Leveraging the Zigbee Pattern of Life to Identify Devices

Kurt Jarvis

LEVERAGING THE ZIGBEE PATTERN OF LIFE TO IDENTIFY DEVICES

A dissertation submitted to Dakota State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Cyber Operations

March 20, 2024

By

Kurt L Jarvis

Dissertation Committee:

Dr. Yong Wang

Dr. Bhaskar Rimal

Dr. Kristian Olivero

Beacom College of Computer and Cyber Sciences

DAKOTA STATE
UNIVERSITY®

DISSERTATION APPROVAL FORM

This dissertation is approved as a credible and independent investigation by a candidate for the Doctor of Philosophy degree and is acceptable for meeting the dissertation requirements for this degree. Acceptance of this dissertation does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department or university.

Student Name: Kurt Jarvis

Dissertation Title:
LEVERAGING THE ZIGBEE PATTERN OF LIFE
TO IDENTIFY DEVICES

Graduate Office Verification: DocuSigned by: Abby Chowning F44C8D9E621C417... Date: 04/10/2024

Dissertation Chair/Co-Chair: DocuSigned by: Dr. Yong Wang 70AB505BC7B649E... Date: 04/10/2024
Print Name: Dr. Yong Wang

Dissertation Chair/Co-Chair: ______________ Date: ______
Print Name: ______________

Committee Member: DocuSigned by: Dr. Bhaskar Rimal 91D1AA4B61574EF... Date: 04/11/2024
Print Name: Dr. Bhaskar Rimal

Committee Member: DocuSigned by: Dr. Kristian Olivero A6B1E74F69AB4EE... Date: 04/15/2024
Print Name: Dr. Kristian Olivero

Committee Member: ______________ Date: ______
Print Name: ______________

Committee Member: ______________ Date: ______
Print Name: ______________

Submit Form Through Docusign Only
or to Office of Graduate Studies
Dakota State University

ABSTRACT

ZigBee is the open-standard enabling smart devices to be adopted in new and innovative ways. In this research, the network layer of the ZigBee protocol is examined to further the understanding of security impacts it brings to the environment. The first research question is determining of the ZigBee beaconing patterns reveal the device type. This is the first layer that introduces encryption and the results indicate that beacon layer data with the lower layer information do not provide enough information to confidently identify the device prior to admittance to the network. This is important in the identification of rogue devices. The second research question is how applying machine learning to a set of features extracted from network traffic can reveal device types. The results are yes, training a model to identify traffic is possible leveraging network-layer traffic to identify device types within the network. The third research question is revealing if the traffic being captured at the network layer can be categorized as abnormal leading to potential malicious traffic identification. The results indicate a yes when the abnormal traffic is greater than one standard deviation from the average packet size. A reverse majority-vote classifier is created that classifies devices based on observed traffic. This research expands network identification, inventory, and potential detection for ZigBee smart devices that can be leveraged in the field environments today.

DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another. I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Kurt L Jarvis

Kurt L Jarvis

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Introduction

Internet of Things (IoT) devices have been rapidly adopted into almost every aspect of business, commercial, industrial, and personal life at a rate exceeding the ability to secure them. The IoT infrastructure is a heterogeneous combination of technologies, protocols, and proprietary devices. The wireless spectrum is dominated by the Wireless Fidelity (Wi-Fi) Alliance protocol, which is heavy in communications and requires high power consumption. The addition of new protocols reducing the communication burden and lowering the power consumption extending the life of devices has opened up a new window to innovation. Some proprietary protocols are available, along with an open standard called ZigBee riding over the RFC 802.15.4. This protocol has allowed vendors to offer new devices in new areas that have exploded into the market. Taking simple devices that enable automation and placing them in new areas has opened the door to innovation. With the adoption of the word "smart" being applied to electronic devices that allow connectivity with control, these devices are controlling mundane functionality like outlets and coffee pots. However, they are also being applied to water treatment plants, railway control devices, traffic controllers, and power grids that affect many aspects of daily life.

1.2 Motivation

This rapid adoption of these devices without a holistic security plan is a risk that has been forced upon all who adopt these devices into their lives and businesses. The consumer desires to have a plug-and-play feature and has demonstrated this quality with every new technology adoption that is introduced. When vendors have a monopoly, they can add security features that support rapid adoption by consumers. The current state of the industry has multiple vendors with niche areas of the market with very little concern for providing security. Due to the range of the ZigBee protocol, the adoption into homes may provide risk but it is localized to a personal or family extent that can be accepted by the owner. We are observing the adoption into industrial and business settings that are touching finance, industrial controls, traffic safety, and public safety that can have dire effects to the public and potential life.

1.3 Existing Problems

ZigBee is built on top of the RFC 802.15.4 protocol that defines the wireless medium standards that focus on low power consumption and lower transmissions compared to Wi-Fi. After the first two layers, such as the physical layer and the data link layer, ZigBee operates at two layers: the network layer and application layer. The network layer offers 8 different levels of security options ranging from zero security to encryption with frame checks. Using level 8 provides the most security but is rarely used in heterogeneous environments. Most devices are designed to communicate with a controller without configuration until it is admitted into the network. Once in the network, a set of keys can be shared to increase security. However, the network is only as secure as the least secure device. Networks that cannot reach level 8 with all devices are retrograded to the security level of the lowest device. This common scenario allows for a plethora of attacks into the heterogeneous network that propels an attacker into corporation and industrial areas that were previously hardened. Additionally, the number of devices that are added to these spanning networks is rapidly catching up with the number of wired devices, if not surpassing them. With coordinators extending the range of ZigBee networks, the complexity of device mapping has exceeded the tools and technologies available. The problem is the ability for ZigBee devices to be infiltrated into a network with malicious intent without the availability of monitoring to identify the devices or malicious behavior.

1.4 Research Gap

The majority of ZigBee research has focused on RFC 804.15.4 looking at the signal layer to identify device locations and investigating the attacks rampant in the protocol stack. While many of these attacks are similar to attacks that were found in earlier versions of Wi-Fi standards, the ZigBee network layer provides a new area of research regarding the security implementation and device enumeration. A heterogeneous collection of devices is being added rapidly that are increasing the vulnerability landscape of homes, businesses, and industrial complexes. As shown in the literature review section, the research indicates the available attacks while missing the research to enable monitoring with indicators of compromise through network traffic analysis.

1.5 Research Questions

The first question (R1) is: *Do ZigBee devices have a consistent beaconing pattern that can be uniquely identified from other devices based on timing?* There are two subquestions: Does the introduction of a new ZigBee device in a heterogeneous environment cause the network to switch network keys; can we fingerprint the device based on beaconing patterns? The second question (R2) is *Can we identify device types based on network layer traffic using machine learning (ML)?* There are two subquestions: Is there enough indicators at the network layer to provide enough uniqueness for a device to be identified into categories; can we fingerprint the device based on network layer traffic? The third question (R3) is *Given an inventory of items authorized to communicate on a ZigBee network, can traffic analysis of the network layer identify devices that are sending abnormal traffic?* This question involves defining abnormal traffic as traffic outside of a learned range of expected traffic patterns. A sub-question to this includes attempting to expand the ML model from R2 to identify these abnormalities. These three questions form the foundation of the research and lead to knowledge discovery and potential future research.

1.6 Approach

This research takes a pragmatic approach to a growing problem. After completing an extensive literature review, the laboratory listed in Chapter 3 is used to collect network layer data to answer the research questions. These data are analyzed to perform feature extraction to apply machine learning. From the machine learning model, an assessment can be made that could automate the inspection of traffic to identify devices based on their network-layer traffic patterns.

1.7 Research Challenges

There are a few challenges to overcome. First is isolation of network traffic. There are fourteen channels available within the spectrum that can cause interference. This will be accomplished by performing some spectrum analysis of the lab environment first and forcing the hubs to operate in the channel with the lowest interference. The second challenge is control of network key exchanges. The Eria ZigBee Hub provides the interface for full control. The Amazon Alexa Echo does not provide a user option to change the network key. The key will have to be extracted during network capture in the transport key link exchange. This requires a little more effort at the APS layer [1], but once extracted can be applied to the python script for accurate data extraction. The last challenge identified is the automation of sensors. The hubs provide an automation sequence that can be used to change the state of all the other devices. To automate the sensors, it will required some inventive ideas. To automate the motion sensor, the researcher plans to put the sensor facing automated window blinds. The movement of the blinds trigger the motion sensor, allowing for a repetitive and predictable pattern that can be used for data collection. The door and window sensor are more complicated, but a similar technique can be used. A hot-wheels race track can be set up to place one side of the sensor next to the track while the other side is attached to a hot-wheels car. Each time the car passes the sensor, it will trigger the alert. This may cause **R3** to become predictable, so a separate dataset may be needed to prevent this type of sensor from being predictable.

1.8 Dissertation Outline

This dissertation describes the rigorous method applied to answering the research questions. Chapter 2 introduces the background literature of previous research that set the foundation for these experiments. This covers methods and topics at each of the ZigBee layers relevant to this topic. Chapter 3 presents the methodology used during this research to determine the validity of the procedures and results. Chapter 4 outlines the experimental lab leveraged to establish, collect, analyze, and record the data and devices. Lab design is a simplistic view of an industrial automation setup that mimics a real scenario. Chapters 5, 6, and 7 define the research implementation for each research question individually. Details of data capture, analysis, and model building are provided. Chapter 8 provides a concrete conclusion to this research, what it means to the field of cyber operations, and where the next evolution of this research should be taken.

Chapter 2

Literature Review

The limitations of the Wi-Fi protocol delayed adoption of devices requiring low power consumption. Every transmission requires power consumption that makes small devices without continual power have a short lifespan. The 802.15.4 standard was introduced as an alternative option on 2.4 GHz that changes the burden to allow the wireless protocol to use less transmissions, but also limits the distance. Built on top of 802.15.4, wireless standard options include ZigBee, ZWave, and EOcean [2]. The ZigBee Alliance is the only open-standard out of the three that invite adoption at minimal costs with transparency. Companies jumped at the opportunity to provide connectivity to their devices, enabling the evolution of "smart devices" [2]. The changes were made so fast that the adoption of devices outpaced the ability to provide security contexts. Each device independently provides some small advancement in control automation, but combined with a multitude of other devices provides a large problem-set for security professionals. When devices are applied to the home, the impact is localized and risk is assumed by the home owner. However, when applied to power plants, electrical grids, traffic signals, and airport controls, the risk increases exponentially. Even the drive to telework to use home computers, networks, and resources bring undue risk to companies. Mandatory Virtual Private Networks (VPNs) are an attempt to isolate local network traffic from the corporate network, but does not prevent pivoting on the machine itself. One attempt is to apply the same centralization of network controls used on wired architectures. The second attempt is to force centralized authentication. Both of these concepts break the implementation of the 802.15.4 wireless protocol. This literature review focuses on the techniques being used today and the gaps currently in this space.

2.1 Integration Into Everything

IoT has been deployed in almost all aspects of business, industry, and homes. Figure 2.1 shows graphically an ecosystem leveraging multiple mediums to communicate. While its convenience is appealing, it brings a multitude of concerns. First is a privacy perspective. Taking a simple addition to the power grid to allow electric-metering devices to transmit to a *SmartMeter* allows the company to use a device to drive down the road and collects readings without visually inspecting the meter at, on, or in the home [3]. There are lots of benefits for the company profits and efficiency. However, this is also information available to anyone else reading the signals. Allowing someone to do a little pattern analysis to determine when a home owner draws more power can indicate activity such as when shower hygiene is happening. While not too intrusive, reading when power is being consumed from industry entry gates provides a little more insight into operations. Taking the concept to an extreme, knowing when power is drawn at a missile launch site has a dramatic impact on national security [4]. Second is the idea of autonomy and taking cyber-physical systems into an interconnected environment making decisions, such as vehicle operation. The consequences of attacks within this realm become deadly quickly. We already incorporate cellphones into car entertainment systems, which took over 5 years to convince engineers to stop connecting vehicular controls with the entertainment systems.

As society continues to push from “man-in-the-loop” to “man-on-the-loop” to, eventually, autonomy; the security concerns continue to slow the progress [4]. Although ZigBee is just one of the wireless options in this context, attackers will continue to target the

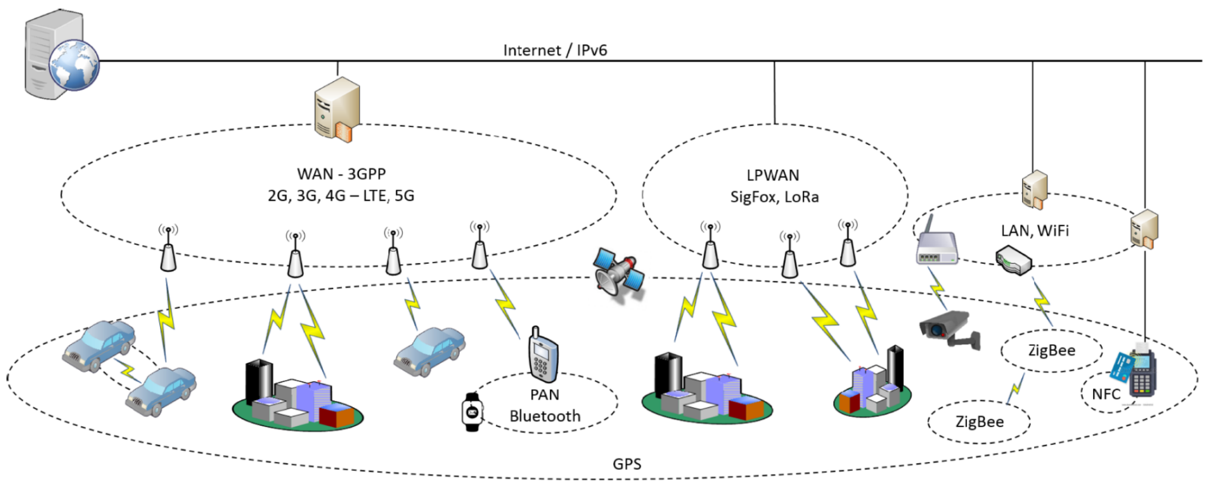


Figure 2.1: Wireless Hierarchy [5]

weakest and most vulnerable areas. Third, the sheer amount of devices accessible to the Internet is drawing interest from script kiddies to nation-state cyber actors. With the IoT expanding into billions of devices, corruption of one device rapidly spreads to adjacent devices, providing a foothold and compromise. Even when an intrusion detection system (IDS) is built into the network, the amount of damage is insurmountable depending on the devices that these IoT are controlling [6]. While the locality of ZigBee devices may prevent a wide-spread impact, recent open-source demonstrations with devices attached to drones called "warflying" are showing how wide-spread an attack can get [6], [7].

