker can misuse vulnerable devices, *e.g.*, remotely starting smart heater and air conditioner devices to increase energy bills or create house damage. Also, attackers can harass the user by turning on alarm systems and lights at night.

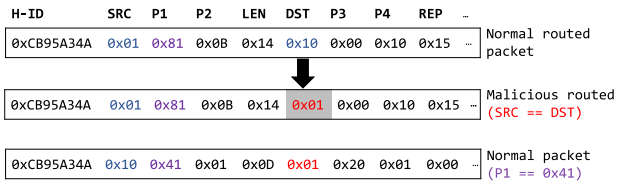


FIGURE 3. Routing attacks using valid P1 = 0 × 81 but same SRC and DST values (0 × 01).

5) DEFENSE

An ideal IDS should detect anomalous injected packets, raise an alert in case of a valid intrusion, and take specific countermeasures such as activating alarm systems and disabling vulnerable devices *e.g.*, turning off remotely the smart outlet connected to the vulnerable device.

B. CHALLENGES

Several existing approaches (*e.g.*, [8], [42]) attempted to provide a solution to monitor Z-Wave wireless traffic. However, they are limited in detecting malicious attacks owing to the following challenges of applying anomaly detection systems to the Z-Wave IoT networks.

C1. Covering all supported Z-Wave command classes per target network. The Z-Wave protocol specification defines more than one hundred command classes, which are a set of commands and responses related to a particular functionality of a device. To analyze packets and find abnormal behavior effectively, such CMDCLs should be covered as much as possible per target Z-Wave network. However, existing MBIDS rule-based approaches only cover a small portion of them (*e.g.*, 11 CMDCLs [8]) owing to the manual and complex coding of "*if*" conditions to verify their validity. Therefore, we need a comprehensive approach that can cover all the supported CMDCLs per target Z-Wave network.

C2. Handling heterogeneous Z-Wave packet types. Z-Wave networks may have several devices that implement distinctive features according to their types (*e.g.*, controller, switch, motion sensor, alarm, door lock). Moreover, Z-Wave devices can communicate using singlecast, multicast, broadcast, and routed transmission modes. Manually defining these device features and transmission types is complex and error-prone. Therefore, an IDS that can address distinct types of packets simultaneously and precisely is required.

C3. Addressing Z-Wave semantically structured packets. Z-Wave packet fields do carry a specific semantic according to their position in the packet. The validity and semantics of a field depend on other neighbor fields. For

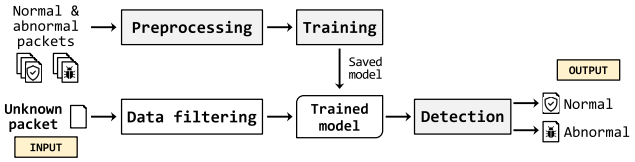


FIGURE 4. High-level workflow of ZMAD.

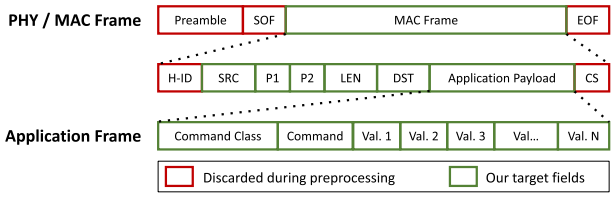


FIGURE 5. Illustration for the Z-Wave packet structure.

example, in the case of the routing attack [3] shown in Figure 3, two packets with the same value of 0×01 in `DST` field position can be either a normal or a malicious packet depending on the value in `SRC`. Therefore, an IDS system that can detect attacks according to the *semantics* of each field is required.

C4. Constructing lightweight, fast, and maintainable IDS. To minimize overhead while operating in real-time, we need a lightweight and fast IDS. However, the existing rule-based or deep learning model-based IDSes are hardly lightweight because they are complex and contain many unnecessary functions, which are blackboxes to the IDS operator who does not understand their internal decision making process.

C. GOAL STATEMENT

Our goal is to investigate, detect, and mitigate attacks that affect the Z-Wave smart home network. In this regard, a suitable anomaly detection system is required to overcome the aforementioned challenges and also to provide mitigation towards wireless attacks targeting the Z-Wave legacy devices. Specifically, the IDS we consider should be lightweight and fast, as well as considering all Z-Wave command classes per target network, heterogeneous packet types, and semantically structured packets.

IV. METHODOLOGY

Here we present the methodology of ZMAD, which is an IDS that detects external wireless attacks for Z-Wave smart home networks.

A. DESIGN CHOICES

ZMAD utilizes a novel ANN built and tailored on the Z-Wave specification requirement to detect abnormal wireless packets that deviate from the normal network behavior. Instead of considering Z-Wave's numerous CMDCLs one by one (C1 in Section III-B), we decided to extract patterns and respond to them with a learning model. In addition, instead of using one device for model training used in other approaches [16], [43], we consider a *centralized learning* approach for network traffic from various Z-Wave devices. This decision allows us to train our model on any possible command classes used in the monitored Z-Wave network.

In feature selection, simply extracting fields from Z-Wave packets hinders the model accuracy owing to the heterogeneity of Z-Wave packets (C2 in Section III-B). Hence, ZMAD uses a technique called *packet formalization*. ZMAD focuses only on fields that can be effectively used for abnormal behavior detection, and normalizes various types of packets as inputs of the same length so that the learning model can easily proceed. Moreover, the semantically structured Z-Wave packets used in the known attacks are also included in the training data of ZMAD (C3 in Section III-B). Finally, to avoid the possible network overhead, we decided to run ZMAD on an external device (C4 in Section III-B), as Z-Wave IoT devices are resource-constrained with low memory and processing power.

B. ZMAD OVERVIEW

ZMAD has the following three phases: preprocessing, training, and detection. Figure 4 describes the high-level workflow of ZMAD. In the preprocessing phase, given normal and abnormal Z-Wave packets in the dataset (see Section V-A), ZMAD formalizes them by considering only their core fields, in order to address the heterogeneity of Z-Wave devices. Then with the formalized packets, ZMAD trains its ANN and generates model during the training phase. Finally, in the detection or deployment phase, ZMAD assesses incoming Z-Wave packets for anomalies utilizing the trained model.

C. PREPROCESSING PHASE

Given various types of normal and abnormal Z-Wave packets, ZMAD normalizes the packets for training the learning model. The process of constructing a dataset by collecting normal and abnormal Z-Wave packets is introduced in detail in Section V-A. We first introduce the Z-Wave packet structure.

1) INTRODUCTION TO Z-WAVE PACKET FIELDS

A Z-Wave legacy packet consists of fields that determine the action that should be taken by the receiver. Figure 5 illustrates the Z-Wave packet structure, which encompasses the following fields.

1) `PRE`: Preamble used for synchronization at receiver.
2) `SoF`: Start of frame.
3) `H-ID`: Home ID value of the Z-Wave home network.
4) `SRC`: The sender ID.
5) `Frame control` (`P1` & `P2`): Packet type and routing information.
6) `LEN`: Packet length.
7) `DST`: Target destination ID.

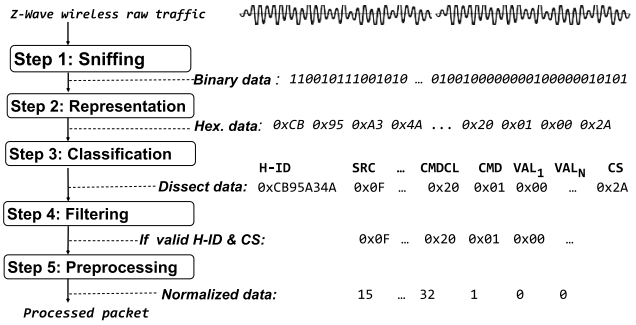


FIGURE 6. ZMAD packet formalization illustration.

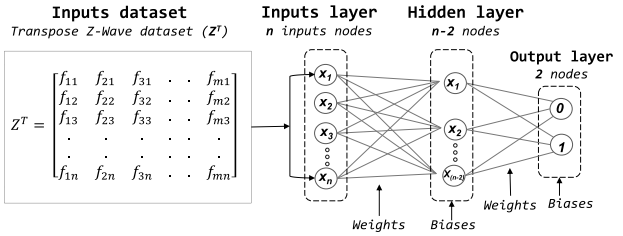


FIGURE 7. ZMAD ANN architecture.

8) Command Class (CMDCL): Upper Command Class related to certain functionality of a device.
9) CMD: Command that the receiver should run.
10) VAL: Command payload values *e.g.*, turn on/off a device.
11) CS: Checksum of the Z-Wave packet.
12) EoF: End of frame.

