Securing Internet-of-Things Devices

By

Leigh Fix

December, 2021

Director of Thesis: Dr. Karla Varnell

Major Department: Department of Technology Systems

ABSTRACT

Smart home devices, also known as the Internet of Things (IoT) devices, are utilized more and more each day. As these devices grow in popularity, users connect to personal and private networks with devices that were unheard of ten years ago. The problem examined in this study is the security posture of IoT devices. Attackers are finding it relatively easy to access data on personal IoT devices. As the researcher, I examined the vulnerability of various types of IoT devices. IoT has allowed the public to take devices with them, creating a larger footprint, opening multiple attack vectors to exploit the data we produce daily. Ideally, these devices should be secure out of the box, so that users can trust the devices they have connected. Smart home technologies allow both autonomous and managed connections to a variety of network-connected devices. Using the penetration-testing framework known as the Information Systems Security Assessment Framework, the vulnerabilities present on these devices were examined. Kali Linux provided the best platform when trying to breach the IoT devices. Utilizing Kali Linux, I was able to breach more devices than using ParrotOS or Commando VM. Of the different types of IoT devices examined in this study, Kasa was the most susceptible to a breach.

I was able to determine the IP address and hostnames of all 15 devices. On 47% (7 of 15) of the IoT devices, I was able to obtain the location of the rooms these devices were in. On 80% (12 of 15) of the IoT devices, I was able to render them useless with a DoS attack. This study will contribute to the overall body of knowledge specific to the security and vulnerability of IoT devices and provide information for users who are likely to utilize them.

Keywords: Internet-of-Things, IoT, attack vectors, cybersecurity, smart home, attackers, exploit

Securing Internet-of-Things Devices

A Thesis

Presented To the Faculty of the Department of Technology Systems

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Network Technology

by

Leigh Fix

December, 2021

Securing Internet-of-Things Devices
By

Leigh Fix

APPROVED BY:

Director of Thesis _____
Karla Thompson Varnell, PhD

Committee Member _____
Peng Li, PhD

Committee Member _____
Philip Lunsford, PhD

Committee Member _____
Name and Degree of Committee Member

Committee Member _____
Name and Degree of Committee Member

Chair of the Department of Technology Systems _____
Tijjani Mohammed, PhD

Dean of the Graduate School _____
Paul J. Gemperline, PhD

Table of Contents

List of Tables

List of Figures

Chapter I: Introduction

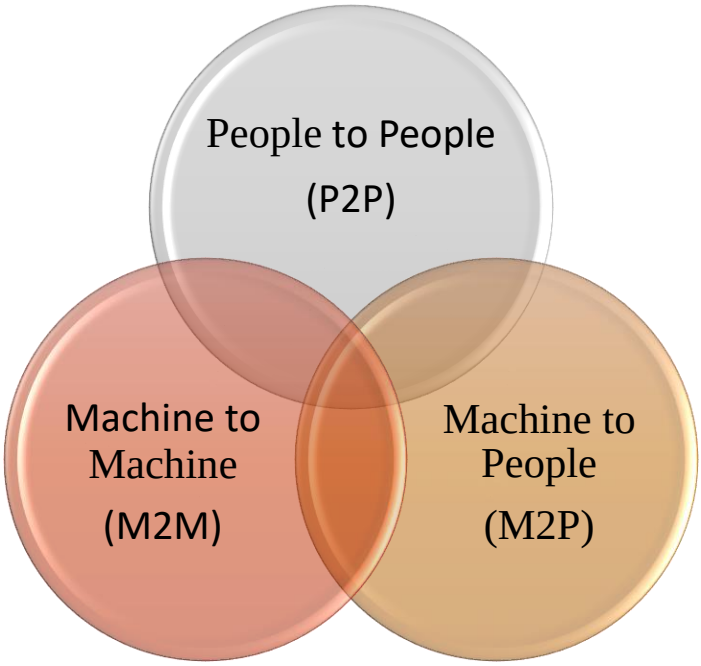
IoT devices have gained popularity over the past few years and are growing in numbers. According to HIS Markit, an information, analytic and solution firm, the number of IoT devices will swell to 125 billion by year 2030 (News release, 2017). The number of IoT devices with inherent security flaws continues to grow, which drives the importance of securing these devices properly. I performed this study to provide more data on the security of these devices. This study will benefit the information security community, providing a greater knowledge on the base level of security in the devices of this assessment. I will walk through what these devices are, how they are connected and the importance of this study. First, I will discuss the makeup of an IoT device.

Essentially, an IoT device is a sensor, embedded in the device that transmits data from one object to another without human interaction (Minerva et al., 2015). The end-user will turn the device or devices on and configure them to connect via their sensor to the end-user's network. Once connected, the end-user can then set certain parameters for the IoT device to follow for it to function. These devices, after configuration, begin generating data. Most IoT devices are always on, and most of their interactions take place non-transparent to the end-users. This allows for constant, real-time monitoring of multiple areas of our lives. These devices have the ability to collect information, transmit data to a hub, and put this information into a readable, usable form for the end-user. For example, users can setup sensors in a room that monitors the temperature. The Heating, Ventilation, and Air Conditioning (HVAC) system can adapt in real-time to adjust a room that is not within the range set at the thermostat. This type of communication falls into one of three forms of communication within IoT devices, Machine-to-Machine (M2M)

communication. This allows the devices to sense the surrounding temperature and communicate to the hub. Once this is done, the hub gathers the information and decides if it needs to adjust to keep the temperature stable. The hub will adjust until the sensor falls into the specific area of acceptance. In addition to M2M communication, there are two other connections available for IoT devices. Figure 1 shows three different connection types in relation to IoT.

Figure 1

IoT connection types



Note. The three different connection types for IoT devices.

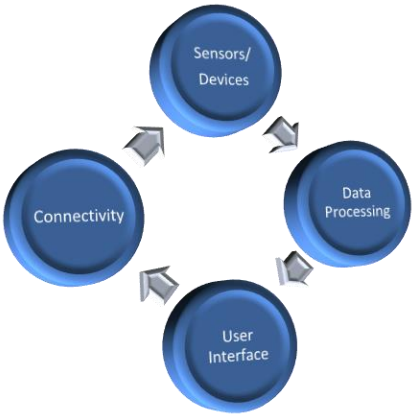
People-to-People connections are everyday collaborations in the workplace (Farhan et al., 2018). This is ultimately a social connection; we take the knowledge we have and share it with other people. Machine-to-People (M2P) connections are another form of connections in which the machines relay information in a readable format for humans to analyze and interpret. These connections are in use in our everyday lives. Vulnerability tools can obtain some information

about what IoT systems are vulnerable. The analyst or administrator will take the information and decide if they need to remediate or accept that risk. Finally, Machine-to-Machine (M2M) connections are Sensor-to-Sensor or System-to-System. This information is passed without any human interaction needed (Farhan et al., 2018). IoT takes elements from all three connection types and allows the information collected through multiple ways to be tied together to form a bigger picture. The goal of this technology is to collect information and turn that information into an asset capable of changing the way we respond to a plethora of scenarios. From traffic control to agriculture, this technology could help us maximize efficiency in multiple areas of our lives.

The Internet of Things combines the digital world of computers to the physical world we live in, mapping the real world to the virtual world (Buckley, 2006). This allows the systems to constantly mine data about the location, trends, log files, certain product usage, video surveillance feeds and many other types of data. It can use the data it mines to forecast trends or habits in our everyday lives to assist in the specific ways each device is designed for. As seen in Figure 2, sensors, data processing, connectivity and user interface is needed for IoT devices to function.

Figure 2

IoT components



Note. This is the components involved in an IoT system.

Sensors collect data from the environments they are placed in. These sensors can be paired with multiple devices to collect a deeper understanding or operate standalone (How IoT works, 2018). After the data is collected, it then needs a transportation medium for connectivity; this allows the sensor to send the data it has collected for the next step, which is Data Processing (How IoT works, 2018). Data Processing allows the data to be aggregated in one location and then put into a more readable format for the User Interface (How IoT works, 2018). The User Interface is where the end-user now has access to the data in a format that is easy to read and respond to (How IoT works, 2018).