2.2 Radio Frequency Analysis

Radio Frequencies (RF) are everywhere and almost never have encryption on the headers. Due to the open airwaves, there is no way to prevent signal capture. Even with directional antennas, it does not prevent data interception (or injection). Regardless of the payload, the RF emissions provide a set of information that may lead to conclusions. Research has recently been conducted to use what is available to find devices and train models with that limited feature set. Because ZigBee is on top of IEEE 802.15.4 standards, ZigBee devices suffer attacks at the lower layers. Several proposals have been made to update the IEEE 802.15.4 standard, but they have not yet been accepted or published.

2.2.1 Triangulation

Any device that puts off a signal can be received. The basic measurement of signal strength is the received signal strength indicator (RSSI). While it is impossible to know the amount of signal sent, the device can measure what it received. With two devices, concentric circles can determine potential locations narrowing the device location to two. With a third device, triangulation occurs. Two approaches have been researched to further the topic. First, a neural network was applied with input provided from receivers spread across an area that uses known devices and their locations to find devices not previously identified (labeled as rogue) [1]. This approach requires registration of devices based on signatures. The second approach is conducting signal processing to determine devices based on other devices within the proximity [1]. This involves a high processing classifier researchers called Zero-Effort Proximity Detection (ZIP). Both of these approaches require high amounts of processing and distinct hardware modifications not inherent in ZigBee devices today.

2.2.2 RF Fingerprinting

Efforts are researched to find device-dependent emissions used to distinctly native attributes [8]. Distinctive attributes were researched leading to coining a term "RF-DNA" [8]. Efficiencies were researched to make the technique practical. The research is limited to the information available within the physical layer. Outside of packet sizes and frequency of packets, there are few additional features available for training models. Follow-on research requires additional features that are unavailable from the physical layer transmissions alone. IEEE 802.15.4 focuses on the low power consumption and combines packets within the standard to keep the communications lower than Wi-Fi. This research indicates to move up the technology stack for fingerprinting.

2.2.3 Jamming

At the RF level, the jamming technique most applicable is flooding [9]. This paper refers to five types of jamming but in all cases some signal is transmitted to drown out the intended signal. The technology limits the range to 14 channels (country-dependent) within the 2.4 GHz band. A quick listen by a scanner [7] and anyone can begin flooding the channel with a jamming signal. It does not matter if the signal has any meaning. The only anti-jamming techniques offered is to implement spectrum spreading [9], but the consequences of that include increase power consumption and transmission times; both of which are undesirable in most IoT use cases.

2.2.4 Radio Frequency Gaps

Research at the IEEE 802.15.4 standard has reviewed the ability to detect that a device is on, transmitting, and then use signal strength to determine location. The ability to identify they type of device is missing due to the lack of information extracted at those levels. The standard is designed to provide the physical transmission and be agnostic from the protocol riding upon the transmission. This disconnect has limited the ability to support ZigBee understanding.

2.3 Power Attacks

A key advantage to ZigBee is the feature of long battery life. The longer battery life is proportionally to the amount of transmissions. Effort in design reduced the bandwidth compared to the 802.11 frameworks enabling this feature. Fully-functional devices that have constant power are less prone to power attacks, but the reduced-functionality devices that are in more remote positions living from a battery cell can be reduced quickly.

2.3.1 Energy Depletion Attacks

Instead of attacking the communications to high-jack for intercept, these devices can fall victim to a depletion attack that renders the device useless [10], [11]. Similar to a de-authentication attack on Wi-Fi, a continuous beacon requiring the device to respond keeps the device active for as long as messages are being received. Leveraging beacons takes a considerable amount of time as the power consumption for a beacon is much smaller than other protocol packets. However, if the attacker could target a specific device with replay attacks at the network layer then depletion could be sped up tremendously (best case of 90 minutes in the example) [10]. Research conducted by Khanji, Iqbal, and Hung highlight that this attack could occur at the MAC layer with slower results [11]. An attack with frames are much smaller and incomplete frames would be dropped. This would still incur the power to process the packets, but not consume as much as using transmit power in some response. While this attack may not be a viable option for an attacker, it still exists within the architecture. Regardless of the type of attack, the criticality of the node within the topology can create Denial of Service attacks which lead to a desired effect by the attacker.

2.3.2 Embedded Multiple Flows

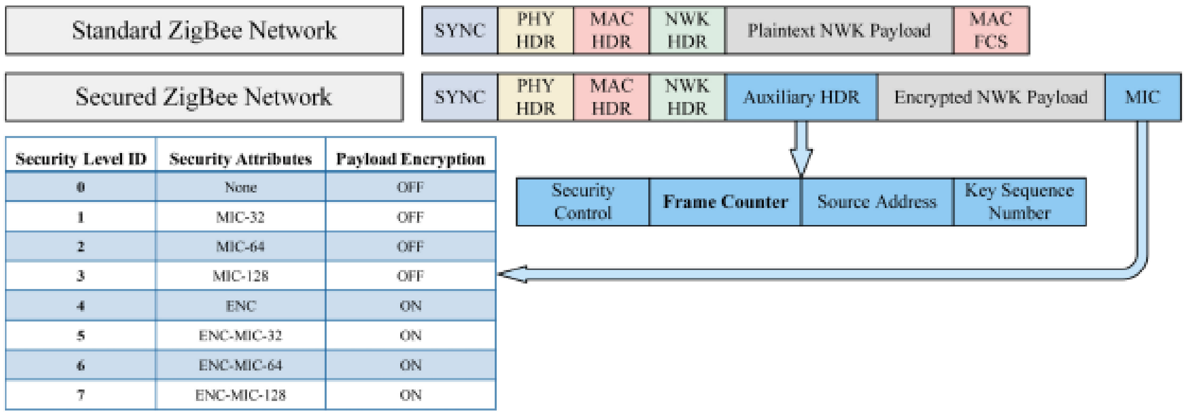
A counter to the power attacks is reducing the number of communications within the 2.4 GHz band that is currently cluttered with collisions and devices. Embedded Multiple Flows (EMFs) use two techniques to interweave the ZigBee communication in-between the Wi-Fi transmissions [12]. This solution is a software-only solution that could be done with firmware updates and is encouraging. The research gap here is prioritization of packets. Similar to Quality of Service (QoS) within Wi-Fi, the saturation of packets within the 2.4 GHz band leaves the ZigBee packets stranded waiting for its slot within the protocol. Wi-Fi issued new modulation techniques that implement improved Read to Send/Clear to Send (RTS/CTS) procedures that limit the opportunity for EMF to be effective [12], [13].

2.3.3 Power Attack Gaps

The approaches to reduction of power attacks are attempting to limit the transmission need but do nothing to prevent such transmissions. Due to the agnostic nature of the RFC 802.15.4 layers, any intelligent processing to determine attacks at these layers have to be implemented in the control layer above, which is the network layer of ZigBee. Even the proposed integration of the EMF focuses on changing the Wi-Fi messaging to support ZigBee integration, which does nothing in ZigBee-dominant topologies.

2.4 Protocol Attacks

The protocol itself has security options that can provide confidentiality and integrity. The options are listed in Figure 2.2. ZigBee Alliance recommends that level 5 and above should be used to prevent payload introspection. Unfortunately that is a recommendation and conflicts with many business models that require pre-installed encryption keys. This reduces the interoperability with other manufacturers resulting in lowers profits which is not desirable within the community. A nonce could help with the interception issues that could provide an element of integrity, but a nonce is difficult to keep a common starting point without a network time stamp or similar item that each device can use to validate created nonce [2]. As long as devices are designed to last, having unique nonces without repeating will also be a challenge. Lastly, the wired community continues to recommend an intrusion detection system (IDS) for solutions where devices designed for this implementation do not have the computing space or power to conduct [2]. Many of these IDS recommendations are attempting to take a wired solution and apply it to a domain that cannot support it.



Security Level ID	Security Attributes	Payload Encryption
0	None	OFF
1	MIC-32	OFF
2	MIC-64	OFF
3	MIC-128	OFF
4	ENC	ON
5	ENC-MIC-32	ON
6	ENC-MIC-64	ON
7	ENC-MIC-128	ON

Figure 2.2: ZigBee Network Layer Security Options [14]

2.4.1 Replay Attacks

A set of research has been conducted on the attacks to reuse data to get the same results. If a ZigBee packet is captured that causes a light bulb to turn on, we could get the same effects without knowing what was actually in the packet by conducting replay attacks. The amount of encryption would not matter as nothing needs to be broken, just resent. At the ZigBee network layer, there are a set of integrity checks to validate the packet. The frame counter is located at the layer below that does not have encryption. As long as the frame counter is greater than the last one received, the packet is accepted. Researchers have provided a potential solution that would incorporate a timestamp that would validate that packets are old and should be ignored [14], [15]. Using this approach, a software update for fully-functional devices could be deployed and potentially a hardware update for reduced-functionality devices. A consorted effort by each manufacturer would be necessary for change, which is not applicable to heterogeneous environments. Another attack in this category is the Sybil Attack [16]. This allows a device to masquerade as a valid, trusted device already admitted in a network. In the ZigBee hierarchy (see Figure 2.1, it requires the capturing of the trust key that is exchanged at the application layer (one layer above the network layer). While that may appear technically challenging, the interoperability default design leveraging an auto-join policy allows for the decryption of the transport key by monitoring the link exchange. This exposes the entire network opening up rogue device possibilities. Even when the auto-join is disabled, there are other techniques like downgrade attacks that force the network to exchange the network key that can be decrypted allowing for interception of the transport key anyway. This is a business decision for rapid adoption of devices for convenience vice a stringent management plan. A difficulty with the Sybil attack is the data storage requirement needed to determine outliers [16]. The size and processing requirements move this processing to a dedicated computer that usually is NOT a ZigBee device.

2.4.2 Deadlock Attack

Similar to the Replay Attack, the frame counter must always be higher than the last number one accepted. In networks that reboot and reset prior to reaching the maximum value, it is not an issue. However, IoT devices are designed to be left on for years without reboot. If the frame counter is set to the maximum value and sent to a ZigBee device, that device will reject further packets as that 8-bit value will wrap back around to zero [17]. While no changes to the RFC have been made yet, this situation generates the need for a network reset still yet to be implemented.

2.4.3 Passive Attack

Attacks do not have to respond or send data to the device. The confidentiality of information or the loss of that data to unknown devices also impacts the protocol [11]. While home automation networks are less concerned with data transmission leaks, industries and businesses are highly sensitive to that data protection [3]. A device that just listens to the open airwaves for RFC 802.15.4 and collects that information impacts data privacy. As stated above, level 5 does provide some inherent protection that requires key management. Real-time attacks can be difficult for decryption, but adversaries do not always need it in real-time. The ability to collect data then use high computing power later is a technique used across nation states globally. The use of these devices within critical infrastructure makes these devices a high value target within the landscape.

2.4.4 Protocol Attack Gaps

The previous research on protocol attacks highlight a need for additional monitoring that is not appropriate for the ZigBee devices. While a dedicated network device with constant power may provide a network defender with information flow that can be processed and scrutinized, it does not improve the protocol or the situation; it provides a response and investigation capability. There is no suggested effort to innovate or iterate the protocol standard, only deploy additional items within the ecosystem to provide detection.

2.5 Summary

The ZigBee landscape has garnered attention at each of the ZigBee layers that allow the inspection of transmitted data. While ZigBee is being fully integrated into critical spaces, the available data to be captured provides both opportunity and threats to the devices and the networks in which they are apart. The radio signals themselves can be leveraged to identify where devices are located. Due to the nature of radio waves, ZigBee is vulnerable to other radio attack techniques that impact the signal-to-noise ration. Apart from the radio signals itself, this chapter outlined the techniques used against the design of the protocol that have been researched. While some of the techniques impacting ZigBee can be mitigated, there are others that still need researched solutions. Understanding the history of what has been researched allows the research questions of this experiment to build upon the knowledge foundation. The next chapter outlines the methodology used to build the experiments to answer the research questions.

Chapter 3

Methodology

This chapter outlines the research methodology for identifying ZigBee devices through pattern of life analysis. The approach is a quantitative design science methodology that focuses on a single-case mechanism experiment in a laboratory environment with a representative sampling of ZigBee devices. The approach is discussed in depth in this chapter, originating with the analysis of the research question, designing the environment with engineering rigor, and the approach to data analysis.

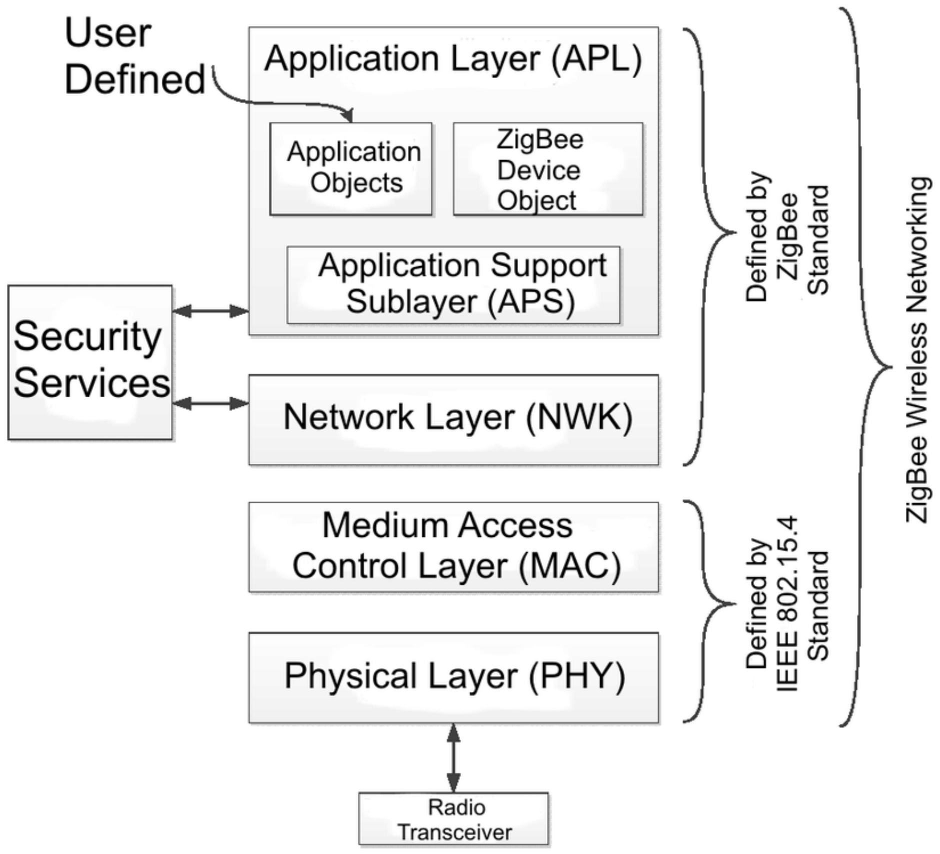
3.1 Methodology Selected

The main premise of the research is that there are numerous devices transmitting across radio frequencies that can be captured, collected, analyzed and evaluated to categorize and make predictions. The relationship is defined by the theory of quantitative research by Creswell as "an interrelated set of constructs ... that specify the relationship among variables" [18]. Design science is selected to find solutions to the research questions listed in Chapter 1. Artifacts are generated by the environment and call for a change in the implementation in the real world [19]. Each research scenario and field deployment may contain different devices in the array of configuration possibilities, but it continues to depend on collecting transmitted data and analyzing the pattern to predict types. Because of this commonality between environments, it allows the application of single-case mechanism experiments to answer the above research questions.