2) PACKET FORMALIZATION

ZMAD discards the fields that have less impact on causing bugs on a target device and leaves only the core fields. First, we decided that application frame fields have high importance because they have a payload that actually controls the device, *e.g.*, "CMDCL $= 0 \times 25$, CMD $= 0 \times 01$, VAL1 $=$ 0xFF" will turn on the target switch device. Hence, ZMAD retains the "MAC Frame" field that contains the application payload, while discarding noise data, the preamble, the start of frame (SoF), and the end of frame (EoF) from the packet, which have identical values and are not directly used to exploit target devices.

ZMAD then examines the retained fields. Here, ZMAD first evaluates if the packet home ID (H−ID) is the same as the one our IDS is monitoring. If it is the same, then ZMAD computes the CS to verify if the packet is complete. If the H−ID and CS are valid, then ZMAD drops these fields and focuses on the remaining important fields that may cause an anomaly on the device. Other fields in the MAC frame are important as attackers can manipulate transport fields (P1 and P2) to inject fake routing tables into the controller and devices, *e.g.*, a packet with "SRC 0×01 and DST 0×01" with malicious routing payload in P1 and P2 can cause a denial of service on the controller (see Figure 3) [10]. Note that the total packet length, excluding the PRE, SoF, EoF, is 64 bytes; in our dataset, we also observed that the maximum length was 30 bytes. Hence, external packets exceeding the valid length could be either dropped or flagged as abnormal.

Finally, the remaining application payload fields are crucial as their role is to control the target device operations. Attackers can misuse them and control legacy devices remotely as they do not employ data confidentiality, integrity, and authentication. Moreover, an attacker can inject malicious unencrypted packets to secure S2 devices in order to alter their normal network operation such as changing their internal routing table information, manipulating their S2 NONCE exchange, etc [10]. Therefore, specialized *unencrypted* targeted attacks can affect not only *legacy* Z-Wave devices, but also *recent S2* ones.

D. TRAINING PHASE

Figure 6 represents ZMAD packet formalization on Z-Wave protocol. As previously introduced, H−ID and CS fields are checked and removed after ZMAD validates them in consideration of the monitored network ID. Packet formalization of ZMAD also contributes to increase abnormal behavior detection accuracy. This is because ZMAD can train our model more precisely by removing the fields that are unnecessary for attacks (*e.g.*, Home ID), but exist in both normal and abnormal packets (see Section VI-A3).

One thing to note is that Z-Wave devices can transmit packets of different lengths. When each field of a packet is viewed as one dimension, packets of the same length are needed for standard ANN model training, thus ZMAD unifies the length of all packets contained in the dataset by using zero padding. Based on the packet having the longest length in the dataset, the remaining short-length packets are filled with 0 values as much as the insufficient length. Finally, ZMAD converts the value of each field of the packet into an integer and trains the model using the formalized Z-Wave packets.

1) ZMAD CUSTOM-BUILT ANN

We can use existing machine learning (ML) and deep learning (DL) models; however, they contain numerous functions and modules, and do not provide a concise understanding of the internal prediction process to the IDS operator. Moreover, the size of the trained models from existing ML and DL libraries is also excessively large compared to the generated model from a custom ANN that targets only Z-Wave (see Section VI-C1). During deployment phase, trained models from ML & DL require installation of extra large libraries and dependencies in order to run (see Table 8). Trained models size, libraries, and dependencies are important for the future development of TinyML [44] on micro-controllers units (MCUs) that have very limited computing (1MHz - 400MHz), memory (2 - 512KB), storage (32KB - 2MB), and power (150 μW - 23.5mW) compared to traditional computer

with high performance (1GHz - 4GHz; 512MB - 64GB; 64GB - 4TB; 30W - 100W; and GPUs) [45]. Therefore, we create a new neural network model from scratch. Lastly, in case of a Z-Wave specification update a customized ANN will provide a fast update to fit any new requirements.

2) REPRESENTATION OF A PACKET RECORD IN THE DATASET

To train a model, ZMAD represents a Z-Wave packet instance (p) in the dataset as follows.

$$p = f_1, f_2, f_3, f_4, \ldots\ldots, f_{n-1}, f_n \tag{1}$$

where f_i denotes each field in the packet.

A Z-Wave dataset is a collection of Z-Wave instance packets that all share common fields length after packet formalization preprocessing. Hence, a dataset can be represented by a matrix where rows represent the total number of instances of the data while columns indicate the total number of features of the dataset. Therefore, Equation (1) can be expanded as follows.

$$P_i = f_{i1}, f_{i2}, f_{i3}, f_{i4}, \ldots\ldots, f_{i(n-1)}, f_{in} \tag{2}$$

where P_i represents i^{th} instance (*i.e.*, packet) of the dataset and f_{ij} indicates j^{th} packet field in the i^{th} instance.

The final Z-Wave network packet dataset Z will be represented as $m \times n$ order matrix where m represents the total number of instances and n represents the total number of attributes as follows:

$$Z = \begin{bmatrix} f_{11} & f_{12} & f_{13} & \cdots & f_{1n} \\ f_{21} & f_{22} & f_{23} & \cdots & f_{2n} \\ . & . & . & & . \\ . & . & . & & . \\ f_{m1} & f_{m2} & f_{m3} & & f_{mn} \end{bmatrix} \tag{3}$$

For ANN processing, each Z-Wave packet field should be assigned to the input layer in order. Hence, ZMAD transposes the matrix, which can be represented as follows.

$$Z^T = \begin{bmatrix} f_{11} & f_{21} & f_{31} & \cdots & f_{m1} \\ f_{12} & f_{22} & f_{32} & \cdots & f_{m2} \\ f_{13} & f_{23} & f_{33} & \cdots & f_{m3} \\ . & . & . & & . \\ . & . & . & & . \\ f_{1n} & f_{2n} & f_{3n} & \cdots & f_{mn} \end{bmatrix} \tag{4}$$

3) ZMAD ANN ARCHITECTURE

Regarding the Z-Wave dataset, our ZMAD neural network uses three layers: the input layer, one hidden layer (for lightweight and ease of upgrade reasons), and an output layer. Figure 7 illustrates the ZMAD ANN architecture.

- X: The input layer contains as many nodes as the fields in the packet (in this case, n nodes).
- One hidden layer contains $n - 2$ neurons.
- y : The output layer contains two neurons as we have two output classes (normal and abnormal).

Algorithm 1 ZMAD ANN Training Algorithm.

Input : X /* Datasets for training */
Y /* X records labels (0 or 1) */
α /* Learning rate */
N /* #Training iterations */
n /* Length of Z-Wave packet in X */

Output: $W1, W2$ /* Weights of trained model */
$b1, b2$ /* Biases of trained model */
ACC /* Accuracy list of trained model */

procedure ZMAD(X, Y, α, N)
  /* Initialize parameters with random values */
  /* hidden layer: $n-2$ neurons and n inputs */
  $W1 \leftarrow random$(n-2,n)
  $b1 \leftarrow random$(n-2, 1)
  /* outputlayer:2 neurons and $n-2$ inputs from previous layer */
  $W2 \leftarrow random$(2,n-2)
  $b2 \leftarrow random$(2, 1)
  $epochs \leftarrow N$
  $ACC \leftarrow []$
  while $epochs \neq 0$ **do**
    /* Do Forward propagation */
    $F1, A1, F2, A2 =$ ForwardProp($W1, b1, W2, b2, X$)
    /* Do back propagation */
    $dW1, db1, dW2, db2 =$
     BackProp($F1, A1, F2, A2, W1, W2, X, Y$)
    /* Update the parameters */
    $W1, b1, W2, b2 =$
     UpdateParam($W1, b1, W2, b2, dW1, db1, dW2, db2, \alpha$)
    $predictions =$ GetPrediction($A2$)
    $accuracy =$ GetAccuracy($predictions, Y$)
    $ACC.append(accuracy)$
    $epochs = epochs - 1$
  return $W1, b1, W2, b2, ACC$

Algorithm 1 summarizes the whole process of ZMAD model training. Below we list the major processes.

a: FORWARD PROPAGATION (FP)

ZMAD first computes the dot product of the inputs X and random weights W, then adds a random bias b term. This can be expressed as follows.

$$F[1] = X \cdot W[1] + b[1] \tag{5}$$

ZMAD then puts the weighted sum obtained in step one through an activation function. An activation function defines the output of a neuron or node given its inputs. This function helps artificial neural networks learn complex patterns in data. ZMAD uses the rectified linear unit (`RELU`) activation function defined as $RELU(x) = max(0, x)$. Therefore, the activation of Equation (5) is as follow:

$$A[1] = RELU(F[1]) \tag{6}$$

Thereafter, ZMAD computes the dot product of the previous hidden layer and weights, and then adds a bias term.

$$F[2] = A[1] \cdot W[2] + b[2] \tag{7}$$

Finally, ZMAD applies the `Sigmoid` activation function to the output layer. The Sigmoid function is defined as $SIG(x) = \frac{1}{1+e^{-x_i}}$.

$$A[2] = SIG(F[2]) \tag{8}$$

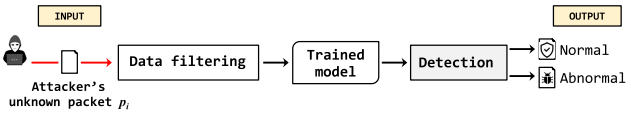


FIGURE 8. Detection phase of ZMAD.