With all the data IoT devices have access to, it becomes imperative that steps are taken to keep these devices secure. Every addition of a new smart home device creates another attack vector that a threat actor could exploit. With more devices being utilized, there will be a higher number of devices that are either misconfigured or configured using the baseline configuration, making an easy target for an attacker (Unit 42, 2020). Knowing there will only be more IoT

devices added to the global network, there is a need to understand the baseline configurations of these devices. If we can discover and mitigate possible issues that come "out-of-box" it could help to ensure these devices are stable for years to come. End-users are given some freedoms when it comes to how these devices are configured, but they all stem from the same base configuration. When users have the ability to configure devices, sometimes they will be incorrectly configured.

IoT devices attempt to make our lives easier. They allow us to track certain aspects of our lives that we previously have not been able to gather data on. These devices allow us to track our fitness, shopping habits, access to our homes, what happens inside our homes while we are gone, tracking our pet's behaviors, what temperature our house is, television habits, etc. From a cyber-security perspective, all these devices are an entry point to a network that could have useful information. There is a need for more information on the security of these devices. Many IoT devices communicate transparently to the end-users. This transparency could create an invisible threat that goes undetected until a larger breach is discovered. These are the reasons I examined this topic and share what devices can be exploited easily.

Background of the Problem

IoT devices are known for having a lack of security, which may result in a network compromise (Lack of security in internet of things devices, 2014). I performed a penetration test against multiple IoT targets, using three open-source software solutions to assess the security flaws that are present in out-of-box IoT devices. This assessment was carried out following the Information Systems Security Assessment Framework (ISSAF) in order to collect, organize and present the findings to assist in determining the vulnerabilities present. Previous studies seemed

to focus on a singular IoT device. There are many different examples that can be shared. For example, two people took control of a Jeep Cherokee remotely (Miller, 2019). Another study documented how to use a robot vacuum as an entry point into someone's house, using the camera. The majority of these single studies show that these devices have a large number of vulnerabilities on them (Sami et al., 2020). Many of these devices allow for an exfiltration of data. Ninety-Eight percent IoT devices have unencrypted transmissions. This means that the data can be easily pulled (Unit 42, 2020). The problem with IoT security derives from the way we are told to operate these devices. Many of these devices come with a standard out-of-box configuration in which the user will plug the device in and connect to their network. This is inherently flawed because the average user has no knowledge that these devices lack security or how to enable security. If users had more knowledge about what devices might be susceptible to, as well as the steps to help prevent exploits possible on these devices, the security posture may benefit greatly. Most of the literature available has a very narrow scope showing device security on a case-by-case basis, taking the known exploitable devices, and proceeding to discuss the issues after the exploit is released. I performed this study to openly examine what information can be exfiltrated from these devices. The assessment shows a different perspective from addressing issues known, to uncovering new issues and presenting the corrective actions. This study evaluates the security vulnerabilities of IoT home devices. This study presents the necessary steps to easily correct any of the issues uncovered through the security assessment. Running a security assessment on various devices will give the IT community an updated base of knowledge in a variety of IoT areas. IoT devices are subject to a myriad of issues related to their cybersecurity configurations. This study will bridge the gap between the necessary security settings of these devices and the security settings they have "out-of-box."

Statement of the Problem

IoT devices inherently are not secure with an out-of-box setup. As a result, users are not aware of this vulnerability. IoT devices accounted for around thirty-three percent of breached devices in 2020 (Attacks on IoT devices continue to escalate, 2020). With the amount of IoT devices currently in use, and with growth expected to be 125 billion devices by the year 2030; security is a glaring problem (News release, 2017). With billions of devices connected, we need to know if we can adequately trust the factory configuration of these devices. This study will benefit the security population, giving an "in the wild" simulation of possible IoT vulnerabilities and threats. This will fill in the gaps from the single device assessments and show the possibilities of multiple IoT genres and how they could be vulnerable. Publishing information found from assessments will always raise the possibility of someone using this information detrimentally. However, to begin understanding what data can be exfiltrated from these devices with an "out-of-box" configuration will prove beneficial to correcting security flaws present in these devices.

Purpose of the Study

The purpose of this study is to evaluate the inherent security flaws of IoT devices utilizing easily accessible open-source software. This study examined the security vulnerabilities inherent in IoT devices. By using open-source software and the ISSAF, I examined how these devices are vulnerable and how to correct the flaws, if there is an easy solution. This study leveraged an already established framework for a security assessment. I documented my steps from reconnaissance, information gathering, discovery, assessment, exploitation and maintaining access. The assessment is designed to collect the data from each step before proceeding to the

next step. Documenting where vulnerabilities are found opens the door to evaluating where and when corrective measures can be deployed.

Research Questions

This study is based on three questions that guided the research. Question 1) What types of information can be compromised from an IoT device? The majority of IoT owners probably have no idea as to the type of information collected and/or processed via their IoT devices. Question 2) Will certain types of devices be more at risk than others? Testing 15 devices in this assessment will examine which devices tend to be more susceptible to an attack. Question 3) Which open-source penetration toolkit is the most effective at exploiting IoT devices? I utilized three open-source toolkits to examine the vulnerabilities in 15 IoT devices. This research provides data beneficial to both the consumer and the manufacturer regarding vulnerabilities the devices tested have.

Attack Methodology Overview

To perform this assessment, I used the Information System Security Assessment Framework. This framework is an open-source framework designed to obtain information through a specific set of steps utilizing the penetration test process (Rathore et al., 2006). With this framework, I segmented the assessment into multiple steps, documenting all the information gathered while at each specific phase in the test. This framework breaks the assessment into information gathering, network mapping, vulnerability identification, penetration, gaining access and privilege escalation, enumerating further, compromise sites, maintaining access and covering tracks. All these steps provide the user the ability to leverage what is discovered at previous steps

of the process. I did not cover tracks. This step will not provide any necessary information important to this study. I documented what happens at each step in the framework. This information can then be evaluated to determine the various threats present due to vulnerabilities discovered for IoT devices involved in this assessment.

Delimitations and Limitations

This study will be limited by the number of devices evaluated. I only performed this assessment on devices that I own. This presents a bias on my own network security. I addressed this by configuring a guest network with a randomly generated security password. I also reset all my IoT devices to their factory defaults, removing all additional security measures I have taken to secure them. Another limitation of this study is performing only one methodology. When an IoT device is configured in the wild, there is a variety of unknowns; for example: will the device be maintaining constant updates, will the owners change the default passwords, and can an attacker gain physical possession of the device? I performed a targeted, organized assessment to generate useable data. However, that means this study was performed within scope limitations. An attack in the wild will not have these limitations; the only limits would be those that the attackers would place on themselves.

Assumptions

The assumption that most IoT devices are vulnerable is validated in many IoT studies. For example, around 57% of IoT devices are vulnerable to attack, with an astounding 98% of IoT traffic being unencrypted (Unit 42, 2020). Another assumption is that an attacker would be able to construct an authored exploit to compromise these devices. This assumption means that some

of what may not be capable with open-source tools is capable by a seasoned attacker with the knowledge to construct their own exploits. Without taking the time to code my own specific exploits, I will not be able to accomplish this for the scope of this specific study; but the assumption is that this is within the realm of possibility.

Definition of Terms

Architecture: the design and structure of a computer system, which controls what equipment can be connected to it and what software can operate on it (Cambridge dictionary, n.d.).

Attacker: A party, including an insider, who acts with malicious intent to compromise a system (Grassi et al., 2017).

CVE: Common Vulnerabilities and Exposures, a dictionary of common names for publicly known information system vulnerabilities (Editor, n.d.).

Penetration Test: A test methodology in which assessors, typically working under specific constraints, attempt to circumvent or defeat the security features of an information system.

Pwned: To defeat or take control of a system (Cambridge dictionary, n.d.).

Protocol: a computer language allowing computers that are connected to each other to communicate (Cambridge dictionary, n.d.).

Router: a piece of electronic equipment that connects computer networks to each other and sends information between networks (Cambridge dictionary, n.d.).

Switch: is a high-speed device that receives incoming data packets and redirects them to their destination on a local area network (Techopedia, 2011).