3.2 Single-Case Mechanism Experiments

This research aims to link the cause and effect correlation between the ZigBee traffic transmitted and intercepted with the functional properties of the device. Single-case mechanism experiments have an inference design that is used to validate the results through descriptive inference, normally through charts, graphs, and tables of analyzed data [19]. These experiments are systematically set up with a design that supports data integrity and data preservation. An important distinction is the "set of experimental methods that can be used to test the efficacy of the intervention [treatment]" that this research aims to answer [20]. Wieringa provides a checklist to ensure that all the elements are needed to contribute to abductive inference [19]. The first element is interpreting the signals and analog data. RFC 802.15.4 defines the first two layers that convert between analog waves and digital fields. This is the physical layer and the media access control layer as shown in Figure 3.1. These layers also provide the basics of power management, signal strength, and communication frequencies that allow RFC 802.15.4 to outperform RFC 802.11 in power consumption, but reduces the signal strength.

The next two layers are defined by the ZigBee standard, which enables device communications for control functionality. Between these two standards, data fields are structured for systematic extraction. The standards do provide the option for optional data, which can cause variation between vendors implementing the standard to include/exclude refined information. The second element is the summaries of the data planned within the experiment. According to Figure 3.1, all ZigBee communications involve at least the first three layers of the stack, but generally include at least one element of the fourth layer. The first exception lies in the router configuration where the device serves as a repeater, and the fourth layer would not be processed (or decrypted), but sent out again to allow the communication to expand across a larger span to the targeted endpoint. The second exception is the beaconing process, which contains nothing in the network layer while



User Defined
Application Layer (APL)
Application Objects
ZigBee Device Object
Application Support Sublayer (APS)
Security Services
Network Layer (NWK)
Medium Access Control Layer (MAC)
Physical Layer (PHY)
Radio Transceiver
Defined by ZigBee Standard
Defined by IEEE 802.15.4 Standard
ZigBee Wireless Networking

Figure 3.1: ZigBee Layers [2]

establishing association with the coordinator. Regardless, the analysis is available at each of these layers with the potential to provide summaries, graphical comparisons, and analyzed correlations. The third element is the inherent support for data preparation. The laboratory discussed later in this chapter enables the data to be prepared in a repeatable manner that can be reproduced and validated, as show in Figure 3.2.

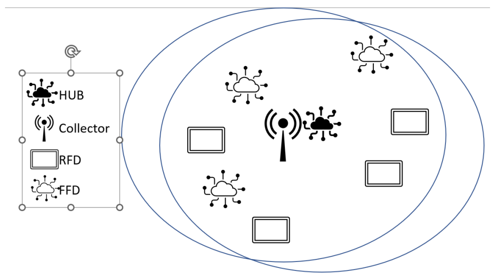


Figure 3.2: Methodology Diagram

The data collected during this experiment is representative of the field environment as it is the same device type and configuration communicating in the normal established patterns per the RFC. The fourth element is data interpretation, which is essential to validity with industry peers. In this research, the results of the raw, intermediate and final analysis results are presented and shared to show the logic between interpretations and correlations. The fifth element is descriptive statistics. The usage of the RFC-defined data leveraged both in field deployments and the lab environment being the same allows for any statistics to be applied and directly correlate between the environments. The conclusions are directly applied to the field. The sixth and last element is repeatability. This design allows for the full disclosure of all environment configurations, collection tools, and processing programs that can be used by any future researcher or industry to repeat these tests and produce similar results. Without the ability to seed communication starting points and device enrollment timing, it is impossible to get exact raw results. However, any raw results generated from a similar lab environment can repeat the data curating, analysis, graphing, and reach the same conclusions. Although single-case mechanism experiments require rigor in engineering and demonstrated logic flow that any researcher can follow, the researcher must have a minimal level of expertise and experience.

3.3 Trustworthiness

The amount of sampling required for this research is dependent on two criterion: the communication mechanisms used for each device and the number of devices being used within the environment. Commercial adoption has reduced cost and increased availability of devices that can be leveraged in a lab environment. Simplex devices such as plug adapters, contact sensors, and motion detectors are expected to communicate in a similar fashion. Complex devices such as colored light bulbs, doorbell keypad sensors, and thermostats have extended communication requirements because of their increased capability to control systems as a full function device. Additionally, the number of devices used within an environment increase the traffic load but does not change the communication schemes outside of the network design (as mentioned above). Treatments applied to samplings are in the context of being able to identify the device through its normal usage model. For those sensors or devices that would be at a daily frequency, will need to be sped up (not with simulation but with manual triggering) so that the device captures are authentic and can be leveraged for identification. Many ZigBee hubs come with automation scenes that allow for the application of communication at a designated interval or event (i.e., dawn and dusk). Inference is valid in this case because the same mechanisms used for communications, alerts, and triggers in real-world examples are replicated within the lab environment. This allows accurate modeling and scalability for commercial applications. The design of the experiment is focused on transparency in data collection and a repeatable process for future collections. The data are obtained directly from communication devices without alteration, allowing data cleansing, curating, and presentation. Moving the collection devices to a new environment requires no change in the data analysis stages or machine-learning process. As indicated by Batley, Contractor, and Caldas; the goal of single-case methodology is to show that the observations made during the treatment are a function of the applied treatment [21]. Trustworthiness is inherent in this design methodology.

3.4 Ethical Concerns

ZigBee has the potential to have proprietary and sensitive information flowing across ZigBee layers (Figure 3.1). However, there is encryption happening at the network layer and application layer. The network layer encryption key is known (or can be extracted) allowing transparency to extract communication information. However, the application layer performs a key exchange that is not being decrypted during any step of this process. The information within that layer is captured and processed as a data stream that remains encrypted. Analysis techniques like entropy, stream lengths, and patterns are performed, but data within that layer remains protected during all phases of the experiment design. No new information disclosures are seen, induced, or attacked. However, there is a potential for uncovering weak encryption techniques being used by a vendor which requires the researcher to do the due diligence: reporting the findings to the vendor through that vendor's submission processes. No devices used within this research include health devices that regulate any aspects of human health. There exists devices such as glucose monitors that leverage the ZigBee protocol. While a monitor itself may not impact health, but taking action on what that monitor reports can lead to deadly results. This research should be applied to the health industry devices to protect individuals using this protocol within their devices.

3.5 Summary

The rapid adoption of automation devices across the home, office, and industrial have allowed networks of heterogeneous devices to be affordable, varied, and available. The research methodology explained in this chapter focuses on a single-case mechanism experiment approach for design science methodology. The laboratory design described in this chapter allows the transfer of findings from this research to be directly shifted to industry and field environments to reveal the same patterns of life for ZigBee devices. The experiment design enables the research questions to be answered with the analysis available with the treatment. The next chapter takes this design and analyzes the data to reach the conclusions for the research questions.

Chapter 4

Experiment Setup

This chapter outlines the tools, hardware, software, and overall environment needed to reproduce this research. This approach takes the methodology for Chapter 3 and puts in practice the environment. The devices that are chosen reflect the popular smart devices being deployed on ZigBee networks within homes. It was important for these experiments to have a clean capture of data ensuring that there was limited packet loss and reduced duplication. While many configurations can be leveraged, this chapter outlines what was used to collect the data leveraged answering the research questions in Chapters 5, 6, and 7.

4.1 The Lab Design

The lab design is based on a star topology. Other lab layouts provide repeater function that allows long-haul communication across a large area, commonly for industrial areas (see Figure 4.1). Repeated communication provides a set of noise that would drive the need for data reduction during the machine-learning phase of the research. However, the trained machine-learning model is able to accept any duplication created as it will be uniquely identified by the source (RFC 802.15.4 layer 2). This reduces the dataset for collection and analysis. The factors that influence this design are the type of devices that can be controlled as well as the location needed to fulfill their function. As indicated in Figure 4.1, there exist three categories of devices: the full function device, the coordinator, and a reduced-function device. The unique controller for any network is the coordinator that serves as the master to all the other devices within the network. Once a device is designated as a coordinator, that device controls the network key and enrollment processes. The full function devices are capable of being a coordinator and serving as a repeater for any traffic that destined for a device not currently in range of the emitter.

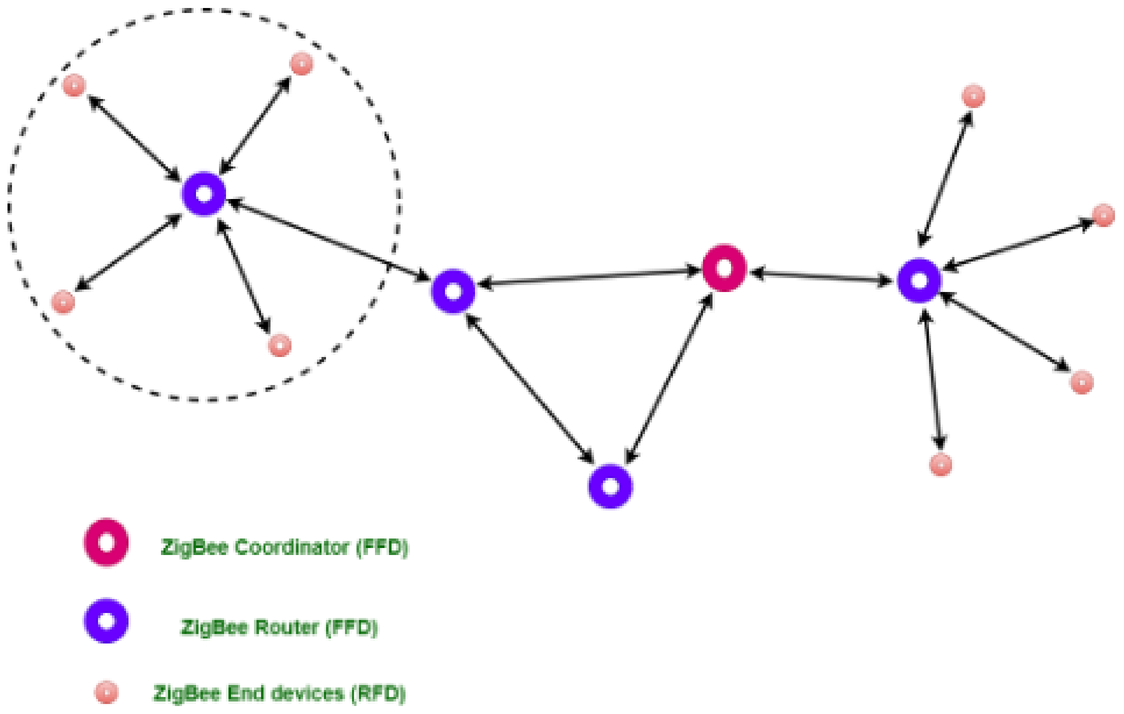


Figure 4.1: ZigBee Topologies

Any full function device that is designated to be a coordinator when another coordinator exists within the same channel establishes a different network that allows other devices to enroll with it. Allowing different networks to reside on the same channel increases network traffic flow and reduces battery life. Reduced-function devices do not have those capabilities and are usually placed at the edge locations of a network topology design. These devices scan all channels looking for controllers to enroll. Once enrollment occurs, the reduced function device stops beaconing on all channels and communicates with the network on the enrolled channel. Upon dis-association, the device resumes the beaconing process across all channels looking for a new enrollment option. The controlled environment of this scenario is not to influence any industrial control device that regulates safety equipment or life-manipulating devices. This allows the researcher to mimic commands and inject different control features without regards to the effects of any device these ZigBee controllers influence. This stimulus is designed to collect data and speed up collection efforts. The devices selected for the lab have the following requirements:

- the device must be inter-operable with a heterogeneous hub and
- the device must be able to accept configuration and control commands from the hub.

The first requirement enables multiple manufacturers to be included in the lab and removes bias associated with a manufacturer's timing, RFC adherence, and proprietary restrictions. The second requirement allows for an automation setup to speed up the data collection timeline. For example, if a light-bulb is the unit under test (UUT) within the environment, the normal usage of that device would be associated with the user of that light-bulb which would drastically reduce the amount of times it is engaged and extend the collection time required. Automation allows a stimulant to imitate user behavior and remove the dependence on human interaction to collect the data needed for machine learning to perform classification. The UUTs are listed in Table 4.1.

The two devices on the list that serves as coordinators is the Alexa Echo Dot and the Eria AduroSmart Hub. The rest of the devices have reduced functionality. Both of these devices provide a user-interface available via an app on a smartphone. This allows for setup, monitoring, and automation of any device paired with it. For these experiments, only one coordinator is used at a time.

4.2 Data Collection

An abundance of low-cost devices provides easy access to multiple device types in local environments. This lab uses two ZigBee cards (Figure 4.2).