This step is done for all the layers of the ANN and the prediction result output can be obtained from the output layer.

b: BACK PROPAGATION (BP)

If the computed prediction output during forward propagation is different from the actual packet label, an error occurs. Therefore, ZMAD needs to update the initial random weight and bias values based on the error. This process is called back propagation (BP) [46]. During BP, the ZMAD ANN determines the output's loss (or error) and then propagates it back to the neural network. Consequently, this process updates the weights per neuron in order to minimize the error. ZMAD calculates the loss or error by subtracting predicted and actual outcome through derivative (d) of each parameters.

c: PARAMETER UPDATES

The last step is to update the parameter by multiplying gradient of weights with a learning rate α and then subtracting the results from the current weights.

E. DETECTION PHASE

After training, ZMAD saves the trained model. During the detection phase (see Figure 8), the model is loaded into the detection API function for intrusion detection. Suppose ZMAD checks whether an incoming packet p_i intends abnormal behaviors on a target device D_1. ZMAD first determines if p_i needs to be filtered (*e.g.*, targeting same `H-ID` as the monitor network) as in the pre-processing step. If so then ZMAD performs packet formalization. Then the detection API analyzes the packet using trained ZMAD ANN model parameters. ZMAD decides p_i is an abnormal packet when the prediction output is 1. To reduce false alarms, ZMAD further validates the result: first by a ping test to assess whether the affected target device D_1 is running. If the device does not respond, ZMAD should take optional countermeasures, such as turning on sirens and lights. ZMAD could also send a packet to the controller, which will activate the *intrusion scene* that will remotely notify the homeowner via his mobile app. Alternatively, ZMAD can turn off the main smart outlet that supplies power to the affected device.

V. EXPERIMENTAL SETUP

This section introduces the experimental setup of ZMAD, including datasets and evaluation metrics.

A. DATASETS

To construct the dataset, we selected various Z-Wave devices from different vendors based on their popularity and usage. Table 1 summarizes the selected 17 devices. To have a real environment and a representative testbed, we built a real initial Z-Wave network with several devices (Dev1 to Dev14) and set them up in a home environment. Later, devices Dev15, Dev16, and Dev17 were added to test ZMAD model retraining and detection effectiveness in case of update of the initial network. A Z-Wave device supports few operations (see Table 2); therefore, the traffic profile per device can be recorded in less than 10 min. However, we collected extra traffic for 10 days to fully cover all the normal allowed traffic of devices in terms of human interactions and responses to pre-configured automation and scenes on the Z-Wave controller. We collected Z-Wave wireless traffic using Zniffer [47] tool. The collected Zniffer packets were converted to CSV format for data pre-processing and model training.

TABLE 1. Test device overview.

Num.	Device type	Vendor	Model	Chipset
Dev1	Controller	SiLabs	UZB-7	700
Dev2	Controller	Samsung	ET-WV520	500
Dev3	Door Lock	Schlage	BE469ZP	500
Dev4	Contact Sensor	Aeotec	ZWA008	500
Dev5	Contact Sensor	Linear	WADWAZ-1	300
Dev6	Motion Sensor	Aeotec	ZWA005-A	500
Dev7	Motion Sensor	Aeotec	ZW100-A	500
Dev8	Motion Sensor	Linear	WAPIRZ-1	300
Dev9	Siren	Aeotec	ZW164-A	500
Dev10	Siren	Dome	DM501	500
Dev11	Smart Plug	Zooz	ZEN20	500
Dev12	Smart Plug	Fibaro	FGWPB-111	500
Dev13	Smart Plug	Jasco	ZW4201	500
Dev14	Smart Light	Linear	LB60Z-1	500
Dev15	Controller	Samsung	STH-ETH200	500
Dev16	In-wall Switch	Jasco	ZW42003	500
Dev17	Contact Sensor	Linear	WADWAZ-1	300

TABLE 2. Supported operations per device.

Device Type (#)	Supported User Operations
Controller (3)	Inclusion, exclusion, device status reading, device control
Door Lock (1)	Lock, unlock, read battery level, set key
Contact Sensor (3)	ON, OFF
Motion Sensor (3)	Trigger presence ON or OFF
Siren (2)	Turn ON, turn OFF
Smart Plugs (3)	ON, OFF, meter reading
Smart Lights (1)	Turn ON, turn OFF, adjust brightness
In-wall Switch (1)	ON, OFF, report state

1) NORMAL DATASET

The dataset of normal (positive) traffic is obtained from the normal communication of the devices. Z-Wave devices in our environment only use specific commands per device type (*e.g.*, turn lights on or off and get device status in the case of smart switches). Table 2 lists the user operations supported by each device in the testbed. In particular, we used the controller smartphone app (SmartThings [48]), as is commonly used by users, to activate the device and collect the generated packets. The same operation was repeated 10 times per device to vary the device activity. We collected 158,110 normal packets over

TABLE 3. Detection scope of ZMAD (S = Z-Wave specialized attacks and L = legacy network attack).

N	Type	Vulnerability	Description
1	S	Impersonation	Inject a rogue packet with valid device ID to the controller to activate automation
2	S	Invalid Find-Node-in-Range command class	Send a malicious application payload that will cause a target device to find non-existent nodes and cause a DoS
3	S	Invalid route	Manipulate the device routing table and network traffic
4	S	Invalid application payload	Send invalid application payload (`CMDCL, CMD,VAL`)
5	S	Invalid transport payload	Send invalid transport payload (`P1, P2`)
6	S	Packet fields manipulation	Send unknown random packets via fuzzing techniques
7	L	Uncontrolled resource consumption	Unlimited request to device to cause them crash or exhaust their battery
8	L	Invalid sender	Send packet with invalid `SRC`

10 days and stored them with the label "normal" in the dataset CSV file.

2) ABNORMAL DATASET

The dataset of abnormal (negative) traffic is obtained from the payloads of known and existing Z-Wave attacks [3], [10] (*e.g.*, invalid route parameters, route table manipulation, remote code execution attacks, invalid payload injection, and DoS attacks). We also added some random payloads generated using fuzzing techniques to the abnormal dataset. The collected abnormal packets are stored with the label "abnormal" in the dataset CSV file.

3) SPECIALIZED Z-WAVE ATTACKS

As we previously mentioned, the Z-Wave protocol is structured and the manipulation of fields can cause an attack on the devices [49]. Therefore, we have to consider both legacy network and Z-Wave specialized attacks affecting secure S2 devices. Table 3 shows the attacks that ZMAD intends to detect. ZMAD considers the packets of listed attacks as abnormal.

4) DATASET SELECTION FOR EXPERIMENTS

We collected 158,110 normal packets and gathered 5,171 abnormal packets. To avoid model over-fitting problems and to increase the efficiency of the training, we select 5,325 normal packets (the same proportion as the abnormal packets), and use them in the experiment. Our packet datasets encompass the traffic of all devices used in our testbed.