Threat: Any circumstance or event with the potential to adversely impact organizational operations (including mission, functions, image, or reputation), organizational assets, individuals, other organizations, or the Nation through an information system via unauthorized access, destruction, disclosure, modification of information, and/or denial of service (Grassi et al., 2017).

VLAN: is a logical group of workstations, servers and network devices that appear to be on the same local area network despite their geographical distribution (Techopedia, 2011).

Vulnerability: Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited or triggered by a threat source (Dempsey et al., 2011).

WiFi: a system for connecting electronic equipment such as computers and electronic organizers to the internet without using wires (Cambridge Dictionary, n.d.).

Wireless: using a system of radio signals rather than wires to connect computers, cell phones, etc. to each other (Cambridge Dictionary, n.d.).

Significance of the Study

This study fills the research gap that exists regarding the level of vulnerability that comes with an out-of-box IoT device. This study highlights the level of trust these devices embody without applying any type of security. Typically, users do not apply configurations to IoT devices before deploying them into a network. This study shows the vulnerabilities and the information someone can exfiltrate from IoT devices with a factory setting. After showing the security flaws on these devices and the types of information that will be available from these devices, this study presents the corrective actions necessary when available. The information that

is presented from this study will assist in the proper mitigations for a standard user to ensure their devices are properly secured.

Chapter I Summary

Chapter I included information on what an IoT device is. In addition, information was shared on to how these devices function. The overall number of IOTs in use today that may lack basic security and be vulnerable to an attack is growing exponentially. Using a standardized penetration-testing framework, ISSAF, I obtained information on the vulnerabilities of IoT devices. In Chapter 2, I will evaluate the literature that already exists on IoT device security. I will walk through some of the different methodologies known to the security world today. Then I will present some of the open-source tools that perform security assessments. I will also review on the protocols that IoT devices leverage and evaluate how they work.

IoT device security is something that many people take for granted. In today's security landscape, many people ignore securing IoT devices in lieu of securing other devices on their networks. To help understand the overall security posture of these devices we need to understand more about them.

Chapter II: Literature Review

With the attacks of IoT devices increasing, there is a need for more information on how to secure them. The current literature provides the functional level of security configured on these devices. The literature also documents the working architectures these devices have from the manufacturer. Examining this information with the ports and protocols that allow for the communication of these devices, a threat actor can start to build an attack. In this study, I have examined 46 total sources of information; 24 journals, 7 web sites, 7 publications/documentations, 3 whitepapers, 3 dictionary/glossary, and 2 reports that document the correlation between functional capability and device security.

In this chapter, I will highlight findings from previous studies on IoT devices. First, I will discuss the architectures that are present on many IoT devices; this gives the base understanding for how they operate. After touching on IoT architectures, I will discuss the communication types amongst IoT devices. Next, I will provide a brief overview of the protocols running on IoT devices via the communication channels. Then I will introduce the penetration toolkits used for this study. Finally, I will touch on the gaps filled by this study and a summary before moving on to the next chapter.

The current literature on IoT devices provides a broad overarching theme to apply security in a manner that we are already used to. The problem is that many IoT devices are not secure with factory configurations. These studies show the necessary information needed to begin understanding how IoT devices work (O'Neill, 2016). An attacker can take the studies and gain the knowledge for performing a successful attack. An attacker can leverage these studies to exploit IoT devices, so it is necessary to understand the underlying environments these devices

will be in. To begin understanding more about IoT devices, let us discuss the architectures these devices may operate with.

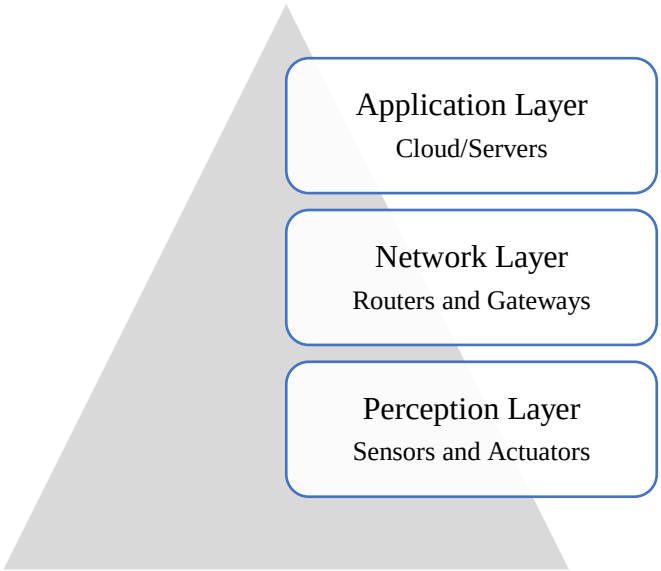
Architectures

IoT devices can operate within several types of architectures. With the systemic approach, a fourth node is added to a classical three-node environment; this approach adds an Intelligent Object node, which will lie in the center of the triangle nodes of Person, Process and Technological System (Riahi et al., 2013). The first node, Person, plays a fundamental role in the security framework of the IoT ecosystem; they would be responsible for setting the security of the devices via rules, roles, practices and ensuring the operational capability from any security adjustments. The second node, Process, is described as being tasks accomplished from the IoT environment. This node has the security compliance and policies coded into the system to ensure the ecosystem is safe; this node creates the tradeoff required to keep the system secure while also maintaining a level of complexity that is useable. The third node, Technological Ecosystem, is the environmental changes made to ensure the IoT devices are secure. This node requires security requirements, but will tradeoff with the ability of the devices and the evolution of technology to ensure the system is secure while not degrading the ecosystem. The study provided by Riahi et al., looked at the addition of the fourth, new node. The Intelligent Object would communicate with each of the previous nodes through a direct connection to each. This allowed the environment to cooperate, share and exchange environmental information, which is fundamental for security due to the pervasive connectivity of IoT devices (Riahi et al., 2013). Adding this Intelligent Object blends the artificial intelligence into the conventional IoT systems allowing for a greater level of security to be provided (Riahi et al., 2013).

Another study looked more into the current state of IoT devices. This study relies on the layout of the IoT architecture. The architecture of these devices all falls into the same category. They have three layers that they are dependent on to function. Figure 3 (Calihman, 2020) shows the three-layered architecture.

Figure 3

Three-layer IoT architecture.



Note. This is the setup for the three layers of an IoT architecture

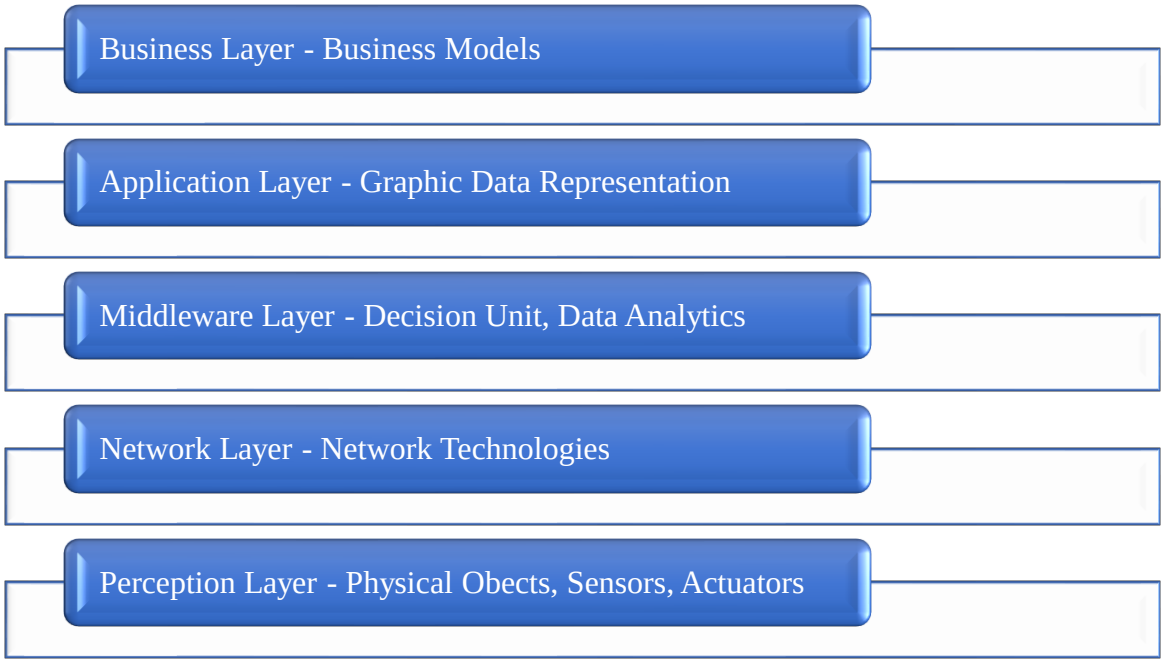
The first layer is the perception layer, this layer is where the sensors are; the data is collected and processed at this layer then handed off to the next layer to continue the chain (Mahmoud et al., 2015). The study documented this layer having three primary security issues: signal strength of the wireless devices, man-in-the-middle attacks on the IoT nodes and the security risks inherent to network topology. Signal strength becomes an issue because these devices are becoming smaller and smaller, so they are easier to place around different environments. Some of the areas in which these devices will be placed could open connection