Table 4.1: Devices Used

Device Type	Name	Model	Quantity
Smart Plug	Wemo	F7C063	2
Smart Plug	Wemo	WSP080	2
Smart Plug	Sengled	E1C-NB7	4
Smart Plug	Sengled	E1C-NB7	4
Contact Sensor	Sonoff	M0802070007	1
Mini Smart Socket	Apromio	US-101	2
PIR Motion Sensor	SOUJAMAO	B08P6QMYC5	1
Motion Sensor	Eria	81823	1
Adjustable White Bulb	Sylvania	71831	2
Adjustable White Bulb	Sengled	E21-N13A	4
Full Color LightBulb	OSRAM	73693	2
Color Changing LightBulb	Sengled	E11-N1EAWA	2
AduroSmart Hub	Eria	81821	1
Echo Dot	Alexa	5th Gen	1

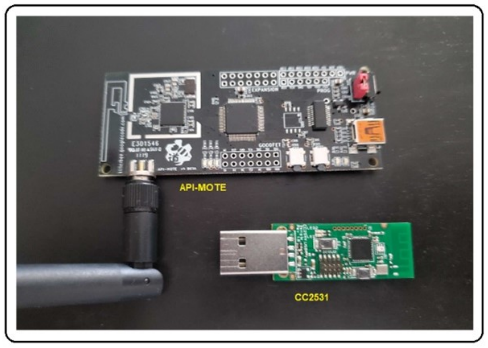


Figure 4.2: ZigBee Hardware [7]

The first card is a CC2531 used to scan one channel at a time and does not allow packet injection. The second device is an Apimote capable of both scanning and packet injection, but cannot do both at the same time. These were used on an Ubuntu 18.5 virtual machine leveraging the killerbee python library to sniff and inject packets. While newer versions of the Ubuntu kernel are available, the firmware drivers would not work with the newer version without manual modification of the firmware source code. The researcher rolled back the operating system versions until 18.5 to allow the firmware drivers to communicate with both ZigBee cards. To ensure a spectrum of device types available to home and small businesses, a combination of smart plugs, light bulbs, motion detectors, and contact sensors that connect to two different hubs: Amazon Alexa Echo Dot and Eria SmartHub are used. The ZigBee framework allows for full function devices and coordinators so both hubs can serve as both routers and coordinators. In this configuration, neither will serve as a router to reduce the rebroadcasting of data. Research question 1 (R1) presents the follow-on question of changes to communications based on which hub is serving as the coordinator while the other device is the full-functioning device. This lab allows for repeatability to swap out coordinators and re-running the data collection and analysis. Like hub vendor options, each of the device types are selected with different vendor types to remove bias based on manufacture tolerance and RFC adherence.

4.3 Procedures Followed

A number of stages are required to ensure that laboratory results provide repeatable and consistent data. The first stage is the establishment of a collection device and proper positioning to allow for a star topology. An Ubuntu 18.5 operating system is installed on a virtual VMWare device. The APIMOTE leveraged factory firmware that comes pre-installed to collect and transmit ZigBee signals. The CC2531 does not come with installed firmware, so the researcher utilized the CC-Debugger hardware (Figure 4.3) from Texas Instruments to flash the firmware per manufacturer's guidance. The firmware is downloaded from the Texas Instruments repository, and compile instructions are followed to generate the compiled firmware for loading with the CC-Debugger device [7]. The collection device will be placed so that it was within range of each device, ensuring that its received signal strength indicator (RSSI) is strong enough to collect data and not need any device serving as a router or repeater. While anything above -80 begins to drop packets, the RSSI must rise above -60 for all devices to limit dropped packets.

The second stage is to prepare the software to run the data collection. Python 3.9 is installed from the default Ubuntu repository with *sudo apt install python3* with no

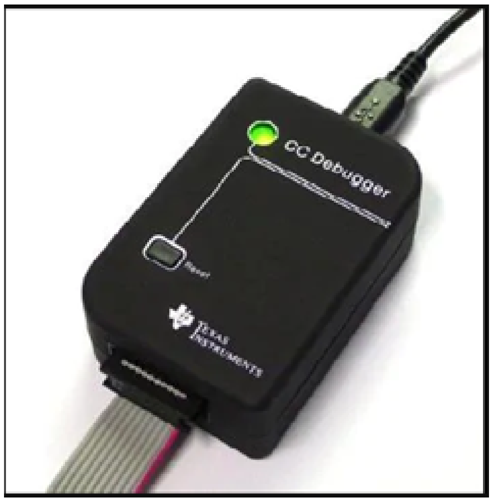


Figure 4.3: CC-Debugger [7]

modifications. Killerbee is cloned with git from GitHub.com and installed following the instructions in the Readme.txt. Wireshark is also installed from the default repository with *sudo apt install wireshark* to be used to validate traffic as an independent check. This should complete all the software required to collect the raw data.

The third stage is to systematically capture the phases from each device. The first phase is collecting the device transmitting without associating with a coordinator. Plugging in the device and running a network capture of the beacons for at set amount of time for each device. The second phase is collecting the enrollment of each device with a coordinator. The capture starts with the device beaconing, then interacting with the coordinator to search for available devices. After selecting the device on the coordinator and joining the coordinator, the data capture will terminate. This is repeated for each device independently to only have the coordinator and that device be communicating on one channel. The third phase is the collection of data while all devices are connected over a pre-defined period. During this period, all devices are fully interacted with each other to collect all actions that the individual devices can perform. To ensure repeatability, the automation tools used within the hubs will be documented and available for later reproduction.

These procedures ensure repeatability and validation of the lab environment. The procedures are repeated, allowing both hubs to serve as a coordinator. This allows for multiple runs with ML training and validates the collection. To ensure clean data, each coordinator is established on one of the sixteen possible channels within the ZigBee frequency band. Killerbee is used to dump data on each channel, ensuring that the channel was free of other communications before a coordinator is assigned to a channel. This reduces any bleed over from other devices within the area that inject packets during the collection method.

4.4 Data Analysis

With raw data collected, an extraction is required to migrate data from pcap files (killerbee saving format with zbdump) to analyzable text files for ML. A python program is written to read the pcap files, extract the data layers into a row of column-separated values (CSV), and save the results into a CSV file for further analysis. The killerbee library allows RFC 802.15.4 traffic to be extracted consistently supporting both ZigBee and ZWave technologies. Depending on the manufacturer's adherence to the standard, any traffic not containing RFC-compliant ZigBee traffic should be dropped. With the above requirement for RSSI that is implemented, the number of dropped packets should be minimal and are removed potential interference from the environment. Multiple tools are available to present CSV data, manipulate it, graph it, and provide insight into its usage. Kazdan indicates that visual inspection is the primary method of data evaluation [20]. Excel is initially used to review the results and leverage filtering to provide preliminary analysis. Since python is already being leveraged, machine-learning libraries are readily available. A Jupyter notebook is leveraged to conduct exploratory data analysis (EDA) and use the seaborn and matplot libraries to ingest a CSV file and present different correlations, data cleaning, data reduction, data normalization, and data discretization.

Using a notebook allows for a repeatable look at each data capture. Common looks at data distribution, statistical tests, and correlations between dependent and independent variables can be represented and graphically analyzed to make logical inferences and assessments. In machine learning, this is a classification problem that attempts to put the data into distinguishable buckets. The multiple techniques available within the sklearn and panda libraries are available for convergence and repeatability.

4.5 Converting Questions To Answers

The research questions are derived from the gaps in previous research that intersect with problems in the field. For each of these questions, each have a hypothesis:

- **R1**: Manufacturers provide a unique signature in their implementation of the wireless signal propagation that can be used to identify a captured unknown signal's source. The captured data are listed in the following section, but the metrics that will be used to prove this are a machine-learning model that can predict the model based on signal packets as input. The F1 score for that model reflects the ability to determine the type of device. Results are expected to be $>$ 80 percent accurate.

- **R2**: Manufacturers implement the network layer protocol uniquely enough to identify device types operating within a trusted network based on traffic flow features. The metrics are the same as **R1**, but the data input into the model is a feature set that calculates fields that evaluate not only the transmitted data, but the relation of these packets over time. Results are expected to be $>$ 60 percent accurate by the model, with iterations of feature selection getting the model to above 80 percent.

- **R3**: Devices have a normalcy in their traffic pattern and a model can be used to ingest traffic that indicates whether the traffic is abnormal or normal. The metrics are the set of accuracy tests passed to the model in relation to the F1 score. Results expect a threshold to be evident in the analysis of the data that indicates what makes traffic abnormal based on feeding abnormal data into the normal dataset.

In this research, the corpus is missing benchmarks in relation to ZigBee findings. There are plenty of benchmarks that indicate the strength or confidence in a model's prediction, but none that relate to ZigBee. This research is contributing to the body of evidence to establish a baseline for future research in this area.

4.5.1 Research Question 1

R1 consists of collecting beacons from devices. Devices perform beaconing to search for available networks to join or transmit the notification that the device is still available in a joined network. Using the procedures above to collect beacons in the lab, analysis will be performed on the beacon data to identify and extract features that can be used for uniqueness based on multiple data points. This answers **R1**'s second subquestion. To answer the first question, a set of devices must already be connected to the network prior to focusing on the shared network key. Three of the devices selected for the network default to the preshared key from the ZigBee Standard [22]. Using the hub interface to set specific network keys, a series of test runs will be performed to check for specific use cases:

- The hub starts with the same pre-shared key as the device. The device is added to the network and the data is captured to validate the behavior. A sequence of tests are preformed with the two different hubs.
- The hub starts with a pre-shared key different from the device. The device is added to the network and the data is captured to review behavior. The same sequence of tests are preformed with the two different hubs.

These tests are collected in different configurations with the heterogeneous devices to be analyzed for differences. Manufacturers may provide different behavior that will be apparent with the data extraction comparisons.

4.5.2 Research Question 2

R2 expands past beacons to collect behavior across all network traffic. There are four different stages of data to collect:

- beaconing,
- network acceptance,
- key change,
- steady state.

This requires collecting the data during each phase separately. Multiple runs are needed to collect enough data for trend analysis. Based on the data, there is an expectation that an evolution of understanding will be found. The details of categorization will evolve:

- the distinction between a FFD and an RFD,
- the distinction between a coordinator and a router,
- the distinction between a sensor and a multi-function device,
- and then the distinction between multi-function devices.

The leverage of machine learning models will aid in the identification based on supervised learning. In each of the four expectations above, feature extraction will be performed to push a model above a 90% accuracy.

4.5.3 Research Question 3

R3 provides an opportunity to expand the identification of devices to find devices that send abnormal traffic. This begins with collecting hours of transmission data to expand the feature extraction, focusing on timing and size. The ability to inject packets allows the researcher to specifically leverage some attacks [11] to determine whether a machine model can be trained to identify abnormal traffic patterns. Starting with a supervised model, the research believes that a semi-supervised model can be built to lessen the training set required through known packet identification.

4.6 Summary

This chapter outlines the tools and techniques that are being leveraged to systematically collect data and process it. The network lab is representative of an automation area that combines motion and action integrated with time. Home environments are miniature representations of industrial areas. This design leverages the simple star topology that allows the traffic to highlight operations without repeating the data transmissions to move it across spans. The tools used to in this research are readily available and industry-standards. With the collected data, the machine learning models are then built that ingest the data to provide results. Listed are the research questions with the expectations for each question that are detailed in the next three chapters.

Chapter 5

ZigBee Device Identification Using Beacons

This chapter uses the laboratory setup in Chapter 4 to answer research question 1 (R1) which is: *Do ZigBee devices have a consistent beaconing pattern that can be uniquely identified from other devices based on timing?* Answering this question required beacon data from every device in the experiment design leveraging the star topology. By processing that beacon data through a machine-learning process, the results reinforce the hypothesis to answer R1.

5.1 Data Collection

The laboratory outlined in Chapter 4 enabled collecting raw data by sniffing the traffic being transmitted. R1 requires the collection of beacon data that are transmitted by ZigBee devices that are not admitted to a network. The device is not in the pairing state and is looking for a coordinator to begin the handshake for admittance.

5.1.1 Collection Procedure

The default action by non-paired ZigBee devices is to begin transmitting beacons on each available channel until a ZigBee coordinator responds. These steps are performed to collect the ZigBee data:

- Open the ZigBee Coordinator device and ensure that all devices are unpaired and removed from the coordinator.

- Unplug all ZigBee Devices.
- Start the ZigBee sniffer device on the laptop within the range of the ZigBee device.
- Run *zdump* from the killerbee framework to validate the ZigBee channel being tested is empty. Capture for two minutes to validate no traffic is captured.
- Restart the *zdump* and *tee* the output to a file that will be processed later. Name the device with the format: deviceName-beacon.pcap.
- Plug in / power on the ZigBee Device. Validate that *zdump* is seeing the packets.
- Run the capture for 20 minutes and then stop the capture.
- Unplug the ZigBee device.

This process can be repeated as many times as necessary with as many devices as required for the experiment. The 20 minute timeframe creates approximately 30,000 beacons per device. When running the the model in the future steps, if more data is required that number can be pumped up. At the end of data capture, the file format is a standard packet capture format (.pcap) that can be investigated and reviewed with common network tooling.

5.2 Data Analysis

The standard packet capture format must be ingestible by the machine learning model for the next step. Figure 5.1 is the standard process for machine learning. This is done in a pipeline that converts the standard packet capture format to a comma separated value format that allows the *pandas* library in Python3 to perform most of the data preparations (see Chapter 4 for details).

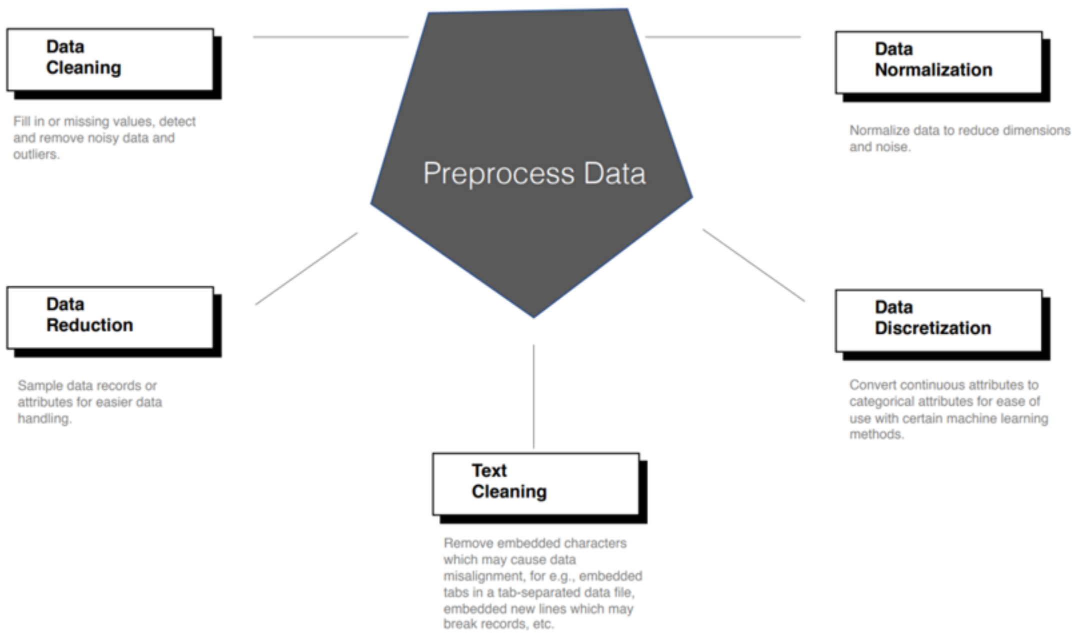

Data Cleaning
Fill in or missing values, detect and remove noisy data and outliers.
Preprocess Data
Data Normalization
Normalize data to reduce dimensions and noise.
Data Reduction
Sample data records or attributes for easier data handling.
Data Discretization
Convert continuous attributes to categorical attributes for ease of use with certain machine learning methods.
Text Cleaning
Remove embedded characters which may cause data misalignment, for e.g., embedded tabs in a tab-separated data file, embedded new lines which may break records, etc.