B. EVALUATION METRICS

We use true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), recall (R), precision (P), F1-score (F), and accuracy (ACC) as metrics to measure the effectiveness of ZMAD. The recall is the number of correct positive outcomes divided by the total number of relevant samples:

$$R = \frac{TP}{TP + FN} \tag{9}$$

The precision is the number of correct positive results divided by the number of positive results predicted by ZMAD:

$$P = \frac{TP}{TP + FP} \tag{10}$$

The F1-score represents the balance between P and R:

$$F = \frac{2 * P * R}{P + R} \tag{11}$$

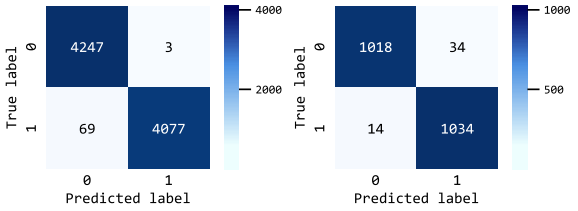
Accuracy is an indicator of how accurately ZMAD detects normal or abnormal traffic. The percentage of all correct predictions for all instances:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

We also evaluate the confusion matrix to show the model's predicted values compared to the actual results. The confusion matrix consists of TP, TN, FP, and FN, where its row represents the normal and abnormal packets in the dataset, and the column represents the packets the model predicted.

TABLE 4. Accuracy measurement results of ZMAD during training and testing.

N	Model	ACC	P	R	F1	Dataset
1	ZMAD-Training	**0.9914**	0.98	1.00	0.99	8,396
2	ZMAD-Testing	**0.9771**	0.97	0.99	0.98	2,100



(a) ZMAD-Training (b) ZMAD-Testing

FIGURE 9. Accuracy measurement of ZMAD.

VI. EVALUATION

In this section, we evaluate ZMAD. Section VI-A investigates how accurately ZMAD can detect abnormal Z-Wave packets. Section VI-B evaluates the performance and resource utilization of ZMAD. Section VI-C compares ZMAD with : (1) existing deep learning (DL) architectures and (2) an existing approach for Z-Wave protocol security. Section VI-D lists additional experiments conducted in the deployment of ZMAD. For training and testing the models, we ran ZMAD Python program on a desktop running Ubuntu Linux 22.04 with an Intel Core i5-10th Gen CPU (2.9 GHz), 8 GB RAM, and a 256 GB SSD.

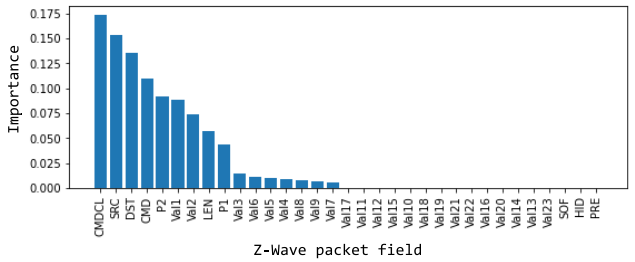


FIGURE 10. Feature importance plot of the raw Z-Wave dataset.

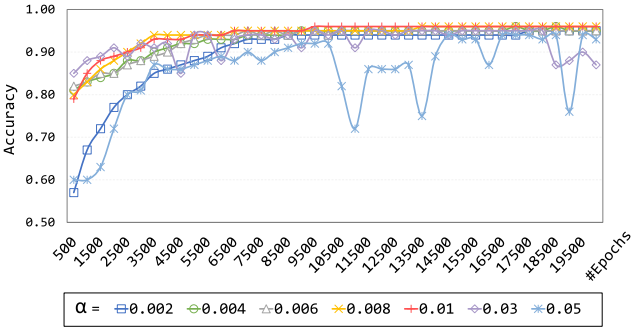


FIGURE 11. Detection accuracy according to Learning Rate α.

TABLE 5. Accuracy measurements between ZMAD and deep learning ANNs (multi-hidden layers) when ANNs are trained on non-formalized raw dataset.

N	Model	Layers	Accuracy	P	R	F1	Size
1	ZMAD	**One**	**0.98**	**0.97**	**0.99**	**0.98**	**23.1 KB**
2	RNN	Multi	0.92	0.98	0.85	0.91	372 KB
3	LSTM	Multi	0.92	0.97	0.86	0.91	1087 KB
4	MLP	Multi	0.92	0.96	0.86	0.91	105 KB
5	DBN	Multi	0.57	0.55	0.57	0.56	35 KB

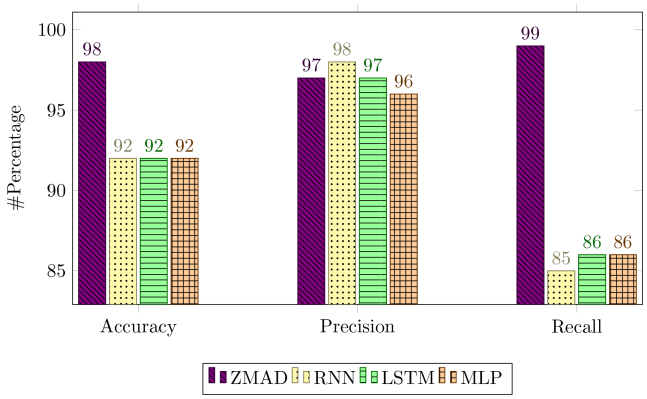


FIGURE 12. Metrics visualization between ZMAD and ANNs when ANNs are trained on non-formalized raw dataset.

A. ACCURACY OF ZMAD

1) EXPERIMENT METHODOLOGY

We measured the accuracy of ZMAD using the constructed dataset that includes 5,325 normal and 5,171 abnormal packets obtained from various real-world Z-Wave devices (see Section V-A). Each packet goes through a packet formalization process (see Section IV-C); the preprocessed packets had a maximum of 31 fields excluding the `PRE`, `SoF`, `H-ID`, `CS`, and `EoF` values. Before training our model, we first randomly shuffle the dataset and then select 80% as the **training** set and the remaining 20% as the **unseen testing set**.

Finally, we measured (1) how accurately ZMAD is trained and (2) how it determined the results of this 20% of the unseen testing set. Here, we trained ZMAD with high epoch values (85,000 epochs) to see what is the maximum accuracy ZMAD could achieve and used an α value equal to 0.01 (see Section VI-A4).

2) RESULT

In our experiments, ZMAD was able to achieve 95% accuracy in 26 sec (4,900 epochs) and a maximum accuracy of 99.14% (85,000 epochs) during **training**.

Moreover, ZMAD successfully discovered abnormal packets during **testing** on **unseen data** with an accuracy of 97.71%. Table 4 summarizes the accuracy measurement results, and Figure 9 shows the confusion matrix reports. ZMAD achieved its *average* accuracy at 85,000 epochs, but in fact, we observed that ZMAD showed almost the same detection accuracy as the maximum value after around 27900 epochs. Because the ZMAD ANN is built with fewer functions customized for Z-Wave packets, the high accuracy can be achieved in a relatively short time (see Section VI-B).

3) EFFICACY OF PACKET FORMALIZATION

ZMAD was able to maximize detection accuracy by using the packet formalization technique. To demonstrate this, we first examined the effect of each Z-Wave packet field on packet label prediction; we can generate feature-important scores using the `feature_importance` function. Figure 10 shows the measurement results. Through the results, we confirmed that the fields that ZMAD considers important in packet formalization (*e.g.*, CMDCL) played an important role in classifying normal and abnormal packets. In addition, we observed that the fields that ZMAD does not consider (*e.g.*, H-ID) do not contribute to determining the packet anomaly.

Next, we assessed the Z-Wave anomaly detection accuracy of the state-of-the-art deep learning architectures [50] without applying packet formalization: Long Short Term Memory Network (LSTM), Recurrent Neural Network (RNN), Simple RNN (SRNN), Multilayer Perceptrons (MLP), and Deep Belief Networks (DBN). Accuracy measurements were conducted in the same manner as we previously mentioned in Section VI-A1. Table 5 summarizes the accuracy measurement results and Figure 12 provides the visual metrics comparison. As a result, the models trained on dataset **without packet formalization** showed 57% to 92% detection

TABLE 6. Classification results per Z-Wave attack types.

N	Attack type	Accuracy	P	R	F1
1	DoS on S2 Controller	0.9969	0.99	1.00	1.00
2	DoS on S2 Route	0.9959	0.99	1.00	1.00
3	Invalid CMDCL	0.9479	0.99	0.91	0.95
4	Invalid CMD	0.9259	0.98	0.86	0.92
5	Invalid SRC	0.9729	0.99	0.95	0.97
6	Invalid P1	0.9889	0.99	0.99	0.99
7	Invalid P2	0.9489	0.99	0.91	0.95
8	Invalid DST	0.9939	0.99	1.00	0.99
9	Invalid VAL	0.9959	0.99	1.00	1.00

accuracy, while ZMAD (**with its packet formalization approach**) achieved 98% detection accuracy. This is because DL algorithms process all raw binary Z-Wave packet data with less important features that do not influence the final outcome prediction; therefore, giving the incorrect criteria in these top DL models during training. For example, the PRE, SoF, and Home ID fields are actually the same in a Z-Wave network, which compromises accuracy when training normal and abnormal packets. The results imply that the packet formalization technique used in ZMAD, which considers only core fields of packets, can effectively work in abnormal packet detection.