issues, a wireless connection can only go so far, and will degrade the further out the device is. As distance between sensor and hub increases the opportunity for is a man-in-the-middle attack also increase. If these devices increase their Wi-Fi capability, the distance between the sensors and hubs will also increase allowing an attacker to hijack the connection without being noticed. The network topology presents its own specific set of weaknesses. The second layer, the network layer, is where the data will be routed from the hubs and internet connected devices. This layer is susceptible to all the standard network level attacks such as w Denial-of-Service (DoS), Distributed-Denial-of-Service (DDoS), man-in-the-middle, or simple network sniffing. These vulnerabilities are well known to the hacking community and make these devices susceptible to attack. The third layer, the application layer, is the layer that the end-user will see; this layer is responsible for the authenticity, integrity, and confidentiality of the data. This layer is also susceptible to security issues. With no global policies for IoT devices, this allows multiple authentication mechanisms for differing technologies (Mahmoud et al., 2015). A user could connect three different technologies requiring one to use three different applications to control each individual technology. If the user does not particularly like bloating phones or tablets with applications, this could become an issue.

Another approach breaks the IoT architecture into five layers: Data gathering, data transmission, process information, smart applications, and system management. Figure 4 shows the five-layer approach.

Figure 4

Five-layer IoT architecture



Note. This figure shows the layout of a five-layer architecture system.

The perception layer, or physical layer, is the layer in which the actual piece of hardware is sitting (Kumar & Smys, 2018). In this layer, the identification and collection of the data from our physical sensors occurs. For example, this is where the data for a smart thermostat gathers its temperatures or where smart leak sensors detect water. The perception layer has the ability to transmit the data it has gathered from the physical sensors to be processed. This all happens via the network layer. The network layer is contingent upon the device used. This transmission could take place via cell service (3G/4G/5G), RFTD, WiFi, WiMAX, satellite, etc. This layer ensures the secure transmission from the perception layer to the next layer, the middleware layer. The middleware layer is the layer that will process the information gathered. This layer has two primary functions, managing the services provided and storing the information provided. This is

the layer that interprets the data as it is being sent and responds accordingly if it does not require human interaction. After the middleware layer comes the application layer. This is the layer that is responsible for building a human readable report from the data provided via the middleware layer. For example, this layer involves the application installed on a cell phone or tablet. It provides the information that has been gathered and transmitted from previous layers into a form easier for people to read and/or interpret. The final business layer is responsible for building out models, graphs, executive reports, etc. This layer is where functional leaders will parse the data gathered to try to make informed decisions and strategies to better suit the business needs (Kumar & Smys, 2018). This architecture aims at preventing security and privacy issues relating to the size and scalability possible in IoT devices (Mehta et al., 2018).

Another study examines the connections of IoT devices and breaks the communication down into three main categories. The first form of communication is Device-to-Device. This can be with or without human interaction and allows the IoT devices to communicate with each other. The second form is Device-to-Human. Humans communicate directly with the device without going through another device. The third is device-to-distributed storage. This allows devices to connect directly to a storage form, whether that be cloud based or network attached (Bello & Zeadally, 2016). These types of communication allow the IoT devices to run without much interaction from the users, to perform the tasks they are developed to perform. Because this communication can happen in different scenarios, these devices can communicate in either a single hop network or multiple hop network; allowing the devices to traverse from a simple environment to a more complex environment to carry out their tasks. This study focused primarily on the Device-to-Device communication. There are multiple approaches to accomplish communication through Wi-Fi, to which most people already have access. ZigBee, which is a

short range, low power networking standard. Bluetooth, which is another standard most people are familiar with, using radio frequencies to communicate, and even cellular data. With a Device-to-Device connection, it will allow IoT devices to ensure they are operating even when other networking equipment might not operate. If a switch or router is not provided with a higher amount of power, they simply will not operate. If one compares that to the Zigbee networks, they operate under a low power, allowing for a smaller electronic footprint and a lower reliance on higher standard electricity. This also allows these devices to be pushed further and further apart, with lower power reliance they will also relay from Device-to-Device until they can put their data onto the hub or device they call home. With them being able to relay their data, that also requires the IoT devices to have some awareness of the network, requiring them to be semi-intelligent, which increases functionality when they need to off-load information to increase capacity, if they lose communication with another device temporarily, when the core network is down, they trigger alarms based on unusual scenarios (Bello & Zeadally, 2016).

There are several types of architectures that IoT devices can utilize. As shown, some are more secure than others are.

Penetration Toolkits

There are numerous tools available to a threat actor that can be used to compromise a network. These tools do not always cater to a user that does not have much experience within the cybersecurity realm.

One of these tools is Kali Linux. This is an operating system designed to assist in cybersecurity related tasks for professionals. Kali Linux is an advanced penetration-testing operating system that includes over 600 tools categorized into the following: Information

Gathering, Vulnerability Identification, Penetration Testing, Wireless Attacks, Web Application, Digital Forensics, Sniffing and Spoofing, Password Attacks and Reverse Engineering (Kali docs: Kali Linux documentation, 2021). The developers of Kali Linux have created the build to be highly customizable, however they advise that one will not be able to use repositories that are "out-of-band" and they will not have inherent compatibility. They also advise a user must have Linux familiarity or this tool will be difficult to work with. The installation requirements for Kali Linux requires 2048 MB of RAM and 20GB of hard drive space (Kali docs: Kali Linux documentation, 2021). This toolkit was utilized in my study.

ParrotOS is another penetration toolkit. This tool is also a Linux-based distro like Kali. However, it separates itself from Kali Linux by being a Debian based distro. ParrotOS requires at least a dual core 64-bit processor, 2048 MB of RAM and 40GB of hard drive space (System requirements - parrot documentation, n.d.). This requires a little more than Kali Linux does. Many of the tools provided by Kali Linux are also on ParrotOS. However, the ParrotOS project has also built some of their own tools and provided them via this installation. In terms of usability, ParrotOS attempts to reach a broader level of user, making security more accessible to a larger number of less experienced professionals. This toolkit is used in my study.

Commando VM is relatively new to the penetration-testing world. This is a "first of its kind" distribution. Commando VM is a Windows based Operating System. Previously all penetration toolkits were a flavor of Linux (Barteaux, 2019). This toolkit provides a more stable variation for Windows based environments. This build is pushing to become the de-facto Windows scanning toolkit. This toolkit needs a bigger build for it to function than Kali Linux or ParrotOS; this build requires 60GB of hard drive space, but the same amount of RAM (2048MB)

as the previous toolkits (Barteaux, 2019). There is very little information about Commando VM, because it is relatively new. This toolkit is used in my study.

Another toolkit utilized in cybersecurity is that of the Penetration Testing Tool for Internet of Thing Devices (PENTOS). This tool was developed with the objective to provide an automated step-by-step instruction to assist novice users in penetration testing (Visoottiviseth et al., 2017). This tool has seven main features for testing IoT security, information gathering, password attack, Wi-Fi attack, Bluetooth analysis, web scanner, web attack and scanner as presented in Table 1, the detailed hardware and software specifications are shared and differ from the other three toolkits discussed:

Table 1

PentOS

Device	Hardware and Software Specifications
PentOS	CPU: 4 processor Cores Memory: 10.24 GB Hypervisor application Enable code profiling Bluetooth USB 2.0 TP-Link n150 tl-wn722n WiFi adapter VMWare fusion 8.0
IoT Device	1.2 GHz 64-bit quad-core IEEE 802.11n Wireless LAN Bluetooth 4.1, Bluetooth Low Energy (BLE) 1GB RAM 4 USB ports Raspbian OS

Note. This table shows the specifications for PentOS.