Figure 5.1: Data Preparation [7]

5.2.1 Data Conversion

To perform the first step is converting from the standard packet capture to the comma-separated values format. There was no industry tool available for research, so a tool was created that uses Python3. The killerbee and pyshark libraries are utilized for ease of extraction and to minimize any deviations from the ZigBee and 802.14 standard. Algorithm 1 is straight-forward for field extraction.

During extraction, there are packed fields within the transmission stored as hex values. These fields are broken down into individual bit values represented as integers to improve the results of the machine learning algorithm. Instead of applying data discretization and data normalization further down the pipeline, this converter separates those packed fields into their bit representations. During the conversion fields are unpacked as listed in Table 5.1. In addition, there are a few fields that are added to the dataset that are calculated based on the data present within the packet. These fields are listed in Table 5.2.

The manufacturer field is a lookup based on the MAC address published by the source of the message in the third layer of the packet. Using that master MAC address list, that

Algorithm 1 PCAP Parser

```
1: procedure PARSER($a$)                                ▷ a is the actual packet
2:     $result \leftarrow null$
3:     if $a$ is a ZigBee Packet then
4:         $b \leftarrow a$
5:         if $b$ has a MAC layer then
6:             for $c$ for each attribute in $b$ do
7:                 $result \leftarrow c$
8:             end for
9:         end if
10:        if $b$ has a Beacon layer then
11:            for $c$ for each attribute in $b$ do
12:                $result \leftarrow c$
13:            end for
14:        end if
15:        if $b$ has a Network layer then
16:            for $c$ for each attribute in $b$ do
17:                $result \leftarrow c$
18:            end for
19:        end if
20:    else
21:        Skip packet and do not process
22:    end ifConduct Lookup of MAC address
23: end procedure
```

Table 5.1: Unpacked Fields

frame control field	hex	Second Layer
destination PAN id	hex	Second Layer
frame payload	hex	Second Layer
security frame	hex	Third Layer
discovery	hex	Third Layer

Table 5.2: Computed Data

entropy	double	entropy of the data field
manufacturer	string	the manufacturer of this device
device Identifier	string	the actual unique device name

manufacturer name is pushed into the row of values. If a value does not exist, that field is left blank. The match with the longest MAC address matching is used for the field. The entropy is also calculated based on the encrypted data field being passed. For beacon data, this field should be zero and is not used for this question. For the full list of fields extracted from the PCAP, see Appendix A.

5.2.2 Data Preprocessing

After the previous step, the data is now in a CSV format ready for ingestion into the Jupyter Notebook for the python3 *panda* library. The data are imported and saved as a *panda array* for manipulation throughout the rest of the process.

5.2.2.1 Data Cleansing

The converter provides a header line that provides field headers for every column in the dataset. This row is removed and saved for later during analysis. Any columns that are all blanks are removed from the dataset. A blank column indicates that the packet did not have data in that field or that field was not available within the Beacon packet type.

5.2.2.2 Data Reduction

Only Beacon packets are kept. This ensures that packet captures with other packet types are removed to keep the dataset solely focused on beacons. This allows the Data Conversion step to process any ZigBee packets.

5.2.2.3 Text Cleansing

The manufacture field is separated for the categorization solution of the model. Source and Destination addresses are removed for the beacon analysis.

5.2.2.4 Data Normalization

The remaining fields that are hexadecimals are converted to integers. The machine learning categorization functions leverage math operations that depend on integers. Leaving fields as hexadecimals resulted in large outliers in the matrix. Changing the data type improved convergence times when the models were building.

5.2.2.5 Data Discretization

Each numerical column is run through a Normalizer function. The StandardScaler function was attempted for each model, but it did not improve the results.

5.3 Results

5.3.1 Model Building

The data were passed into a correlation heatmap, which highlighted 23 fields that were used to build the model. In this approach, a supervised learning model is leveraged because we know what the data should be based on our environment. Referring to Figure 5.2, this research leveraged classification. The sklearn library built for python3 supports multiple classification models that are leveraged in this research. They key selection of classifiers is the requirement to be able to handle both numerical and categorical data, as the data types within the beacon information have both.

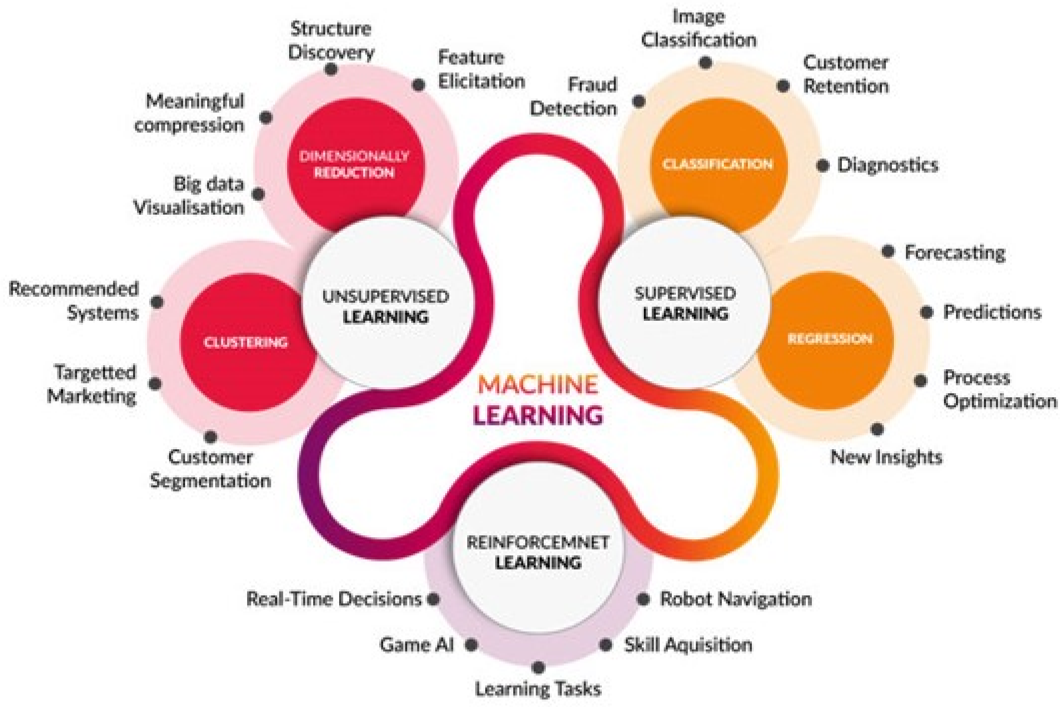


Figure 5.2: ML Options [7]

The following models were used to generate a potential solution:

- Decision Tree[23] - This classifier is a simple multi-class classifier that uses a series of if/else decisions to build its model. It is a foundation for classification and used as a base-line.

- Random Forest[23] - This classifier serves as an extension to the Decision Tree that adds in a randomness to increase prediction based on averages.

- K-Neighbors[23] - This classifier uses a query to find the nearest point and allows the researcher to determine the suppression of noise through the selection of the k value. This can reduce the classification boundary distinction, but could be important during classification like answering R1.

- AdaBoost[23] - This classifier acknowledges that there are weak learners and provides a method for difficult-to-predict results. This was chosen to account for the small variation in beacon data.

- GradientBoost[23] - This classifier builds on the AdaBoost model and uses a sorting mechanism to control overfitting. With the deviations between beacons being small, this method was chosen to highlight those small differences.

The data was divided into a training set and a testing set with an 80-20 split. The *GridSearchCV* function was run on each classifier to systematically determine the appropriate parameters for each model based on the data provided. Once the optimal parameters were found, each classifier was fit and ready for results.

5.3.2 Model Results

Each model generated an accuracy, precision, recall, and F1 score to show the performance of each model with this dataset listed in Table 5.3.

Table 5.3: Machine Learning Results For Identifying Devices Based Off Beacons

Model	Test Accuracy	Test Precision	Test Recall	Test F1
Random Forest	75.6741	83.2233	75.6741	77.8191
Decision Tree	83.5425	73.4368	73.4368	76.1143
K-Neighbors	71.1534	79.0542	71.1534	73.7083
AdaBoost	72.3957	77.1946	72.2589	74.1954
GradientBoost	77.6893	78.1926	78.9922	77.2973

The models were tuned using the *GridSearchCV* module from sklearn. The best parameters were selected based on the accuracy of each model's performance. Each model has different parameters used to tune the model and are illustrated in the tables below. The best factor chosen is highlighted in **bold** in Tables 5.4, 5.5, and 5.6.

To visualize the data, a correlation heatmap was generated after the data processing was complete. To generate the heatmap, the sklearn function *feature_importances_* was

Table 5.4: Model Tuning Parameter 1

Model	Name	Params
Random Forest	estimators	5,8,10,**13**,17,22,28,30,35,40
Decision Tree	splitter	**best**, random
K-Neighbors	n_neighbors	3,5,10,**13**,17,22,28
AdaBoost	base_estimator	DecisionTree,**RandomForest**
GradientBoost	loss	**deviance**, exponential

Table 5.5: Model Tuning Parameter 2

Model	Name	Params
Random Forest	criterion	**gini**,entropy
Decision Tree	criterion	**gini**,entropy
K-Neighbors	weights	uniform,**distance**
AdaBoost	algorithm	**SAMME**,SAMME.R
GradientBoost	criterion	friedman_mse,squared_error,**mse**,absolute_error

used with an *ExtraTreesClassifer* having a number of estimators equal to 50. The following data points had higher than 60% correlations listed in Table 5.7.

5.4 Analysis

5.4.1 Research Question 1

Research question 1 (R1) is: *Do ZigBee devices have a consistent beaconing pattern that can be uniquely identified from other devices based on timing?* The results of the model are unable to perform above a 83% accuracy based on the device. Utilizing all available data within the beacon packet, there is not enough data to distinguish devices. Every device utilized follows the RFC standard and is within the tolerances given. Collecting additional data did not impact the accuracy nor did increasing the testing data percentage.

The RQ1 also leads the question, *Does the introduction of a new ZigBee device in a heterogeneous environment cause the network to switch network keys?* The results are no. It causes the end device to be pushed the network key currently being used by the

Table 5.6: Model Tuning Parameter 3

Model	Name	Param
K-Neighbors	algorithm	ball_tree,**kd_tree**,brute

Table 5.7: Correlation Heatmap

	Frame Type	CMD ID	MAC Payload	Payload Length	Beacon Payload	Beacon Length
Frame Type	1	62%	78%	72%	71%	73%
CMD ID	92%	1	94%	77%	73%	74%
MAC Payload	89%	64%	1	84%	92%	91%
MAC Payoad Length	93%	63%	99%	1	91%	89%
Beacon Payload	91%	63%	97%	93%	1	96%
Beacon Length	91%	64%	98%	90%	91%	1

network once it is enrolled. The ZigBee device can begin with the 2009 ZigBee password standard to start beaconing. Once the device is enrolled in the network, the coordinator sends out an immediate transport key through the APS layer to update the network encryption key. The ZigBee standard allows multiple networks on the same band; it is very inefficient for all devices involved. The most efficient answer is to put the device in the same security posture that the network coordinator is running for everyone else. By sharing the network key through the APS layer, this requires the end device to be accepted into the network and, therefore, trusted. Malicious use cases to abuse this key rely on tricking the coordinator to accept the imposter device. The model developed to support R1 adds a data point for the decision to reduce imposter devices.

R1 rephrased from a different point of view is *Can we fingerprint the device based on beaconing patterns?* The fingerprint that is available between the exposed layers provides a few data points. Although the provided MAC has governing guidance on how it is formed, that does stop threats from cloning or imposing as those MAC addresses. The model that was built is accurate to fingerprint based on transmission characteristics. Because these characteristics are limited to only beaconing signals, the model is able to identify the device manufacturer. The research points to this being based on chipsets, as the prime discriminator is the time between beacons themselves.

5.5 Summary

This set of experiments outlines leveraging beacon data to identify devices that are beaconing to the coordinator asking for admittance to the network. This is the first step in getting any device to communicate along the ZigBee backbone to allow data to be repeated to anyone. The pairing process was initiated once the device was accepted at the coordinator. By using a machine-learning model, we can begin to identify those devices that do not belong in the network. Due to the limited data available within the beacon data itself, there was not enough variance in the transmitted data to fingerprint accurately (above a 90%) what type of device is asking for admittance through the open airwaves traffic. A multitude of analysis was done, pulling apart state fields and attempting to compute additional data fields relevant to the device operations. Although a systematic process was used to perform data analysis, the limited data fields within the beacon packets prevented a model from converging with a set of false negatives low enough to be acceptable.

Chapter 6

ZigBee Device Identification Using Network Traffic

This chapter uses the laboratory setup in Chapter 4 to answer research question (R2) which is *Can we identify device types based on network layer traffic using machine learning (ML)?* This experiment requires a network complete with paired devices where the network traffic can be sniffed, parsed, prepared, and fed to a trained model. The proximity research[8] highlights the proliferation of devices where finding devices are a challenge spread out in an industrial environment. Answering this research question allows the targeted search and removal of devices when they are identified as not allowed on the network.

6.1 Data Collection

The laboratory outlined in Chapter 4 enabled collecting raw data by sniffing the traffic being transmitted. R2 requires the collection of network data that are transmitted by ZigBee devices that are part of a network. The device has already been paired on the network, has exchanged an initial set of network keys, and is performing routine operations based on the device's operations.

6.1.1 Collection Procedure

The default action by paired ZigBee devices is to transmit packets on the channel it was admitted to the network. All commands are embedded in the network packet at the encrypted layer based on the application layer of the RFC. To perform these tests, the data are collected using the two coordinators independently. Each coordinator has automated functions that allow the hub to transmit commands to the devices to activate their functions. In order to get the functionality from each device for testing, a timer was set up for each device that allowed automation to occur. The details of timers are listed in Table 6.1.