4) THRESHOLD SENSITIVITY

As we developed ZMAD from scratch, we need to find a suitable learning rate α (0.01 in our previous experiments) for the Z-Wave dataset to increase accuracy during training. To measure its sensitivity, we evaluated each detection accuracy result of ZMAD during 20,000 iterations (epochs) while varying α: 0.002, 0.004, 0.006, 0.008, 0.01, 0.03, and 0.05. Figure 11 shows the experimental results. When α was increased to 0.01, the difference was not large, but the accuracy gradually increased stably. When α is 0.01, ZMAD achieves up to 95% accuracy, and when α is greater than 0.01 (*e.g.*, 0.03 and 0.05), the accuracy starts to deteriorate. Therefore, we chose α as 0.01.

5) DETECTION ACCURACY ON VARIOUS Z-WAVE ATTACKS

Next, we measured the accuracy of ZMAD on various Z-Wave known attack traffics. We tested malicious unencrypted packets that cause DoS attacks on the S2 controller, alter the controller route table, and invalid CMDCL, CMD, and VAL fields. Table 6 provides the accuracy measurement results. ZMAD achieved an accuracy from 91% to 99.7% on injected attacks packets because it dissects the packet and deeply analyzes every single byte to find anomalies (see Section III). Through this experiment result, we proved that ZMAD can flexibly respond to various Z-Wave attacks.

TABLE 7. Classification results during testing by model with **one-hidden layer** and DL model with **multi-hidden layers** on **formalized Z-Wave dataset.**

N	Model	Layers	Accuracy	P	R	F1	Size
1	**ZMAD**	**One**†	**0.977**	0.97	0.99	0.98	23.1 KB
2	MLP_1	One†	0.958	0.963	0.947	0.955	28 KB
3	$LSTM_1$	One†	0.957	0.961	0.947	0.954	43 KB
4	$SRNN_1$	One†	0.949	0.959	0.934	0.946	31 KB
5	RNN_1	One†	0.949	0.956	0.936	0.946	31 KB
6	DBN_1	One†	0.5914	0.6323	0.3433	0.4450	4 KB
1	MLP_M	**Multi**‡	**0.984**	0.993	0.976	0.984	105 KB
2	$LSTM_M$	Multi‡	0.983	0.990	0.975	0.983	1087 KB
3	$SRNN_M$	Multi‡	0.98	0.995	0.965	0.979	372 KB
4	RNN_M	Multi‡	0.978	0.984	0.974	0.979	372 KB
5	DBN_M	Multi‡	0.551	0.617	0.293	0.397	35 KB

†: Model using only one hidden layer
‡: Model using multi-hidden layers

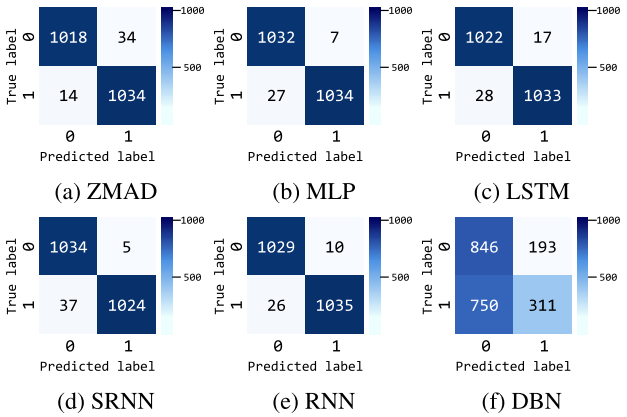


FIGURE 13. Confusion matrix of ZMAD and **multi-layers deep learning ANNs.**

B. PERFORMANCE OF ZMAD

1) PERFORMANCE OF CUSTOMIZED ANN USED IN ZMAD

ZMAD normalizes its model parameters during training. This leads ZMAD to achieve higher accuracy in shortened training times. To verify this, we trained the following two models: the normal model (without any parameter normalization) and the model using parameter normalization. We measured the number of epochs and the time until the accuracy of these two models reached 98%. In our experiments, we confirmed that the model with parameter normalization reached 98% detection accuracy in 27900 epochs (171s), while normal training required 85,000 epochs (522s).

a: NETWORK UPDATE

We added devices Dev15, Dev16, and Dev17 (see Table 1) to the initial Z-Wave network and captured their normal traffic. We added the new device traffic into the initial dataset and retrained ZMAD. In our experiments, we confirmed that ZMAD achieved the same detection rate and training time.

TABLE 8. Required libraries for training and deployment.

	DL ANNs	ML	ZMAD
Model training	Numpy, Pandas, TensorFlow, Keras	Numpy, Pandas, Sklearn	Numpy, Pandas
Model export	Numpy, Keras	Numpy, Joblib	Numpy
Deployment	Numpy, Panda, TensorFlow, Keras	Numpy, Panda Sklearn	Numpy, Pandas

Model training speed and CPU consumption is important during the detection and production phases, because when there are updates to the Z-Wave network (*e.g.*, adding new devices, updating the routing table), the initial model should be retrained to reflect the updates. If retraining takes a significant amount of time, an attacker can use this weakness to launch an attack during the re-training process. Therefore, shortening the training time is important and ZMAD proves to be effective for this purpose.

2) ELAPSED TIME AND RESOURCE UTILIZATION

ZMAD took only 26s (4,900 epochs) for training dataset to achieve an acceptable 95% detection accuracy and it took 522s to improve this accuracy to 99.14% during **training** phase. On average, the throughput of ZMAD for the dataset is 180 epochs per second. In addition, the exported trained model size of ZMAD is only 23.1 KB. This implies that ZMAD uses only small resources in terms of disk storage and could be suitable for running on resource-constrained devices (such as a Raspberry Pi [51]) and single-board MCUs.

C. COMPARISON WITH EXISTING APPROACHES

1) COMPARISON WITH DEEP LEARNING ANN ARCHITECTURES

We then evaluated the accuracy of ZMAD by comparing it with the state-of-the-art existing deep learning ANN architectures: LSTM, RNN, SRNN, MLP, DBN. We used the same testing datasets used in Section VI-A (**with applying packet formalization**); note that the abnormal packet detection results of the DL ANN architectures **without applying packet formalization** were presented in Section VI-A3 in Table 5. Here, Table 7 summarizes the accuracy comparison between ZMAD and the ANN architectures, and Figure 13 illustrates the confusion matrix.

The most notable result was that ZMAD succeeded in achieving high detection accuracy with a **single hidden-layer** and a **small trained model size**, while other architectures failed to achieve this. Except for DBN, other architectures, with **multiple hidden layers**, required as little as 105 KB and as much as 1,087 KB of disk space to achieve similar performance to the anomaly detection accuracy of ZMAD; this is approximately five to 47 times more disk space than ZMAD requires. This is because existing DL ANN architectures include several unnecessary functions and libraries, which can increase the trained model size and require extra libraries for deployment (see Table 8). However, ZMAD, which focused on lightweight for practical usage in smart homes, was able to achieve high detection accuracy even with a small model size by using minimal libraries and functions. Despite using **one hidden layer** and a few libraries for lightweight model training and deployment, we can demonstrate that ZMAD is a solid tool that can achieve high accuracy and can be implemented on resource-constrained devices with low overhead.

TABLE 9. Average detection rate for MBIDS and ZMAD in our current testnet.

Test	Trials	Detection rule†	Range	MBIDS	ZMAD
1	10	`if CMDCL not Valid:`	**11** CMDCL‡	**100%**	99.69%
2	10	`if CMDCL not Valid:`	**73** CMDCL	15%	**95%**
3	10	`if P1(FC) not Valid:`	**9** P1	n.a	**99%**
4	10	`if P2(FC) not Valid:`	**16** P2	n.a	**95%**

†: MBIDS uses manual conditional constructs for detection of **11 CMDCLs only** while ZMAD uses ANN for detection of anomalies on all supported **73 CMDCLs**.
‡: 11 CMDCLs implemented by MBIDS.