The developers created this tool to help expand user's knowledge of IoT devices, ensuring we can continue to securely develop and trust the devices we bring into our home. PENTOS is actively adding more functionality and turning open source to aid in development of this tool

(Visoottiviseth et al., 2017). I did not use this tool in the penetration test; however, I did find it necessary to highlight in this study.

Protocols

Understanding the underlying protocols that IoT devices use to operate is functional to determining how exploitable these devices are. IoT devices run on a large pool of differing technologies, so uncovering information about these protocols is a necessary step. Table 2 (Vidales, 2017) shows the protocols involved at the connection level of IoT devices.

Table 2

Connection protocols

Technology	Frequency	Data rate	Range	Power usage	Cost
2G/3G	Cellular Bands	10 Mbps	Several Miles	High	High
Bluetooth/BLE	subGhz, 2.4Ghz	1,2,3 Mbps	~300 feet	Low	Low
802.15.4	subGhz, 2.4Ghz	40, 250 kbps	> 100 square miles	Low	Low
Wi-Fi	subGhz, 2.4Ghz, 5Ghz	0.1-54 Mbps	< 300 feet	Medium	Low
ZigBee	2.4Ghz	250 kbps	~300 feet	Low	Medium
Z-Wave	subGhz	40 kbps	~100 feet	Low	Medium

Note. This table shows connection protocols for wireless standards.

Starting with Bluetooth/BLE (Bluetooth Low Energy), this protocol not only reduces the energy consumption of normal Bluetooth, but it also reduces the cost via energy consumption (Dragomir et al., 2016). BLE has a very low distance range, due to its low range this protocol is typically used for IoT devices that will be close to each other and the hub that connects them together. Introduced in 2003, revised in 2015, 802.15.4 is another protocol, which is geared towards low power and low-cost technologies; this protocol offers a much better range than BLE but is

primarily used in industrial settings. Wireless Fidelity (Wi-Fi) is the protocol which most people are familiar with. Wi-Fi is the technology that the public to connect to the internet. This protocol is a solid, secure choice for ensuring IoT connectivity. Wi-Fi operates on multiple bands and offers a stable, secure protocol enabling IoT devices to be connected to already established networks. This allows for an ease of control and management of distributed IoT devices. Z-Wave is another protocol based on low power usage, but at a little higher cost. Z-Wave is a protocol that covers multiple layers; it is not just a connection-based protocol. It covers the physical layer all the way to the application layer. This protocol has a mesh-based network; it has multiple nodes that connect to each other, all managed by a specific home. ZigBee is another total protocol stack. ZigBee builds on 802.15.4, making this protocol compatible with other devices from other manufacturers. ZigBee is based off keys; the keys are then shared between devices providing them connectivity (Dragomir et al., 2016). There are other layers of the protocol stack which house technologies that allow IoT devices to function properly. Table 3 shows more of the protocols used at various layers.

Table 3

Protocols

Application	Transport	Convergence	MAC	Physical
COAP, MQTT, LLAP, AMQP, SMCP, DDS	IPv6, IPv4, UIP, UDP, TCP, uIP, ROLL, RPL, DTLS	6LoWPAN	IEEE 802.15.4e, Zigbee, TSMP	IEEE 802.15.4

Note. This table shows the protocols used by the different layers.

6LoWPAN is a protocol geared towards bringing IoT compatibility for IPv6. This has mainly been used for home automations and smart meters. However, having the larger address space that comes with IPv6 provides the ability for a larger number of devices to connect (Sobin,

2020). UDP is the connectionless protocol the IP stack uses. This protocol provides IoT the ability to send a large amount of data without congesting the network up with many replies. Fire and forget is the UDP motto. uIP or MicroIP is a miniaturized version of the TCP/IP stack for embedded systems. This protocol provides the ability to minimize code and memory required to use. CoAP is a transfer protocol, primarily used for low energy capable devices. CoAP allows multicasting, which is very similar to broadcasting, only the information is transmitted to users. Through multicasting, this protocol has the benefit of reducing the strain of multiple IoT devices on a shared network with limited bandwidth. MQTT employs a client/server-based setup; the server is sent the data that the client has collected. This protocol is a lightweight, scalable protocol that can operate on a one-to-one, one-to-many or many-to-many setup all while preventing network congestion. This technology, however, only supports TCP, which in turn allows for packet loss to become an issue. DDS or Data-Distribution Service is a standard that supports real-time systems. DDS relies on a publisher/subscriber model. This allows the publisher to only supply the information a subscriber is requesting. This eliminates the need for complex methods (Sobin, 2020).

The sheer amount of IoT device manufacturers, as well as different functions for these IoT devices, make the entire list of used protocols difficult to understand. Most of these devices will run similar protocols, but depending on the need of the device, they could all leverage different solutions to determining how the individual device should connect and share information.

Overview of Challenges for IoT

The challenges that present themselves for IoT device security vary, as they are no longer a standard network type device. The CIA triad: Confidentiality, Integrity, and Availability, are key concerns when it comes to IoT devices (Jose & Vijyalakshmi, 2018). Other challenges for IoT devices are a lack of standards and metrics (Kumar & Mallick, 2018). There are also studies showing the lack of privacy protection models for IoT. This makes the expectation of privacy a large concern when it comes to device security (Kumar & Mallick, 2018). Another privacy concern is unauthorized reconnaissance from users. IP cameras or listening devices can be used to spy on end-users (Miloslavskaya & Tolstoy, 2018). Scalability is another glaring challenge for IoT devices; the sheer number of devices in circulation raises questions on managing these devices throughout their lifecycle (Sedrati & Mezrioui, 2018). A large number of devices introduces a larger pool of security vulnerabilities with no generic management to help assist in remediations for IoT devices (Sadique et al., 2018). Another challenge for IoT devices is the human element. The end-user will be responsible for updating these systems. If this does not happen, then that device is vulnerable (Sadique et al., 2018). Another challenge of these devices is the hardware being compromised. In 2018, the hardware of the Raspberry Pi system had critical privilege escalation vulnerabilities disclosed (Security Compass, 2019). With more and more autonomous vehicles on the road, the trend could also shift to spoofing of road signs (Security Compass, 2019). Threat reports are showing that hackers are actively attacking IoT devices trying to find a way into the networks they want access to (Security Compass, 2019). Ensuring these devices maintain a secure level of patching will become another hurdle for IoT devices, an astonishing 83% of medical imaging systems are running on end-of-life operating systems (Unit42, 2020). This is a glaring challenge for these devices, and it shows how quickly

the situation can progress to becoming a problem. Securing all these devices will be a complicated task, but some efforts lead toward enforcing unique device entities and ensuring access control to all of the IoT devices in the landscape (Heiser & Ryan, 2019). This will be a challenge on its own, with the low cost and ease of deployment, managing the number of devices on a network could become overwhelming. With all the challenges IoT devices are facing, additional information on the vulnerabilities of these devices may encourage manufacturers of these devices and the end-users who purchase them to be more mindful of security.

Gaps Filled

During this study, I attempted to penetrate 15 IoT devices. Most assessments performed on IoT systems are primarily on a single device. This will benefit the security community by showing how one can leverage the information to pivot internally to advance status. Many studies show IoT devices are insecure (Kumar & Mallick, 2018; Miloslavskaya & Tolstoy, 2018; Sadique et al., 2018; Sedrati & Mezrioui, 2018). Drawing data from a "smart home" will provide a stable environment to see exactly what can be done with these devices. My study provides information that documents the flaws in IoT devices and how to properly configure them to prevent any type of security incident.