Table 6.1: Timers For Automation

electrical outlet	switch state	15 minutes
light bulb	switch state	13 minutes
color light bulb	change color	13 minutes
blinds	open/close	13 minutes
robot vacuum	clean room	12 minutes

In order to automate the motion detector and the contact sensor, A vacuum robot was used to trigger both the motion sensor and the contact sensor. Half of the contact sensor is attached to the vacuum enabling the sensor to trigger every time it touched the other half connected to the baseboard of the room. While running the vacuum, it also triggers the motion sensor setup in the room. Triggering the vacuum robot is done through a timer setup in the iRobot app not incorporated into the coordinator due to lack of proprietary protocol usage. These steps are performed to collect the ZigBee network traffic:

- Open the ZigBee Coordinator device and unpair all items listed.
- Plug in each device and conduct the pairing process in accordance with the device.
- Verify in the ZigBee Coordinator that all devices listed in Chapter 4 are paired.

- Start the ZigBee sniffer device on the laptop within the range of the ZigBee devices.
- Run *zdump* from the killerbee framework to validate the ZigBee channel being tested has traffic.
- Restart the *zdump* and *tee* the output to a file that will be processed later. Name the device with the format: hubName-lengthofCapture.pcap.
- Run the capture for 240 minutes and then stop the capture.

This process can be repeated as many times as necessary with as many devices as required to collect the data. At the end of data capture, the file format is a standard packet capture format (.pcap) that can be investigated and reviewed with common network tools.

6.2 Data Analysis

The same data processing program is used from Chapter 5 to convert the packet capture to the comma-separated values format. The small differences are listed below.

6.2.1 Data Conversion

After following the data conversion from pcap to csv, the converter addresses the fields that are unpacked during the conversion as listed in Table 6.2. Additionally, there are a few fields that are added to the dataset that are calculated based on the data present within the packet. These fields are listed in Table 6.3.

The manufacturer field is a lookup based on the MAC address being published by the source of the message in the third layer of the packet. Using that master MAC address list, that manufacturer name is pushed into the row of values. If a value does not exist, that field is left blank. The match with the longest MAC address matching is used for the field. The entropy is also calculated based on the encrypted data field being passed. For entropy could change based on the security setting of the coordinator

Table 6.2: Unpacked Fields

source	int	unique id of who sent the message
dest	int	unique id of who should receive the message or broadcast
frame type	int	identifier on the type of packet
discovery	int	identifier on the discovery bits as a whole number
extended src	bool	if long or short addressing is being used for src
extended dst	bool	if long or short addressing is being used for dest
security	bool	if security is on or off
radius	bool	whether radius server is being used
protocol version	int	version being transmitted
sequence number	int	positive value of sequence for reliability
multicast	bool	if this is a multicast call or not
extended src64	int	the long MAC of the src as an integer
extended dst64	int	the long MAC of the dst as an integer
data length	int	length of the encrypted data portion
data	hex	data being transported
sec frame	int	the security counter of this packet

Table 6.3: Computed Data

entropy	double	entropy of the data field
manufacturer	string	the manufacturer of this device
device Identifier	string	the actual unique device name

managing the network. The RFC published in 2018 recommends running the networks in encrypted mode (level 4 or above) to prevent eavesdropping, replay-attacks, and other activity. These experiments are conducted with encryption turned on as should be the standard for ZigBee deployments.

6.2.2 Data Preprocessing

After the previous step, the data is now in a CSV format ready for ingestion into the Jupyter Notebook for the python3 *panda* library. The data are imported and saved as a *panda array* for manipulation throughout the rest of the process.

6.2.2.1 Data Cleansing

The converter provides a header line that provides field headers for every column in the dataset. This row is removed and saved for later during analysis. Any columns that are all blanks are removed from the dataset. A blank column indicates that the packet did not have data in that field or that field was not available within the Beacon packet type.

6.2.2.2 Data Reduction

Only network packets are kept. This ensures that packet captures with other packet types are removed to keep the dataset solely focused on the network layer. This allows the Data Conversion step to process any ZigBee packets and allow the model preparation steps to use the data applicable to that experiment.

6.2.2.3 Text Cleansing

The manufacture field is separated for the categorization solution of the model. Source and Destination addresses are removed for the each row of data for analysis. If a model was being created that leveraged traffic flow patterns, these fields could be kept. The focus of this question is on patterns of life of traffic being transmitted, not related to who is communicating with whom across the network.

6.2.2.4 Data Normalization

The remaining fields that are hexadecimals are converted to integers. The machine learning categorization functions leverage math operations that depend on integers. Leaving fields as hexadecimals resulted in large outliers in the matrix. Changing the data type improved convergence times when the models were building.

6.2.2.5 Data Discretization

Each numerical column is run through a Normalizer function. The StandardScaler function was attempted for each model, but it did not improve the results. Secondly, there are multiple fields that are not really an integer value, but a state encoded as a number. A transformer was applied to these fields in order to perform a OneHotSwap expanding that column into multiple columns to account for the state. The following fields were transformed.

- Frame Type
- MAC CMD ID
- Protocol Version

6.3 Results

6.3.1 Model Building

The data were passed into a correlation heatmap, which highlighted the 38 fields that were used to build the model. In this approach, a supervised learning model is leveraged because we know what the data should be based on our environment. Referring to Figure 5.2, this research leveraged classification. The sklearn library built for python3 supports multiple classification models that are leveraged in this research. For consistency, the same models used for R1 were used for R2. The are listed in Chapter 5. The data was divided into a training set and a testing set with an 80-20 split. The models were tuned using the *GridSearchCV* module from sklearn. The best parameters were selected on the basis of the accuracy of each model's performance. Each model has different parameters used to tune the model and are illustrated in the Tables 6.4, 6.5, and 6.6. The best chosen factor is highlighted in **bold**.

Table 6.4: Model Tuning Parameter 1

Model	Name	Params
Random Forest	estimators	5,8,10,13,17,22,28,30,**35**,40
Decision Tree	splitter	**best**, random
K-Neighbors	n_neighbors	3,5,10,**13**,17,22,28
AdaBoost	base_estimator	**DecisionTree**,RandomForest
GradientBoost	loss	log_loss, exponential

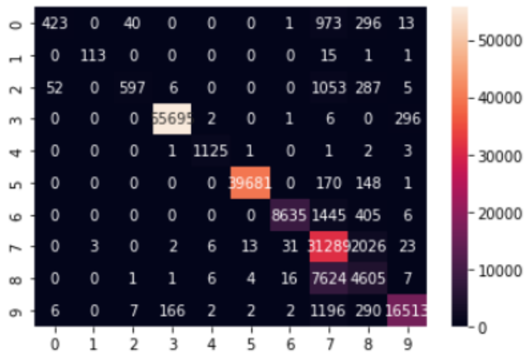
Table 6.5: Model Tuning Parameter 2

Model	Name	Params
Random Forest	criterion	**gini**,entropy
Decision Tree	criterion	**gini**,entropy
K-Neighbors	weights	uniform,**distance**
AdaBoost	algorithm	**SAMME**,SAMME.R
GradientBoost	criterion	friedman_mse,squared_error,**mse**,absolute_error

6.3.2 Model Results

Each model generated an accuracy, precision, recall, and F1 score to show the performance of each model with this dataset. See Figures 6.1, 6.2, 6.3, 6.4, and 6.5.

```
Best Parameters: {'n_estimators': 35}
Random Forest Training Accuracy is: 0.904957
        Precision is: 0.909892
        Recall is: 0.904957
        F1 is: 0.898902
```

(a) 

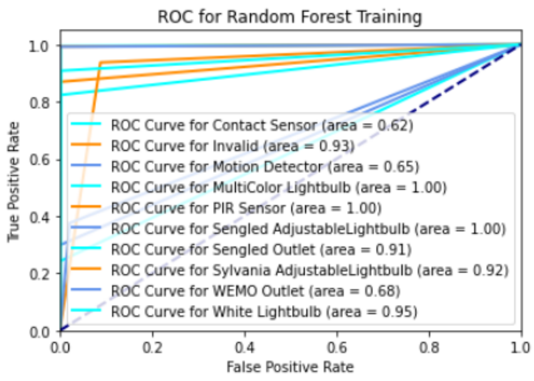
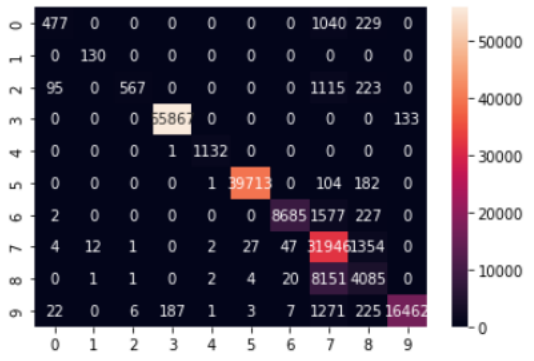
(b) 

Figure 6.1: Network Random Forest Results

Table 6.6: Model Tuning Parameter 3

Model	Name	Param
K-Neighbors	algorithm	**ball_tree**,kd_tree,brute

```
Decision Tree Training Accuracy is: 0.907169
          Precision is: 0.913084
          Recall is: 0.907169
          F1 is: 0.899284
```

(a) 

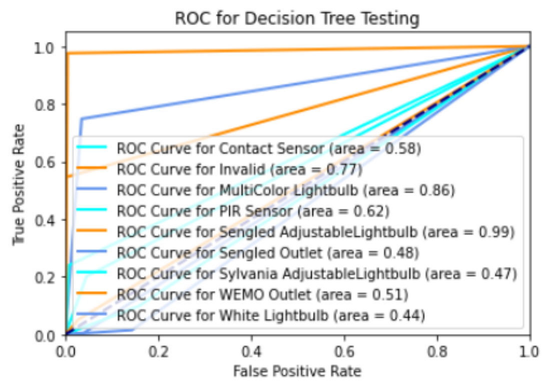
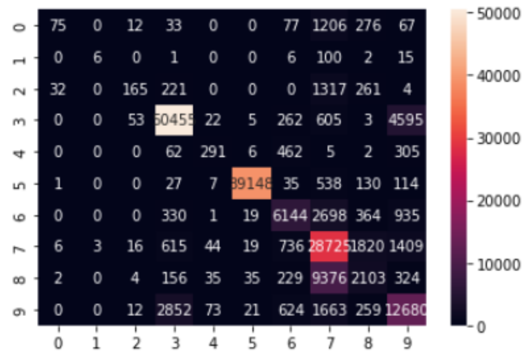
(b) 

Figure 6.2: Network Decision Tree Results

```
KNN Training Accuracy is: 0.797258
          Precision is: 0.793604
          Recall is: 0.797258
          F1 is: 0.782112
```

(a) 

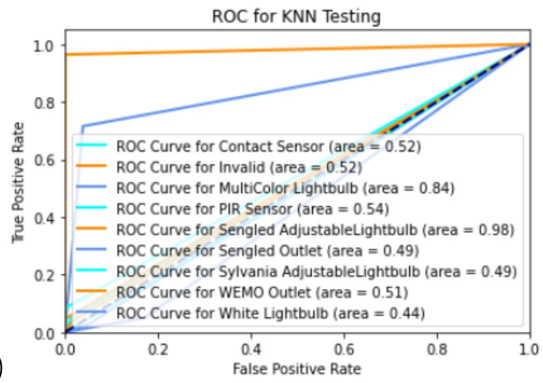
(b) 

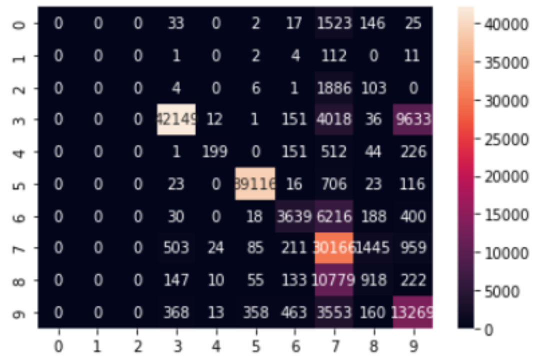
Figure 6.3: Network KNN Training Results

6.4 Analysis

6.4.1 Research Question 2 Results

The second question (R2) is *Can we identify device types based on network layer traffic using machine learning (ML)?* The results show yes. There are over 59 data points that

```
AdaBoost Training Accuracy is: 0.738310
        Precision is: 0.759669
        Recall is: 0.738310
        F1 is: 0.721722
```

(a) 

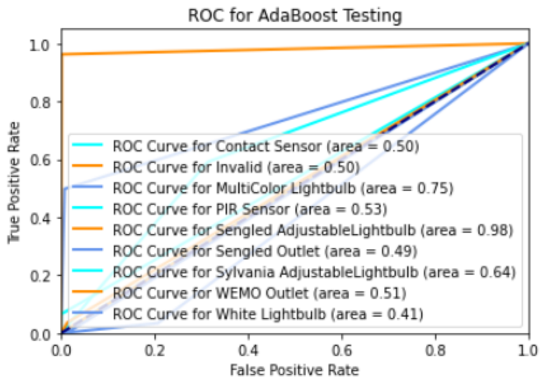
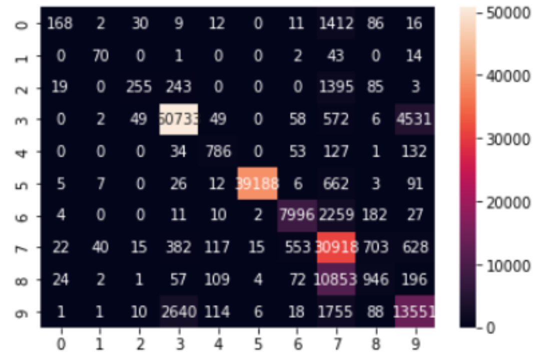
(b) 

Figure 6.4: Network AdaBoost Training Results

```
GradientBoost Training Accuracy is: 0.824742
        Precision is: 0.823930
        Recall is: 0.824742
        F1 is: 0.804161
```

(a) 

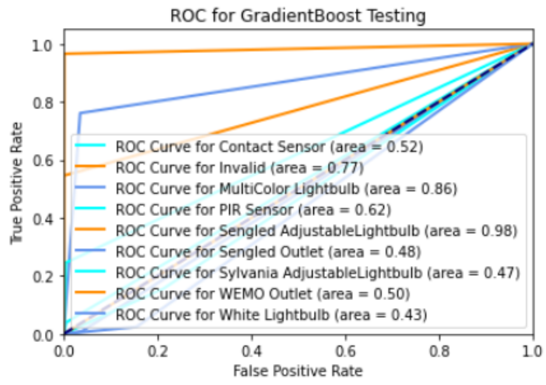
(b) 

Figure 6.5: Network Gradient Training Results

are broadcast, with eight of those fields being bit-fields that are broken apart to have over 70 data points for discrimination. Leveraging multiple classification models, a model with 90% F1 score can ingest network traffic from a device admitted to the network and determine the type of device resembling transmission. The best model was the Random Forest Model. Decision-Tree was the second best with an 80% score with the other models coming in lower than 70%. The fields with MAC addresses are removed to prevent the overfitting that happens during training.