2) COMPARISON WITH A Z-WAVE SECURITY TECHNIQUE

We compare the effectiveness of ZMAD to the related study MBIDS, which is a packet misuse-based detection [8]. However, the MBIDS dataset was not available; therefore, we evaluate the effectiveness through the number of Command Classes that ZMAD and MBIDS can cover. Because most of the well-known Z-Wave attacks use CMDCLs with a malicious command payload, the effectiveness of abnormal behavior detection in Z-Wave networks is directly related to the number of covered CMDCLs [8]. We also assessed if MBIDS can detect specialized Z-Wave attacks exploiting other packet fields such as P1 and P2 which cause routing attacks.

In our experiments, we confirmed that ZMAD could cover seven times more CMDCLs than MBIDS: ZMAD covered 73 CMDCLs supported by our monitored smart home devices (current testnet), while MBIDS covered only 11 CMDCLs. ZMAD takes full advantage of the artificial neural network to learn all possible CMDCLs and their CMD and VAL payloads used in the target Z-Wave network. However, MBIDS used **manual rules** to encode each CMDCL valid parameter manually; obviously, this method has limitations in addressing all the CMDCLs and sub-parameters of the target network.

To assess this more thoroughly, we manage to implement MBIDS algorithm, which detects only 11 CMDCLs misuse out of 100+ available. In this experiment, we simulated an attacker who can inject packets with invalid fields or misuse, while leaving all other fields valid. After 10 trials, we evaluate MBIDS and ZMAD for correctness based on their detection rate. Table 9 summarizes the measurement results.

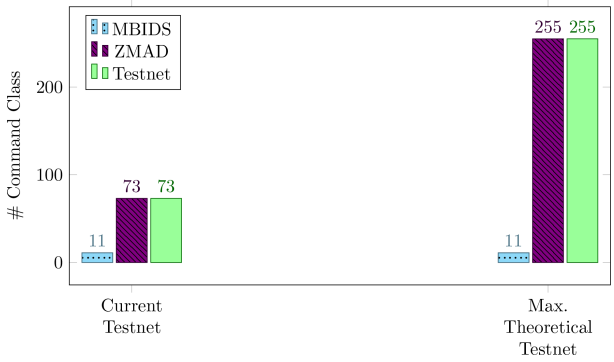


FIGURE 14. Command Class coverage by ZMAD and MBIDS.

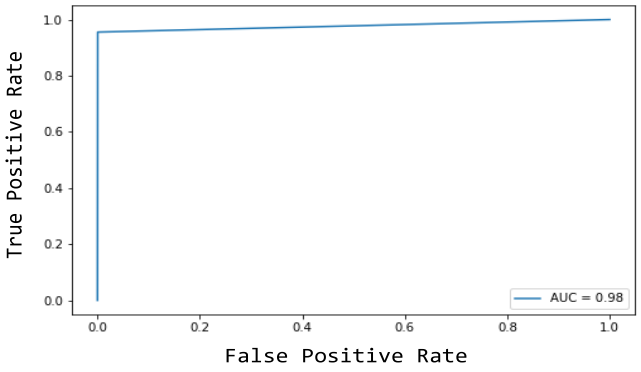


FIGURE 15. ROC curve of True Positive Rate (TPR) and False Positive Rate (FPR) of ZMAD.

Consistent with our assertion that CMDCL coverage is important for detecting anomalies in Z-Wave, ZMAD showed substantially better accuracy than MBIDS in most cases. In particular, in the range beyond 11 CMDCLs, MBIDS hardly detected anomalies. Worse, as MBIDS uses *conditional if* constructs to test for packet validity, it requires over 2,000 possible manual *if combinations* for testing 70+ Z-Wave CMDCLs [8] and thousands of sub-constructs per each single CMD. From the comparison experiments, we confirmed that ZMAD can effectively detect more Z-Wave attacks than the existing approach.

D. DEPLOYMENT TEST

We tested the ZMAD Python program both on the same desktop mentioned earlier as well as on a Raspberry Pi 4. To receive and transmit Z-Wave packets, ZMAD requires a dongle that supports sub-gigahertz frequency range such as the YardStick One [52]. During the deployment test, we assessed False Positive Rate (FPR) and True Positive Rate (TPR). Figure 15 depicts the receiver operating characteristic (ROC) curve and the area under the ROC curve (AUC) of TPR and FPR for ZMAD. The result shows that ZMAD reaches rapidly above 98% TPR while keeping a relatively low FPR (< 0.01). This supports that ZMAD can address the IDS challenge by producing fewer false alarms during real-time operation.

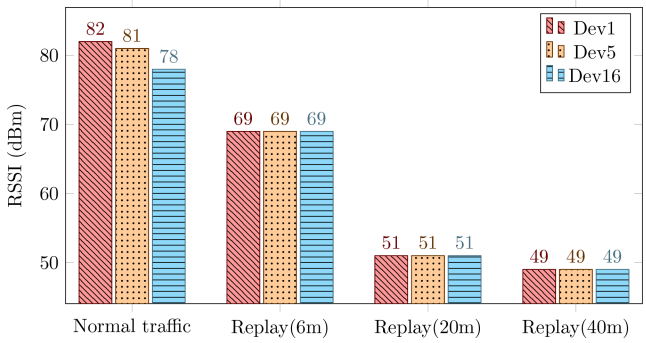


FIGURE 16. Replay attack detection by the evaluation of attacker's injected wireless packet RSSI value from different distances: 6m, 20m, and 40m.

VII. DISCUSSION

A. IMPORTANCE OF A CUSTOMIZED ANN FOR Z-WAVE

With the introduction of the software bill of material (SBOM) [53] requirements, software development has drastically changed. SBOM recommends developers to provide specific details of their application (*e.g.*, [54]) at the module level and comply with a set of standards in order to make their solution resilient from cyber-attacks. SBOM is one of the reasons we developed ZMAD ANN from scratch; this allows us to control every single module and function used, the entire process from input to output. In addition, our solution can be customized very quickly to the requirements of future new Z-Wave specifications. While using ZMAD, novice IDS operators will have a better understanding of the entire system process compared to a third-party black box library-based IDS that takes input, processes it internally, and provides output.

B. EXTERNAL WIRELESS ATTACK MANAGEMENT

1) FUZZING ATTACKS

An attacker can sniff Z-Wave network traffic and retrieve available device IDs. Then, the attacker can use protocol fuzz testing (*e.g.*, [3], [55]), which is a technique that sends random packets, with the aim of causing a target device to crash. However, fuzzing generates random frame fields that change the validity of allowed values, thus ZMAD can determine an injected packet as abnormal.

2) REPLAY ATTACKS

An attacker can use a reconnaissance method to replay valid Z-Wave packets to take control of a target device that does not authenticate the sender. Although it is difficult to detect replay attacks with ZMAD alone, we can detect them by leveraging the following aspects: (1) examining sender's radio signal attributes (*e.g.*, transmission speed, channel, and received signal strength indicator (RSSI)), (2) analyzing inter-packet arrival time (IPA) of the receiver, and (3) challenging the attacker device.

RSSI is an important factor because it varies depending on the distance of the transmitter. While the RSSIs range

of valid devices in the house less variate if assessed, the attacker's RSSI shows an unusual value (see Figure 16). For example, we observed that injected packets for replay attacks on Dev16 have the same speed (40 Kbit/s), an RSSI ranging from 49 to 69, and channel 1, whereas valid Dev1 communication to Dev16 has speed of 100 Kbit/s, RSSI of 82, channel 0; note that in the case of normal traffic, RSSI showed a value close to 80 on other devices. Hence, RSSI evaluation approach could be used to capture replay attacks that mimic legitimate Z-Wave communication.

In addition, IPA can be used to capture replay attacks. A typical Z-Wave IPA is less than 25 ms for two consecutive packets sent from the same Z-Wave device. However, an attacker randomly generates packets in Z-Wave device communication without considering legitimate IPA behavior.

Last, we can use techniques that can challenge attackers if they use other source node IDs that do not exist on the network. For example, we can ping an attacker's spoofed tool or send special Z-Wave frames that require a specific response. If the adversary tool is not a valid Z-Wave device, then it will not respond accordingly. Therefore, we can use this information to take selective countermeasures based on the severity of the attack. Future versions of ZMAD will incorporate techniques that advance these intuitions.

3) COUNTERMEASURES

ZMAD can assess the severity of external wireless attacks and take appropriate mitigation actions. For example, if an attack affects a key Z-Wave network controller, ZMAD can send packets that automatically activate security devices such as turning sirens and lights on. For attacks targeting devices, ZMAD can send countermeasure packets that either turn off the connected smart power or raise an alert.