Chapter II Summary

In this chapter, I provided an in-depth look at key elements of IoT devices. Different classifications of architectures for IoT devices are necessary for a threat actor when building an attack framework. In addition, I shared information about open-source software, which will be used to perform the assessment. Leveraging different security tools against each other will

provide data related to certain devices being more or less vulnerable, depending on the number of findings from each. Kali Linux is the most “well-known” of these security operating systems. While Kali provides the end-user with all the tools necessary, it does not contain the same tools as Parrot OS. Parrot OS is another security operating system. This system is built on the Linux kernel, but the difference is in the tools on the system themselves. Commando VM is the newest of the three Operating Systems I will utilize; this is a windows-based operating system. This will provide a different viewpoint because most of the attack simulation studies have been performed with a Linux-based operating system. Providing literature on the abilities from a Windows system will be completely new information. The results this study provided showcased the capabilities from a brand new, windows-based, operating system. Using this system could be a huge opportunity for more operating systems similar in nature to break into the security assessment realm. This chapter then broke down the protocols IoT devices are using. Knowing the protocols these devices are using is one of the key pieces of information to build a successful attack. The application stack is broken into segments. The segments that will be leveraged heavily upon will be the transport, MAC, and physical layer. The transport layer includes IPv4, UDP and other methods used to transmit data. The information gained from this layer provides the ability to pivot into other systems on the network. The MAC layer may have low power emanations a threat actor is looking sniff from the air. This layer will most likely be used to get on to the network. The physical layer is where the device is physically located. If IoT devices are put somewhere that an attacker has unsupervised access to, the device may be compromised. I then discussed the gaps that my research filled. In the next chapter, I discuss the methodology used in my study.

Chapter III: Methodology

The purpose of this study is to evaluate the inherent security flaws of IoT devices utilizing easily accessible open-source software. In Chapter III, I will cover the Research design followed throughout this study. I will also cover the role of the researcher for this study and then move to the methodology used in the assessment. I will then cover how the assessment will be set up, how the data will be collected, and why this data is credible and trusted.

Research Design

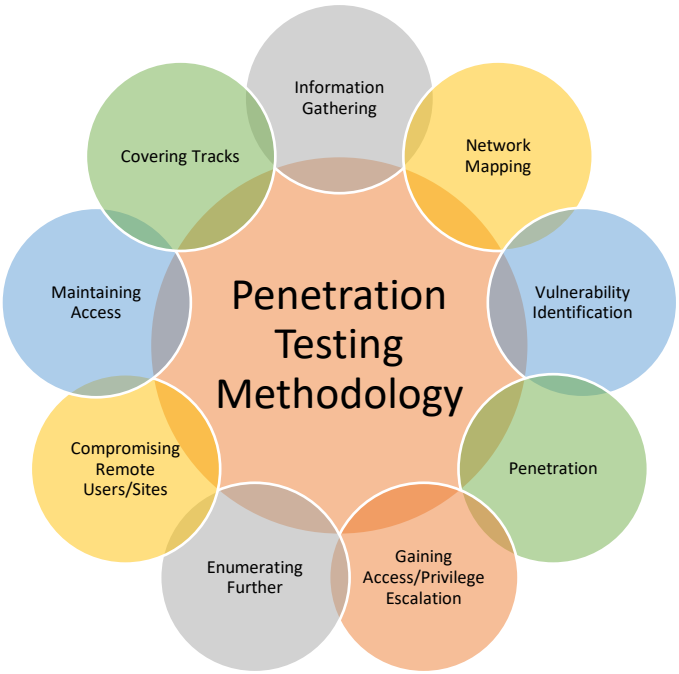
In this study, I analyzed the vulnerabilities of common IoT devices when utilizing three common open-source software programs for attacks. The three research questions for this study are: What types of information could be compromised from an IoT device? Will certain types of devices be more at risk than others? Finally, which out-of-box penetration toolkit is the most effective at exploiting IoT devices? These three questions are the driving force for this security assessment.

With this study, I followed the ISSAF penetration-testing framework. It began with the Planning and Preparation phase. In this phase, I set up the security tools (Kali Linux, ParrotOS, and Commando VM). Once I had my tools configured, I was ready to scan, I proceeded to the assessment phase. This phase is where I will began looking for any IoT devices in the area. The first step is gathering information about the system. After I gathered information, I built a map of the network. Taking device names and IP addresses, tying these devices to protocols that are present on the network gave me the ability to start determining how I may be able to exfiltrate data. After mapping the network, I began looking for vulnerabilities on the devices. I built another table with vulnerabilities tied to the device I was attempting to exploit. After the

vulnerabilities are mapped, I attempted to penetrate the devices. After a compromise is successful, I documented how I penetrated the device and maintained my access. This step is key; many devices have the capability of kicking unknown connections off. I leveraged the ability to keep access in case something like this happened. The final phase is the reporting phase. This is the phase I provided all the information gathered through this process, from the beginning of the assessment to the end. In this phase, I documented how one could prevent these attacks from becoming successful. Figure 5 shows the steps involved throughout this process.

Figure 5

Penetration testing methodology



Note. This is the penetration testing methodology flow.

There are multiple frameworks, which could be used to provide this type of data. One methodology I could use is the Open-Source Security Testing Methodology (OSSTM). This is a

testing methodology developed by the Institute for Security and Open Methodologies, which is used as open-source instructions to test the security of the targets (Herzog, 2010). The OSSTM has adapted from simple testing of desktops and laptops now to encompass the entire depth of a network, eg. Human factors, physical devices, wireless devices, telecommunications equipment, and data networks (Herzog, 2010). For my assessment, setting up controls would not add any beneficial information to the study. Some of the channels would also need to be completely disregarded. This framework will provide a good amount of data to generate a risk level for the devices on the network, but not directly attack the devices.

Another methodology I could use is the National Institute of Standards and Technology (NIST) Special Publication 800-115. NIST SP 800-115 breaks the testing procedure into three phases. The first phase is the planning phase, in this phase, one gathers all the information about the target one can, this could be defined as the assets to be tested, the rules of engagement, the threats interest of assets and the controls to mitigate the findings (Scarfone et al., 2008). The second phase, the execution phase, which is when one identifies and validates the vulnerabilities and the assessment method to be used. The third phase is the post-execution phase, this is the phase in which the vulnerabilities can be tied to their root cause, the mitigation recommendations and the reporting will be done in this phase. NIST has changed the white-hat to an overt and the black-hat to a covert test (Scarfone et al., 2008). For the benefit of this study, I felt it necessary to include this framework; NIST provides a solid foundation for risk management. This framework does provide a high level of documentation and information to ensure a system is compliant but is lacking in exploitation. The framework I chose to use is the ISSAF standard. The ISSAF has created one of the best frameworks for penetration testing; it provides a high level of detail and a

good roadmap to perform an assessment. These are the reasons why I have selected this framework to perform this test.

Researcher Role

My role in this study is that of a participant. I searched for vulnerabilities on devices to attempt actively compromising one or multiple devices on the network. I also took the role of configuring all the devices to a factory-default level. I personally believe the test needs to be performed on devices "fresh from the factory" to show the need for either stricter security sets from the vendors, or a wider knowledge for people to secure these devices after they receive them. I also documented the information gathered from these devices and from the network.

Researcher bias may exist when the researcher relies on their knowledge when breaching IoT devices based on experience. This bias may potentially result in the researcher overlooking other breach possibilities that are unknown to him. In these cases, utilizing three different toolkits available will aid in detecting what may be missed.

I conducted this assessment in a professional manner, following the process I have laid out to ensure a non-biased, professional security assessment. I examined devices from a diverse set of vendors, not focusing on one vendor or device type to avoid any type of manufacturer bias.

Research Methodology

This study utilized three penetration-testing toolkits (Kali Linux, ParrotOS and Commando VM) to actively exploit 15 IoT devices. This provided opportunities to discover and document security flaws that may be present for certain IoT devices. I manually examined the

results from each penetration test on each device, documented what did or did not happen, categorized it, and entered it into a table. The captured data was categorized according to the ISSAF. These data points were used to document each devices vulnerabilities and possible mitigation. To begin the setup of this environment, I created a guest network on my local router. Once I created the guest network, I assigned it a random passphrase, to create a "real-world scenario." After the guest network was setup, I then proceeded to connect all the IoT devices for this assessment to the newly created guest network. Once all the devices are connected to the network, I began the next step. I ran similar tests from each of the three penetration operating systems, repeating the process on multiple forms of IoT devices to ensure the integrity of these devices. This step is where the "attacking" was carried out. Each security assessment was performed by following the steps laid out in the ISSAF. I began with Kali Linux and performed the entire security assessment. I then formulated the success rate of attacks that have direct access to the devices and the success rate of attacks with no direct connection to the IoT devices. I then began again with Parrot OS. I performed another full security assessment and moved to formulating the attack success rate. Finally, utilizing Commando VM, I performed my third and final security assessment before moving to the results.