The RQ2 also gives the question, *Are there enough indicators at the network layer to provide enough uniqueness for a device to be identified into categories?* There are 29 fields that have a high correlation factor to include in the machine learning model that generates the above score. Without using the MAC address fields, device manufacturers are not distinguishable enough to have a success rate of more than 73%. Modifying the categorization into device types trains the model to identify devices according to type. The additional question is *can we fingerprint the device based on network layer traffic?* The result is yes if the devices are separated into categories. There are multiple devices that have almost identical traffic patterns due to their function. For example, the contact sensor and motion detector provide nearly the same traffic patterns when used for similar use cases. Once of these devices placed in a high-use area resemble the same traffic patterns when the situations are reversed. In one experiment, the contact sensor was activated repeatedly while the motion detector was placed in a remote spot that did not have frequent activations. Then in a separate experiment, the contact sensor was not activated frequently but the motion sensor was placed in a high-traffic area. An attempt was made to distinguish between these two models, but the data fields were not distinguishable enough to train a model that could accurately identify the difference in these two device types.

6.5 Summary

The experiments validate the research questions, providing enough information within the network layer to categorize devices based on network traffic without leveraging the source/destination MAC addresses. Collecting the data requires a network sniffer within range of the communicating devices. The data sniffed are stored as a PCAP, then converted to a CSV to allow the sklearn library to ingest those data for feeding the model.

Most of the supervised learning models available in the sklearn library are leveraged to determine the best solution for this experiment. The simpler the device, the more false positives that the devices create with other devices of like simplicity. The more complex the device (in this case the multicolor lightbulbs and coordinators) give the most accurate results with the fewest false positives.

Chapter 7

Identifying Abnormal Zigbee Device Behavior Using Network Traffic

By leveraging the models built in Chapter 6, can these same models identify traffic is abnormal from the traffic that it was trained on? Using the same lab setup as before, this experiment adds in the additional antenna to inject packets into the network with an identifiable MAC address to determine if the models can detect when data is outside of its trained bounds.

7.1 Research Question 3

The third question (R3) is *Given an inventory of items authorized to communicate on a ZigBee network, can traffic analysis of the network layer identify devices that are sending abnormal traffic?* This question involves defining abnormal traffic as traffic outside of a learned range of expected traffic patterns. The training models were trained using 4-hour traffic patterns to train the model. Once trained, the model was able to identify when traffic being transmitted was outside the bounds of normal traffic size. The categorization for this model was binary to "normal or abnormal." Determining abnormal is subject to providing a representative model that includes fluctuations in time. In this model, time of day or day of week were not a factor in developing this model. Based on the results of these experiments, a model could be trained over a duration period that could include those routine activities.

7.2 Data Collection

The data collection presented in Chapter 6 is constant during this experiment. To add in the "abnormal" traffic, the apimote device is utilized to inject traffic into the network that modifies the specific fields in the network layer. Once the network capture begins, Algorithm 2 is started that runs while the captures occur:

Algorithm 2 Abnormal Packet Generator

```
1: procedure GENERATOR($devices$)                    ▷ the list of devices to mock
2:     $devices \leftarrow ListofDevices$
3:     for ever do
4:         for  $c$ in $devices$ do
5:             $packet \leftarrow defaultpacket$
6:             for $c$ of each attribute in $b$ do randomize packet data
7:             end for
8:         end for
9:         sleep(random(10,12))
10:    end for
11: end procedure
```

7.3 Data Analysis

The same process presented in Chapter 6 is leveraged to create the model. In this instance, the model is trained using a reverse majority vote technique that builds a model for each device for the training data. The algorithm flow checks the results of each model per device-type. The models determine a binary solution: is within the bounds or outside the bounds. The data is checked with each model in series and if any model identifies the model data as true the algorithm returns false indicating the data is normal. Any data that makes it through every model without any model giving a true, the data is abnormal for the network.

7.3.1 Data Reduction

In order to account for field data being definitions and not numbers, OneHotEncoding was performed on each field that represented a bit and was not an integer value. This greatly increased the column count, but was necessary to prevent the algorithms from associating standard deviations with unintentional numbers. Since the abnormal data could be anything, a value was added to each OneHotEncoding in case the data in the testing collection did not match the training collection.

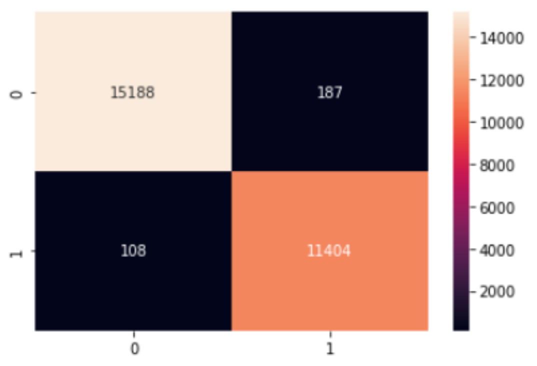
7.4 Results

7.4.1 Model Building

In this experiment, the models are built for a binary classification of each device type highlighting if it is valid or not. The label column for result is swapped with a 1 for matching and a 0 for non-matching during each model build of the training dataset. After the models were built, they were used in sequence to test the accuracy in the testing dataset. If any of the models returns a 1 for a match, Algorithm 3 assumes a match is found and returns the positive indicator. The results are listed in Figures 7.1, 7.2, 7.3, 7.4, and 7.5.

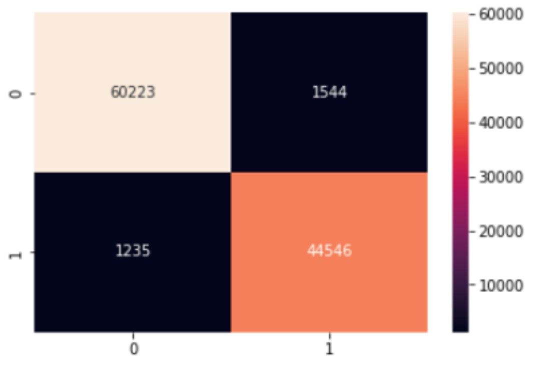
Algorithm 3 Minority_Vote

```
1: procedure CLASSIFIER($a$,$classList$) ▷ a is the packet to be categorized and classList is the list of classifiers
2:     for $option$ for each attribute in $classList$ do
3:         if $option$ returns 1 then
4:             return true
5:         end if
6:     end for
7:     return false
8: end procedure
```



```
)ecision Tree testing Accuracy is: 0.989028
       Precision is: 0.989055
       Recall is: 0.989028
       F1 is: 0.989033
```

Figure 7.1: Abnormal Behavior Decision Tree Results

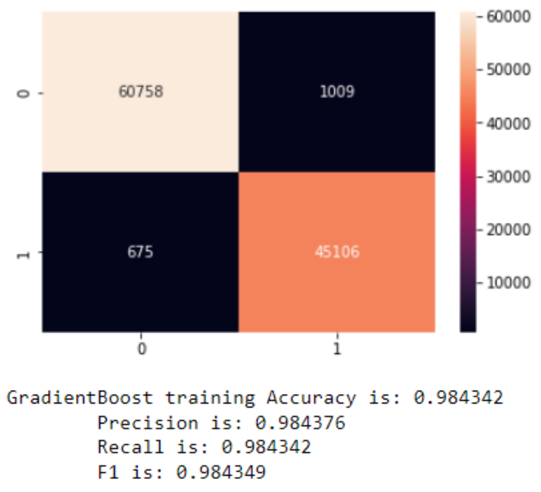


```
AdaBoost training Accuracy is: 0.974160
        Precision is: 0.974199
        Recall is: 0.974160
        F1 is: 0.974171
```

Figure 7.2: Abnormal Behavior AdaBoost Results

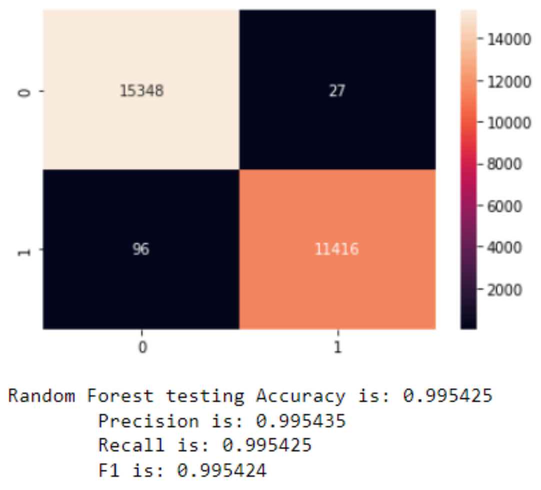
7.5 Analysis

When testing *R3*, the introduction of malicious activity allowed the reduction of accuracy in identifying malicious events when those malicious events were stuffed into fields omitted during training. To regain the ability to capture *R3* fidelity, the fields were retrained in the model to catch future abuse. The correlation numbers used in the Model Building step are skipped and all fields are used (which results in all columns being leveraged for training). As the models were trained for each device in a binary fashion, the model



```
GradientBoost training Accuracy is: 0.984342
        Precision is: 0.984376
        Recall is: 0.984342
        F1 is: 0.984349
```

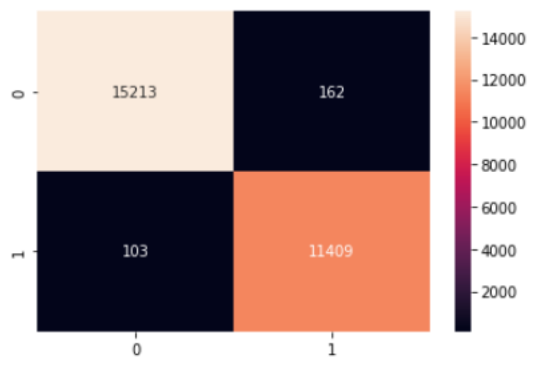
Figure 7.3: Abnormal Behavior GradientBoost Results



```
Random Forest testing Accuracy is: 0.995425
        Precision is: 0.995435
        Recall is: 0.995425
        F1 is: 0.995424
```

Figure 7.4: Abnormal Behavior Random Forest Results

results were in the high 90s for each accuracy. The categorization model reduced accuracy as more types of devices were incorporated into the model. In this approach, a majority-vote model was inverted. A model is built for each type of device that results in a binary decision. After building these models, the models are put in sequence and if any determine a positive the categorization is accepted. This provides a model that allows independent models to be trained, retrained, and inserted into the pipeline allowing flexibility and growing of the models approach. The false negatives are where the research drives the techniques for stealth. This results in an accuracy of the minority-vote model to be 93%.



```
KNN testing Accuracy is: 0.990144
        Precision is: 0.990160
        Recall is: 0.990144
        F1 is: 0.990147
```

Figure 7.5: Abnormal Behavior K-Nearest Neighbor Results

Reviewing the false negatives, the abnormal traffic that gets through the model is those that communicate traffic in traffic sizes similar to normal packets. In the experimental setup, if the algorithm is altered to ensure that the values are always outside the bounds of normal traffic for all devices, the accuracy increases. However, this is not realistic for the field setup. Adjusting the algorithm to ensure that the payloads are always less than two standard deviations outside the average decreases the accuracy from 93% to 84%. Any device that communicates abnormal traffic in the bounds of devices on the network will create false negatives.

7.6 Summary

Testing abnormal behavior within an environment can be challenging without the context of the device. This experiment uses the knowledge gained from RQ2 to see if a model can predict abnormal behavior solely based on traffic patterns. The resulting model provides insight to traffic that is outside the bounds of what has been trained, but fails to highlight traffic that resembles normal network traffic with subtle alterations.

Chapter 8

Conclusion

The research provided in the previous chapters provides a systematic approach to building models for industrial areas. The results of R1, R2, and R3 experiments listed in above provide evidence to validate those research questions. Those results combine to an understanding for the community to move forward with ZigBee implementations and security options for more complex ZigBee configurations.

8.1 Research Findings

Security solutions are in high demand for IoT infrastructure that provides classification, detection, and throughput. When we have traffic available, machine learning classifiers are a proven approach to detect network anomalies for inspection. In the approach shown here, a testbed environment of a smart home provided the complement of devices that represent real-world network characteristics. By using this approach, automation can ingest the near real-time traffic and provide immediate alerts to new devices and activities within the network. To improve the classification accuracy, a majority-voting scheme can be applied that leverages independent classifiers to provide better fidelity based on the characteristics of the device. This lowers the rejection rate and increases the confidence in the models that are used for decisions.

The most recent research has been on the location and discovery of devices based on their RF signature [24]. This vector is based on the detection of signals and the pinpointing of where the source comes from. However, [25] indicates that there is an increasing vector of items masquerading uninvited into the physical locations. The finding of these devices is based on some roving physical system capable of detecting a signal, then using the RF signature [24]. To find these devices, there are already coordinators (and routers) that are FFDs that are plugged into the SCADA backbone for management that have access to capture and record signals. This research provides the evidence to leverage these already-connected FFD to feed a security system to identify and find devices that should not be within the environment.

Real-world network traffic is normally ingested in a control center where logs are filtered with regular expression rules and complex decision-making for alerting. While this capability has not show effective in the ZigBee device realm, this approach enables captured traffic data to be passed in to this research for ingestion into other operations. In this approach the assumption was a centralized logging center was not available. This is not true in the larger industrial areas but continues to leave a gap where captured network traffic is unable to be pulled into a single pane of glass. Several authors discussed the vulnerabilities inherent in the implementation of the protocol, but few tackled the approach to provide systematic detection.

8.2 Contributions

The vulnerability landscape grows with the addition of every new ZigBee device. While the risk in the home may be localized, the risk to the large business and the industrial base continues to multiply. The current approach to work-life balance is encouraging more work-from-home opportunities that transfer localized risk to the global platform. This research adds to the body of knowledge with the following:

- artifacts that show where the ZigBee protocol provides information to identify device types,

- recommendations to changes in the ZigBee protocol to improve its cyber resilience,
- Recommendations for implementing ZigBee within the ecosystem to limit the threat surface of the devices.