C. ADVERSARIAL MACHINE LEARNING

Attacks on IDS and ML systems during training time are known to poison datasets [56], [57]. As described in Section VII-B, an adversary with adequate knowledge of Z-Wave networks can attempt to poison ZMAD's training process by mimicking legitimate devices' traffic. Several methods have been proposed (*e.g.*, [58], [59]) to prevent adversarial machine learning (ML) on images and audio samples. As stated in [43], to forge adversarial ML requires generating minimal modification from the initial input dataset, which could be easy to implement for specific data types such as audio and image files. In contrast, it is challenging to implement a minimal modification in a valid Z-Wave numerical packet fields without altering the packets' semantics and context. Therefore, even if small adversarial modifications to the packet occur, it can easily be flagged as invalid by ZMAD. In other words, this attack can be marked with abnormal ease by a previously trained model, as the assumption that the initial ZMAD model training dataset consists of valid device network traffic.

D. FUTURE WORK

The current version of ZMAD uses one hidden layer for low resource consumption, high efficiency, and high performance. We plan to develop ZMAD as a way to preserve the existing performance benefits while utilizing multiple hidden layers to achieve much higher accuracy. We are also considering devising ZMAD to be supported directly by the Z-Wave controller to monitor and filter all possible traffic. Therefore, future development may include ZMAD as the primary controller acting as both an IDS and an Intrusion Prevention System (IPS).

VIII. CONCLUSION

Z-Wave smart home legacy devices are vulnerable to wireless injection attacks, because they do not support encryption and cannot be updated owing to their one-time programmable memory. Moreover, recent secure S2 Z-Wave devices are still vulnerable to specific unencrypted targeted attacks that alter their normal network operation.

To protect smart home users, we proposed ZMAD, a lightweight anomaly-based intrusion detection system developed on a custom artificial neural network built from scratch for Z-Wave network requirements. By using the packet formalization and centralized learning techniques, ZMAD achieved an anomaly detection accuracy of 95% within a short training time (26s) and 98% in 522s. We also demonstrated that ZMAD can cover more Z-Wave command classes compared to the existing techniques, and that ZMAD with packet formalization technique is effective in detecting abnormal behaviors in Z-Wave networks than existing ANN architectures on raw data. With the help of ZMAD, users living in smart homes will be able to have the benefit of secure Z-Wave services. All collected datasets are publicly available to foster future research.

REFERENCES

[1] Z-Wave Alliance. (2023). *The Smart Home is Powered by Z-Wave*. [Online]. Available: https://z-wavealliance.org
[2] Z-Wave Alliance. (2023). *Z-Wave Products*. [Online]. Available: https://products.z-wavealliance.org
[3] C. K. Nkuba, S. Kim, S. Dietrich, and H. Lee, "Riding the IoT wave with VFuzz: Discovering security flaws in smart homes," *IEEE Access*, vol. 10, pp. 1775–1789, 2022.
[4] N. Boucif, F. Golchert, A. Siemer, P. Felke, and F. Gosewehr, "Crushing the wave–new Z-wave vulnerabilities exposed," 2020, *arXiv:2001.08497*.
[5] C. Badenhop, S. R. Graham, B. E. Mullins, and L. O. Mailloux, "Looking under the hood of Z-wave: Volatile memory introspection for the ZW0301 transceiver," *ACM Trans. Cyber-Phys. Syst.*, vol. 3, no. 2, p. 20, 2018, doi: 10.1145/3285030.
[6] L. Rouch, J. François, F. Beck, and A. Lahmadi, "A universal controller to take over a Z-wave network," in *Proc. Black Hat Eur.*, 2017, pp. 1–9.
[7] C. W. Badenhop, S. R. Graham, B. W. Ramsey, B. E. Mullins, and L. O. Mailloux, "The Z-wave routing protocol and its security implications," *Comput. Secur.*, vol. 68, pp. 112–129, Jul. 2017, doi: 10.1016/j.cose.2017.04.004.
[8] J. D. Fuller, B. W. Ramsey, M. J. Rice, and J. M. Pecarina, "Misuse-based detection of Z-wave network attacks," *Comput. Secur.*, vol. 64, pp. 44–58, Jan. 2017, doi: 10.1016/j.cose.2016.10.003.
[9] K. Kim, K. Cho, J. Lim, Y. H. Jung, M. S. Sung, S. B. Kim, and H. K. Kim, "What's your protocol: Vulnerabilities and security threats related to Z-wave protocol," *Pervas. Mobile Comput.*, vol. 66, Jul. 2020, Art. no. 101211.