Internet-of-Things devices are gaining popularity rapidly. The more we use them to gather data, the more they will be an attack vector for bad actors. They already have a list of known IoT issues, and the more knowledge we have on these devices, the better our ability to correct their flaws will be. The Open Web Application Security Project (OWASP) puts out the top ten IoT flaws as a list. The most common flaw is weak, guessable, and hardcoded passwords as of 2018 (OWASP Internet of Things security team, 2018).

Many IoT devices are set in place only to be forgotten about. This is a real-world problem, as devices like this will generally be exploitable through physical access. These devices could easily be retrieved due to access vulnerabilities, exploited, and then put back in place. There are many different types of IoT devices in production today, so I intend to test many differing products. I utilized a penetration test to discover and exploit vulnerabilities on routers, extenders, locks, smartwatches, light bulbs, thermostats, speakers, cameras, televisions, and outlets. I believe these are devices most likely to be found within households today. This study can be carried out via multiple methodologies; however, I have followed the ISSAF methodology.

The Information System Security Assessment Framework allows me to reflect real-world scenarios and break the assessment into various domains and then detail specific testing for each individual domain (Rathore et al., 2006). I favored this testing framework because it used a specific set of common tools and it allows the assessment to be broken down into each phase. When this framework is used for penetration testing, the assessment will be divided into three phases. The first phase is the Planning and Preparation phase; in this phase, one will complete all the required documentation to perform the testing. This is when the team will be signed on, the rules of engagement will be defined and the in-scope and out-of-scope limits will be agreed upon (Rathore et al., 2006). Phase two is the assessment phase which is where most of the data will come from. This step is the action step, where the tools will be used to perform the assessment. Finally, phase three, is the reporting phase. In this phase, all the documentation is gathered and put into a format to convey how the assessment proceeded (Rathore et al., 2006).

The ISSAF has provided the following “layers” of a penetration test:

1. Information Gathering

2. Network Mapping

3. Vulnerability Identification

4. Penetration

5. Gaining Access & Privilege Escalation

6. Enumerating Further

7. Compromise Remote Users/Sites

8. Maintaining Access

9. Covering Tracks

10. Audit (optional – not a requirement of ISSAF penetration testing methodology)

These layers are each integral to the assessment. This is due to each subsequent layer providing a higher level of access than the previous. The first layer, Information Gathering, is the layer in which a lot of reconnaissance will occur. I utilized the internet to try to uncover any and all information pertinent to the assessment. Network Mapping is when the assessor will find out what operating systems are being ran, what ports and protocols are live on the targets, uncover live hosts and map out the perimeter of the network. Vulnerability Identification is the layer that live scanning takes place. The scans will discover banner vulnerabilities, look for known vulnerabilities and map the location for which one might be able to exploit. The Penetration layer is when the assessor will attempt to gain unauthorized access. Gaining Access and Privilege Escalation is the layer in which the system will become pwned or hacked, this is when the assessor can prove they have access to a system and will begin pivoting to either escalate privileges or compromise more systems. Enumerating Further is when the access can be

leveraged to perform scans inside the network, once a system is compromised, the assessor can utilize an internal network to try to advance their level of compromise. Compromise Remote Users/Sites is the layer in which the assessor will attempt to exfiltrate usernames and password hashes for offline cracking, the assessor can also attempt to pivot to other sites or domains within the network they have successfully infiltrated. The Maintaining Access layer is the layer in which the assessor will set up a covert channel, or backdoor, to ensure their ability to maintain the attack; this technique is used in an attempt to try and subvert any type of anti-virus or threat prevention lock out, kicking the assessor out of the system. Covering Tracks is trying to avoid detection, this layer is when the log files are adjusted, any files touched might need to be hidden, any type of doctoring to prevent someone from being able to detect the attack (Rathore et al, 2006). This layer requires a high level of understanding, there are a lot of files that would be touched through this process and a lot of noise would be created through the logs. A skilled assessor would understand that a cleared-out log file would look more suspicious than just leaving the attacks inside the logs, so the assessor will need a specific touch to ensure all the files they touched are hidden correctly and they would also need to ensure only the log entries they caused were removed from the system. The ISSAF has created one of the best frameworks for penetration testing; it provides a high level of detail and a good roadmap to perform an assessment. These reasons are why I have utilized this methodology to perform my research.

Setting up the Assessment

To configure this assessment, I started by creating a guest network and attaching my devices to the guest network. This is to test the ability to pivot from one of these devices onto the "business" or "home" network. Many of these devices will not be properly configured in the real-world scenario. Most of the devices are set up out-of-box, connected to a network, and then forgotten about. The first process is to factory-reset devices, in order to test an out-of-box setup. After the re-initialization, and connection, I can then proceed forward with the assessment. The first assessment was performed with Kali Linux as the penetration toolkit. The first tools utilized were p0f, unicornscan, hping3 and zeNmap. I used the information gathered from these tools to then move to using zeNmap, Oscanner, BED, and Lynis. The exploitation phase utilized Armitage, Linux Exploit Suggester, and crackle. The second assessment was performed with ParrotOS as the penetration toolkit. The first tools utilized were amap, masscan and recon-ng. I used the information gathered from these tools with Websploit and OpenVAS as well. The third assessment was performed with Commando VM as the penetration toolkit. The first tools utilized were Nmap and Watson. I attempted to exploit by using GhostPack, JuicyPotato, RottenPotatoNG, Vulcan and Metasploit.

The devices tested, in phases, were:

1. BowflexT22 treadmill
2. Google nest mini speaker
3. Lenovo smart alarm clock
4. Arlo HD security camera system
5. Ecobee gen 3 thermostat

6. GE 8000 BTU wifi smart air conditioner
7. NordicTrack SE9i
8. Kasa KP200 smart wifi outlet
9. Kasa HS103 smart wifi plug mini
10. Kasa HS200 smart wifi light switch
11. Sengled smart light bulb
12. LE LampUX smart led light bulb
13. Chamberlain MyQ smart garage hub
14. Fitbit Aria 2 scale
15. Roku Ultra

Instrumentation

To collect the data, I used the excel template shown in table 4:

Table 4

Assessment Template

Device	Threat	Tool used	CVE	Likelihood	Impact	Recommended actions

Note. This is the table template I utilized to record data from the assessments.

I utilized this template to track which device is susceptible to a specific threat. I also used this template to show if there is a specific CVE related to the threat discovered and the likelihood that it could be exploited, the likelihood will be rated as high, medium, or low; depending on how easily it could be exploited. I also was able to show the impact from this attack and the recommended actions to try to mitigate some of the vulnerabilities.

Data Collection

The data was recorded based on the same set of devices for each of the programs used with the three toolkits. I ran the assessment, collected three templates of data (one for each toolkit), and recorded all of my findings. Each assessment ran for about one week, giving ample time for a deep scan to complete with each security tool. All data was recorded, even data that shows no exploitations possible, to show the overall security confidence level on these devices.

Data Analysis Plan

In the impact column I recorded what may be compromised from a device. In the likelihood section I documented how successful this attack may be in a real-world scenario. Finally, by applying the ISSAF to the devices in the assessment, I determined if certain devices are more vulnerable than other devices in the assessment. I documented which tools are successful in uncovering vulnerabilities on IoT devices.

Trustworthiness

Credibility

Credibility, which is also referred to as internal validity ensures that the results of the research are credible and convincing (Neuman, 2009). Credibility helps contribute to the trustworthiness of the overall study (What is trustworthiness in Qualitative Research?, 2021). The credibility of this study begins by utilizing the known baseline of a factory-configured device. When pairing the default configuration of this device with the known version of the

security assessment tool I utilized, you can verify and confirm the steps taken to ensure this assessment is performed in a professional manner.