These artifacts allow the protocol to be iterated upon for increased security enhancements. Devices can be identified solely off of network traffic enabling the detection of rogue devices. This drastically improves the security posture and allows the defenders to gain an advantage in the cat and mouse game of security.

8.3 Limitations

This research is based on available ZigBee Devices on the current market using an open-source ZigBee protocol. Many of the devices being used in industry are proprietary requiring some paid knowledge of the framework. However, the network layer remains the same for those proprietary implementations as the application layer is where the special functionality is added. It still must be compliant with the standard even when proprietary. The devices used were low-functioning similar to the field, but no specialized devices where used. Taking this approach to specialized industries like healthcare can uncover other use cases impacting other critical areas. No experiments were done leveraging unencrypted payloads. While the protocol could allow it, neither of the coordinators used during these experiments were put into the lower security model. The Amazon Alexa Hub will not accept that setting (providing manufacturer safeguards preventing it). Model building can be intensive and some algorithms like the K-Neighbors model consume lots of time with the number of estimators increasing. These experiments limited the number of estimators to less than 40. Online Jupyter Notebooks would time-out due to the long runtime requiring local computations. Leveraging more estimators could provide better results for classification, but were outside the resources available during this research.

8.4 Future Work

The devices levied in this research were limited to available devices within the Smart Home realm. Industrial devices should be incorporated into the training set to allow independent models to account for variations in device operations for custom installations in industrial areas. Showing this capability begs for plugins to be created for current security and incident event management tools where Ethernet transport layers can move data in bulk to filtering centers.

References

[1] C. Shao, Y. Kim, and W. Lee, "Zero-effort proximity detection with zigbee," in *IEEE Communications Letters*, vol. 24, Sept. 2020, pp. 2047–2050. DOI: 10.1109/LCOMM.2020.2998526.

[2] S. Khanji, F. Iqbal, and P. Hung, "Zigbee security vulnerabilities: Exploration and evaluating," in *2019 10th International Conference on Information and Communication Systems (ICICS)*, 2019, pp. 52–57. DOI: 10.1109/IACS.2019.8809115.

[3] S. Karnouskos and F. Kerschbaum, "Privacy and integrity considerations in hyperconnected autonomous vehicles," in *IEEE*, vol. 106, 2018, pp. 160–170.

[4] A. Burg, A. Chattopadhyay, and K. Lam, "Wireless communication and security issues for cyber-physical systems and the interet-of-things," in *IEEE*, vol. 106, 2018, pp. 38–60.

[5] D. Schlette, M. Caselli, and G. Pernul, "A comparative study on cyber threat intelligence: The security incident response perspective," in *IEEE Communications Surveys and Tutorials*, vol. 23, 2021, pp. 2525–2556.

[6] E. Ronen, C. OFlynn, A. Shamir, and A. Weingarten, "Iot goes nuclear, creating a zigbee chain reaction," Dalhouse University, Halifax, Canada, 2020, pp. 1–18.

[7] D. Knuth. "Zigbee cc2531 smart home usb adapter." (2019), [Online]. Available: https://hackaday.io/project/163487/instructions.

[8] X. Zhou, A. Hu, G. Li, L. Peng, Y. Xing, and J. Yu, "A robust radio-frequency fingerprint extraction scheme for practical device recognition," in *IEEE IoT Journal*, vol. 8, 2021.

[9] H. Pirayesh and H. Zeng, "Jamming attacks and anti-jamming strategies in wireless networks: A comprehensive survey," in *IEEE Communications Surveys 'I&' Tutorials*, vol. 24, 2022, pp. 767–809.

[10] X. Cao, D. M. Shila, Y. Cheng, Z. Yang, Y. Zhou, and J. Chen, "Ghost-in-zigbee: Energy depletion attack on zigbee-based wireless networks," in *IEEE Internet of Things Journal*, vol. 3, 2016, pp. 816–829. DOI: 10.1109/JIOT.2016.2516102.

[11] S. Khanji, F. Iqbal, and P. Hung, "Zigbee security vulnerabilities: Exploration and evaluating," in *2019 10th International Conference on Information and Communication Systems (ICICS)*, 2019, pp. 52–57. DOI: 10.1109/IACS.2019.8809115.

[12] Z. Chi, Z. Huang, Y. Yao, T. Xie, H. Sun, and T. Zhu, "Emf: Embedding multiple flows of information in existing traffic for concurrent communication among heterogeneous iot devices," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9. DOI: `10.1109/INFOCOM.2017.8057109`.

[13] I.S.C.C.39, "IEEE Guid for EMF Exposure Assessment of Internet of Things (IoT) Technologies and Devices," in *IEEE Standards Association*. 2020.

[14] F. Farha, H. Ning, S. Yang, W. Zhang, and K.-K. R. Choo, "Timestamp scheme to mitigate replay attacks in secure zigbee networks," in *IEEE Transactions on Mobile Computing*, vol. 21, 2022, pp. 342–351.

[15] M. S. Wara and Q. Yu, "New replay attacks on zigbee devices for internet-of-things (iot) applications," in *2020 IEEE International Conference on Embedded Software and Systems (ICESS)*, 2020, pp. 1–6. DOI: `10.1109/ICESS49830.2020.9301593`.

[16] G. Lee, J. Lim, D. Kim, S. Yang, and M. Yoon, "An approach to mitigating sybil attack in wireless networks using zigbee," in *ICACT*, 2008, pp. 1005–1009.

[17] N. Ahmed, H. Rahman, and M. Hussain, "A comparison of 802.11ah and 802.15.4 for iot," in *ICT Express 2*, 2016, pp. 100–102.

[18] D Creswell and W Creswell, "Research design: Qualitative, quantitative, and mixed methods approaches, 5th edition," in 2018.

[19] A Kazdin, "Design science methodology for information systems and software engineering," in 2014.

[20] A Kazdin, "Single-case research designs: Methods for clinical and applied settings, 2nd edition," in 2011.

[21] A Krasny-Pacini and J Evans, "Single-case experimental designs to assess intervention effectiveness in rehabilitation: A practical guide," in *Annals of Physical and Rehabilitation Medicine 61*, 2017, pp. 164–179.

[22] I. Vaccari and et al, "Remotely exploiting at command attacks on zigbee networks," in *Security and Communication Networks*, 2017, pp. 1–9. DOI: `10.1109/ICESS49830.2020.9301593`.

[23] scikit-learn developers. "User guide: Scikit-learn 1.4.1 documentation." (2011), [Online]. Available: `https://scikit-learn.org/stable/user_guide.html`. (accessed: 02.18.2024).

[24] T. Alhmiedat, G. Samara, and A. A. Salem, “An indoor fingerprinting localization approach for zigbee wireless sensor networks,” in *European Journal of Scientific Research*, vol. 105, 2013, pp. 190–202.

[25] G. Talakala and J. Bapat, “Detecting spoofing attacks in zigbee using device fingerprinting,” in *2021 IEEE 18th Annual Consumer Communications and Networking Conference*, 2021.

Appendix A

PCAP Parsing

The transmitted data over the airwaves was captured using the devices highlighted in section 4 and stored in standard pcap format. These data were then parsed to extract the ZigBee transmission data found at the first three layers of the ZigBee standard, as referenced in Figure A.1.

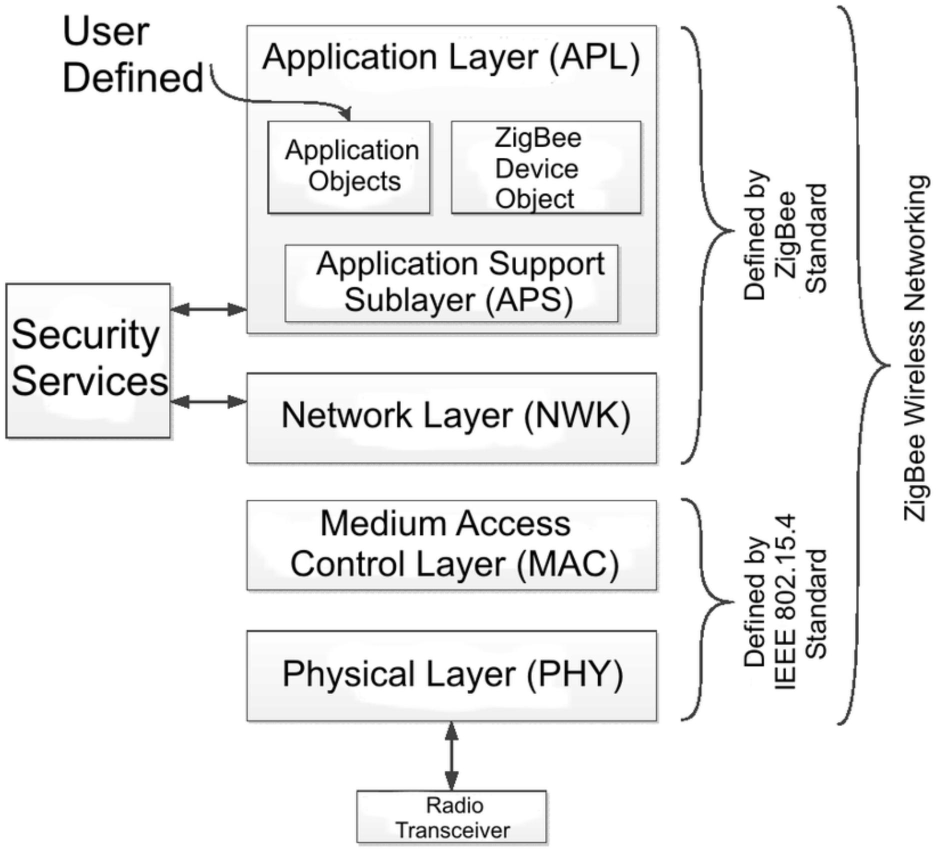

User Defined
Application Layer (APL)
Application Objects
ZigBee Device Object
Application Support Sublayer (APS)
Security Services
Network Layer (NWK)
Medium Access Control Layer (MAC)
Physical Layer (PHY)
Radio Transceiver
Defined by ZigBee Standard
Defined by IEEE 802.15.4 Standard
ZigBee Wireless Networking

Figure A.1: ZigBee Layers [2]

More layer information is available if the appropriate keys are known at the network and application layer. For these experiments, it is assumed that those keys are unknown and there exists a remaining chunk of data that is encrypted as a payload in the network layer. To aid in the pcap processing to ingest data into the machine learning model, the killerbee framework [7] was incorporated into a python class to parse each layer and extract out data.

A.1 Physical Layer

The Physical Layer contains the lowest level of information that allows the device to communicate with the hardware. In the killerbee framework, this is encapsulated as the "frame." Looking at the data that is available, data is extracted from each frame as outlined in Table A.1.

Table A.1: First Layer

number	integer	the frame counter
encapsulation type	integer	the encapsulation used
capture length	integer	amount of data captured
frame length	integer	the length of the frame
capture time	double	time in UTC format
time delta	double	time since the previous frame arrived
protocols	string	the category of message type

Since the protocol demands that the ZigBee encapsulation type is always 104, it could be removed from the dataset. However, when identifying abnormal behavior the encapsulation flag can be used to identify those. In the model training for **R1** this field is removed, but is left in for **R2**. For this layer, we are ignoring the payload field as it is represented in other data below.

A.2 Media Access Control Layer

The next layer up contains the protocol data unit that resembles the MAC layer of Ethernet. This layer is used solely to alert the receiver of who is supposed to receive this message and from where it came. In killerbee, it is encapsulated as the "wpan" layer. data was extracted from this layer as outlined in Table A.2:

Table A.2: Second Layer Initial

source	hex	unique id of who sent the message
dest	hex	unique id of who should receive the message
dest pan id	hex	unique identifier of the network it is associated
dest pan int	integer	the dest pan id as an integer
sequence number	integer	the sequence number of this message
frame control field	hex	frame descriptions
frame control field int	integer	frame descriptions as an integer
frame payload	hex	the payload of this frame based on the FCF

There are a few packed fields that need to be extracted so that the data is represented correctly instead of being packed into those hex bytes. By extracting that data and categorizing it based on the frame control field, it expounds into Table A.3.

Table A.3: Second Layer Unpacked

source	hex	unique id of who sent the message
dest	hex	unique id of who should receive the message
dest pan int	integer	the dest pan id as an integer
sequence number	integer	the sequence number of this message
frame type	integer	options: beacon, data, MAC, ACK, Unknown
frame security	bool	On or off for security
frame intrapan	bool	On or off for transporting or not
reserved	integer	reserved field that should always be 0
MAC CMD ID	char	command frames from 0-15
MAC Payload	hex	MAC command
MAC Payload length	int	length of MAC data
ACK	bool	true or false if it is a ack frame or not
BEACON address	hex	address of recipient
BEACON payload	hex	beacon data being processed
BEACON payload length	int	length of beacon data

A.3 Network Layer

The final layer that is extracted is the network layer. This layer can contain quite a few layers if the network data is decrypted with a network key. The presumption of this research is that the key is unknown which limits the data available to the fields listed in Table A.4.

Table A.4: Third Layer

source	int	unique id of who sent the message
dest	int	unique id of who should receive the message or broadcast
frame type	int	identifier on the type of packet
discovery	int	identifier on the discovery bits as a whole number
extended src	bool	if long or short addressing is being used for src
extended dst	bool	if long or short addressing is being used for dest
security	bool	if security is on or off
radius	bool	whether radius server is being used
protocol version	int	version being transmitted
sequence number	int	positive value of sequence for reliability
multicast	bool	if this is a multicast call or not
extended src64	int	the long MAC of the src as an integer
extended dst64	int	the long MAC of the dst as an integer
data length	int	length of the encrypted data portion
data	hex	data being transported
sec frame	int	the security counter of this packet

A.4 Computed Fields

In addition to the fields presented in the above tables, there are a few additional fields that are calculated during the data extraction to aid in processing. Due to the ease of execution, these were done prior to the data analysis for machine learning input. The first field is the entropy of the data being passed. The second field is the manufacturer of the device based on the MAC address of the device. A running table of MAC to manufacturer lookups are available on GitHub. The final field are listed in Table A.5.

Table A.5: Computed Data

entropy	double	entropy of the data field
manufacturer	string	the manufacturer of this device
device Identifier	string	the actual unique device name
delta_packet	double	time since last packet

A function was written to ingest that lookup table and tag each packet based on the extended src address within the packet to be used to train the model for **R1**. This field was validated through testing to ensure it was generating correct values. The third field is the unique identifier for the device used for training the model for **R2**. The script inputs the same value as manufacturer and requires the researcher to process some *sed* commands afterwards to modify this last field based on the actual devices being used within the experiment as listed in Chapter 4.