[10] CERT Coordination Center. (2022). *Silicon Labs Z-Wave Chipsets Contain Multiple Vulnerabilities*. [Online]. Available: https://kb.cert.org/vuls/id/142629
[11] Atrim. (2022). *700 Series Compared to Older Z-Wave*. [Online]. Available: https://atrim.co/zwave/z-wave-700-series.html
[12] Z-Wave Alliance. (2022) *Aeotec are First to Successfully Certify 700 Series Z-Wave Products*. [Online]. Available: https://z-wavealliance.org/aeotec-are-first-to-successfully-certify-700-series-z-wave-products/
[13] Wikipedia. *Advanced Encryption Standard*. Accessed: Jun. 16, 2023. [Online]. Available: https://en.wikipedia.org/wiki/Advanced_Encryption_Standard
[14] Home Assistant Forum. (2022). *Z-Wave Migration Broke My Entire House*. [Online]. Available: https://community.home-assistant.io/t/z-wave-migration-broke-my-entire-house/401605
[15] M. Roesch, "Snort—Lightweight intrusion detection for networks," in *Proc. Usenix LISA*, 1999, pp. 229–238.
[16] L. Babun, H. Aksu, L. Ryan, K. Akkaya, E. S. Bentley, and A. S. Uluagac, "Z-IoT: Passive device-class fingerprinting of ZigBee and Z-wave IoT devices," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–7.
[17] J.-C. Liou, S. Jain, S. R. Singh, D. Taksinwarajan, and S. Seneviratne, "Side-channel information leaks of Z-wave smart home IoT devices: Demo abstract," in *Proc. 18th Conf. Embedded Networked Sensor Syst.*, 2020, pp. 637–638.
[18] C. K. Nkuba. (2023). *ZMAD Devices Traffic Dataset*. [Online]. Available: https://github.com/CNK2100/ZMAD-Dataset
[19] C. W. Badenhop, B. W. Ramsey, B. E. Mullins, and L. O. Mailloux, "Extraction and analysis of non-volatile memory of the ZW0301 module, a Z-wave transceiver," *Digit. Invest.*, vol. 17, pp. 14–27, Jun. 2016, doi: 10.1016/j.diin.2016.02.002.
[20] J. Hall, B. Ramsey, M. Rice, and T. Lacey, "Z-wave network reconnaissance and transceiver fingerprinting using software-defined radios," in *Proc. Int. Conf. Cyber Warfare Secur.* Norfolk, VI, USA: Academic Conferences International Limited, 2016, p. 163.
[21] J. L. Hall, "A practical wireless exploitation framework for Z-wave networks," Air Force Inst. Technol., Wright-Patterson, OH, USA, Tech. Rep. AD1054454, 2016.
[22] 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 *Proc. Int. Conf. Crit. Infrastruct. Protection*. Cham, Switzerland: Springer, 2015, pp. 209–227.
[23] J. D. Fuller and B. W. Ramsey, "Rogue Z-wave controllers: A persistent attack channel," in *Proc. IEEE 40th Local Comput. Netw. Conf. Workshops (LCN Workshops)*, Oct. 2015, pp. 734–741, doi: 10.1109/LCNW.2015.7365922.
[24] B. Fouladi and S. Ghanoun, "Security evaluation of the Z-wave wireless protocol," in *Proc. BlackHat USA*, vol. 24, 2013, pp. 1–2.
[25] C. Badenhop and B. Ramsey, "Carols of the Z-wave security layer; or, robbing keys from Peter to unlock Paul," in *Proc. PoC GTFO*, vol. 12, 2016, pp. 6–12.
[26] J. Hall and B. Ramsey, "Breaking bulbs briskly by bogus broadcasts," in *Proc. ShmooCon*, Washington, DC, USA, 2016. [Online]. Available: https://shmoo.gitbook.io/2016-shmoocon-proceedings/one_track_mind/02_breaking_bulbs_briskly_by_bogus_broadcasts
[27] J. Hall, B. Ramsey, M. Rice, and T. Lacey. (2016). *EZ-Wave*. [Online]. Available: https://github.com/cureHsu/EZ-Wave
[28] T. J. Bihl, K. W. Bauer, M. A. Temple, and B. Ramsey, "Dimensional reduction analysis for physical layer device fingerprints with application to ZigBee and Z-wave devices," in *Proc. IEEE Mil. Commun. Conf.*, Oct. 2015, pp. 360–365.
[29] Z. B. Celik, G. Tan, and P. McDaniel, "IoTGuard: Dynamic enforcement of security and safety policy in commodity IoT," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2019, pp. 1–15.
[30] Z. B. Celik, P. McDaniel, and G. Tan, "SOTERIA: Automated IoT safety and security analysis," in *Proc. USENIX Annu. Tech. Conf.*, 2018, pp. 147–158.
[31] C. Fu, Q. Zeng, and X. Du, "HAWatcher: Semantics-aware anomaly detection for appified smart homes," in *Proc. 30th USENIX Secur. Symp. (USENIX Security)*, 2021, pp. 1–9.
[32] Y. Jia, L. Xing, Y. Mao, D. Zhao, X. Wang, S. Zhao, and Y. Zhang, "Burglars' IoT paradise: Understanding and mitigating security risks of general messaging protocols on IoT clouds," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2020, pp. 465–481.
[33] D. T. Nguyen, C. Song, Z. Qian, S. V. Krishnamurthy, E. J. M. Colbert, and P. McDaniel, "IoTSan: Fortifying the safety of IoT systems," in *Proc. 14th Int. Conf. Emerg. Netw. Exp. Technol.*, Dec. 2018, pp. 191–203.
[34] Y. Yu and J. Liu, "TAPInspector: Safety and liveness verification of concurrent trigger-action IoT systems," 2021, *arXiv:2102.01468*.
[35] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A. Sadeghi, and S. Tarkoma, "IoT SENTINEL: Automated device-type identification for security enforcement in IoT," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 2177–2184.
[36] S. Kim, S. Woo, H. Lee, and H. Oh, "VUDDY: A scalable approach for vulnerable code clone discovery," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 595–614.
[37] Y. Xiao and W. Shi, "MVP: Detecting vulnerabilities using patch-enhanced vulnerability signatures," in *Proc. 29th USENIX Secur. Symp. (USENIX Security)*, 2020, pp. 1165–1182.
[38] S. Woo, H. Hong, E. Choi, and H. Lee, "MOVERY: A precise approach for modified vulnerable code clone discovery from modified open-source software components," in *Proc. 31st USENIX Secur. Symp. (USENIX Security)*, 2022, pp. 3037–3053.
[39] H. Jang, K. Yang, G. Lee, Y. Na, J. D. Seideman, S. Luo, H. Lee, and S. Dietrich, "QuickBCC: Quick and scalable binary vulnerable code clone detection," in *Proc. IFIP Int. Conf. ICT Syst. Secur. Privacy Protection*. Cham, Switzerland: Springer, 2021, pp. 66–82.
[40] IFTTT. (2022). *Connect Your Apps and Devices in New Ways With Powerful Automations*. [Online]. Available: https://ifttt.com/
[41] J. Mirković, S. Dietrich, D. Dittrich, and P. Reiher, *Internet Denial of Service: Attack and Defense Mechanisms*. Upper Saddle River, NJ, USA: Prentice-Hall, 2004.
[42] J. D. Fuller, "A misuse-based intrusion detection system for ITU-T G.9959 wireless networks," Air Force Institute of Technology Wright-Patterson, OH, USA, Tech. Rep. AD1053808, 2016.
[43] T. D. Nguyen, S. Marchal, M. Miettinen, H. Fereidooni, N. Asokan, and A. Sadeghi, "DÏoT: A federated self-learning anomaly detection system for IoT," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 756–767, doi: 10.1109/ICDCS.2019.00080.
[44] TinyML Foundation. (2022). *Tiny Machine Learning*. [Online]. Available: https://www.tinyml.org/
[45] Embedded. (2022). *How to Quickly Deploy TinyML on MCUs*. [Online]. Available: https://www.embedded.com/how-to-quickly-deploy-tinyml-on-mcus/
[46] R. Hecht-Nielsen, "Theory of the backpropagation neural network," in *Neural Networks for Perception*. Amsterdam, The Netherlands: Elsevier, 1992, pp. 65–93.
[47] Silicon Laboratories. (2019). *Z-Wave Zniffer User Guide*. [Online]. Available: https://www.silabs.com/documents/public/user-guides/INS10249-Z-Wave-Zniffer-User-Guide.pdf
[48] SmartThings. (2022). *One Simple Home System*. [Online]. Available: https://www.smartthings.com/
[49] MITRE. (2022). *CVE-2020-9057*. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2020-9057
[50] M.-P. Hosseini, S. Lu, K. Kamaraj, A. Slowikowski, and H. C. Venkatesh, "Deep learning architectures," in *Deep Learning: Concepts and Architectures*. Cham, Switzerland: Springer, 2020, pp. 1–24.
[51] E. Upton and G. Halfacree, *Raspberry Pi User Guide*. Hoboken, NJ, USA: Wiley, 2014.
[52] G. S. Gadgets. (2021). *YARD Stick One-a Sub-1 GHz Wireless Test Tool Controlled by Your Computer*. [Online]. Available: https://greatscottgadgets.com/yardstickone/
[53] National Telecommunications and Information Administration. (2020). *NTIA Software Component Transparency With SBOM (Software Bill of Materials)*. [Online]. Available: https://www.ntia.doc.gov/SoftwareTransparency
[54] S. Woo, S. Park, S. Kim, H. Lee, and H. Oh, "CENTRIS: A precise and scalable approach for identifying modified open-source software reuse," in *Proc. IEEE/ACM 43rd Int. Conf. Softw. Eng. (ICSE)*, May 2021, pp. 860–872.
[55] H. Park, C. K. Nkuba, S. Woo, and H. Lee, "L2Fuzz: Discovering Bluetooth L2CAP vulnerabilities using stateful fuzz testing," in *Proc. 52nd Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Jun. 2022, pp. 343–354.
[56] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2012, pp. 1467–1474.

[57] J. McHugh, A. Christie, and J. Allen, "Defending yourself: The role of intrusion detection systems," *IEEE Softw.*, vol. 17, no. 5, pp. 42–51, Sep. 2000.
[58] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.
[59] A. Kurakin, I. J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Artificial Intelligence Safety and Security*. Boca Raton, FL, USA: Chapman & Hall/CRC, 2018, pp. 99–112.

CARLOS KAYEMBE NKUBA (Member, IEEE) received the first B.S. degree in computer science from the Protestant University of Lubumbashi, Lubumbashi, Democratic Republic of the Congo (DRC), in 2010, and the second B.S. degree in information technology (major) and global management (minor) and the M.S. degree in computer science information technology from Handong Global University (HGU), Pohang, South Korea, in 2014 and 2016, respectively. He is currently pursuing the Ph.D. degree with the Computer Science and Engineering Department, Korea University. Prior to joining HGU, he was a IT Network Administrator with Gecamines Mining, DRC. His research interests include computer software and network security, fuzzing, and the IoT automated vulnerability discovery.

SEUNGHOON WOO received the B.S., M.S., and Ph.D. degrees in computer science and engineering from Korea University. He is currently a Research Professor with the Center for Software Security and Assurance (CSSA), Korea University. He has published papers on software security and software engineering in top conferences, such as S&P, USENIX Security, and ICSE. His research interests include software security, vulnerability detection, and software composition analysis.

HEEJO LEE (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science and engineering from POSTECH, South Korea. He is currently a Professor with the Department of Computer Science and Engineering, Korea University, and the Director of the Center for Software Security and Assurance (CSSA). He is a Founding Member and the Co-CEO of Labrador Labs. Before joining Korea University, he was the CTO with AhnLab Inc., from 2001 to 2003, and a Postdoctoral Researcher with Purdue University, from 2000 to 2001. He is the Editor of the *Journal of Communications and Networks* and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY.

SVEN DIETRICH (Senior Member, IEEE) was an Assistant Professor with the Computer Science Department, Stevens Institute of Technology, from 2007 to 2014, and a Senior Member of the Technical Staff with CERT and CyLab, Carnegie Mellon University, from 2001 to 2007. He is currently a Professor with the Computer Science Department, City University of New York (CUNY) Hunter College, and The Graduate Center, and the Director of the Computer and Network Security Laboratory, CUNY Hunter College. Prior to joining CUNY Hunter College, he was with the Mathematics and Computer Science Department, CUNY John Jay College of Criminal Justice, as an Associate Professor, from 2014 to 2020. His research interests include computer and network security, cryptography, anonymity, and privacy.