Transferability

The home network utilized in this study is similar to other home networks. The IoT devices utilized in this study are commonly used across the US. Transferability involves providing comprehensive information on the circumstances of a study, and this condition was met (Lodico et al., 2010). Transferability is being able to take information shared from this study so that it is valuable to others in comparable situations with related research questions and questions of practice (Marshall & Rossman, 2011). This study was performed based on factory default for both software configurations and a baseline with the security assessment tool. I followed the ISSAF throughout my study. Taking the vulnerabilities discovered and the operating system level of the devices, it would be possible to correlate the same vulnerabilities across different systems running the same operating system.

Dependability

Dependability was achieved in this study by offering a detailed narrative of the data gathering and examination methods. Dependability is showing the reliability of the data derived from this study (What is trustworthiness in qualitative research?, 2021). The data from this assessment was collected, categorized, and input along multiple steps throughout the process. The same process was used for each device and each toolkit. All the data is available for review to ensure it is verifiable throughout the process.

Confirmability

Confirmability is being able to replicate the results of this study while maintaining neutrality (What is trustworthiness in qualitative research?, 2021). The confirmability of the assessment is related to devices configured at the specific security and operating system level at the time the assessment is performed. When replicating this study, utilizing the same toolkits will also be critical. Cybersecurity is an evolving landscape that will manifest high levels of change from month-to-month. Overall, the confirmability will be present on like devices, around the same age, with the same level of operating system utilizing the same toolkit.

Chapter III Summary

In this chapter, I covered the methodology being employed to conduct this study. The study is based on leveraging the ISSAF penetration testing framework to perform a penetration test on 15 IoT devices. I also shared how the role the researcher will play in this assessment, as well as how this assessment was configured and carried out. In addition, I presented the template for how the data was collected, showing how to categorize, and document all the information gleaned from the assessment. I also addressed the trustworthiness of this assessment regarding credibility, transferability, dependability, and confirmability. With all of the above items documented, the next step will be carrying out the assessment and recording all of the data.

Chapter IV: Results

In this study, the problem I examine is the lack of security in IoT devices. IoT devices are in all aspects of the world today. These devices, however, are not always the most hardened devices and many people treat them as “plug and play” devices. Without understanding the security these devices have, this could be problematic for people willing to connect a number of these devices to their networks. In this chapter, I will provide the setting of the assessment. I will also share the demographics of the assessment, speaking on the devices and the nature of the testing. I will then present the findings from my assessment followed by answering the three research questions that are examined in this study: 1. What types of information could be compromised from an IoT device? 2. Will certain types of devices be more at risk than others? And finally, which out-of-box penetration toolkit is the most effective at exploiting IoT devices?

Setting

This study was performed within my personal residence on a separate network. This was to avoid any type of spillage or disruption to other devices or my personal network. Another task, purposed to avoid any leakage, was disconnecting all the devices connected to the main network at the time of study. I took these steps in order to assure I would not breach any of the devices that were not listed at the time of this assessment. The devices, at times, would be susceptible to unplanned power cycles or unplanned usage. Being that these devices are typically used on a day-to-day basis, some of the usage provided a difficulty to ensure these systems were connected at all times throughout the assessment. In a nutshell, family members in the home would turn these devices off without thinking about me performing the assessment. The results, once

collected, are relatively straightforward. The overall interpretation will fall on individual readers to justify what they may perceive as high or low risk.

Demographics

With no human participants in this study, the demographics are all technological in nature. The assessment is testing the security of 15 IoT devices. All of the devices are personally owned, and I am accepting that an assessment of this nature may possibly damage some of these devices. The data collection time was three-weeks. This process took more time than I had expected due to these devices power cycling off unexpectedly.

Findings

In this section, I will break down the findings from each individual operating system based on the research question. I will show the information gathered from each individual toolkit. Each toolkit has different programs and abilities which are discussed. I will begin with Kali Linux.

Kali Linux

Utilizing Kali Linux, I was able to negatively impact 80% (12 out of 15) of the IoT devices in my testing environment, as shown in the Table 5.

Table 5

Kali findings

Device	Threat	Tool used	CVE	Likelihood	Impact	Recommended Actions
Bowflex T22	DOS	h3ping		Medium	Availability	Firewall inside network or filter packets if router is capable.
Google Nest Mini	DOS	h3ping		Medium	Availability	Firewall inside network or filter packets if router is capable.
Lenovo Smart Alarm Clock	DOS	h3ping		Medium	Availability	Firewall inside network or filter packets if router is capable.
Arlo Camera System	Password	Aircrack	CVE-2019-3949	High	Privacy	Patch the system to the most up-to-date level
GE Window A/C	DOS	h3ping		Medium	Availability	Firewall inside network or filter packets if router is capable.
Kasa KP200	Scripting			Medium	Availability	Known Kasa script
	DOS	h3ping		Medium	Availability	Firewall inside network or filter packets if router is capable.
Kasa HS103	Scripting			Medium	Availability	Known Kasa script
	DOS	h3ping		Medium	Availability	Firewall inside network or filter packets if router is capable.

Kasa HS200	Scripting		Medium	Availability	Known Kasa script
	DOS	h3ping	Medium	Availability	Firewall inside network or filter packets if router is capable.
Sengled Smart Light Bulb	DOS	h3ping	Medium	Availability	Firewall inside network or filter packets if router is capable.
LE LampUX Smart Light Bulb	DOS	h3ping	Medium	Availability	Firewall inside network or filter packets if router is capable.
Chamberlain MyQ Hub	DOS	h3ping	Medium	Availability	Firewall inside network or filter packets if router is capable.
	Jamming		Medium	Function	
Roku Ultra	DOS	h3ping	Medium	Availability	Firewall inside network or filter packets if router is capable.

Note. This table shows the findings from the Kali Linux assessment.

Compromises achieved using Kali vary but include things such as the ability to compromise the network with cameras running to removing the ability for the devices to function. Utilizing the ISSAF Framework, I mapped out the network, finding the devices and the services that were running, and then proceeding to search for any vulnerabilities or possible exploits related to the system and services. Utilizing the ISSAF framework, I was able to answer the first research question: What types of information can be compromised from an IoT device?

Information Gathered. Through following the methodology laid out previously, I was able to build table 6 and begin answering what types of information could be compromised from

an IoT device. As presented in table 6, I list the device, obtained the IP address, hostname, and services.

Table 6

Kali assessment

Device	IP ADDR	Hostname	Services
Kasa plug	192.168.50.20	Hs103.lan	TCP 9999
LampUX	192.168.50.34	Esp_cd460e.lan	TCP 6668
Lenovo smart clock	192.168.50.43	Audiocast.lan	TCP 8008, 8009, 8012, 8443, 9000, 10001, 10007, 10101
Google nest speaker	192.168.50.67	Google-nest-mini.lan	TCP 8008, 8009, 8443, 9000, 10001
LampUX	192.168.50.91	Esp_cd935d.lan	TCP 6668
Kasa plug	192.168.50.94	Hs103.lan	TCP 9999
Kasa outlet	192.168.50.110	Kp200.lan	TCP 9999
Arlo base	192.168.50.118	Vmb4000.lan	TCP 554, 5061, 8100
MyQ	192.168.50.160	Myq-a71.lan	TCP 80, 443
LampUX	192.168.50.168	Esp_d63983.lan	TCP 6668
Kasa plug	192.168.50.183	Hs103.lan	TCP 9999
Sengled bulb	192.168.50.202	Sengled_wifia1960_white.lan	
Roku ultra	192.168.50.219	Rokuultra.lan	TCP 7000, 8060, 8080, 55846
Kasa outlet	192.168.50.224	Kp200.lan	TCP 9999
Lenovo smart clock	192.168.50.227	Audiocast.lan	TCP 8008, 8009, 8012, 8443, 9000, 10001, 10007, 10101
Kasa switch	192.168.50.239	Hs200.lan	TCP 9999
Bowflex	192.168.50.240	Bowflex-t22.lan	TCP 5555, 9999
Kasa plug	192.168.50.254	Hs103.lan	TCP 9999

Note. Information provided by the Kali assessment.

In Table 6, I noted the number of devices on the network, with their IP address, and the ports they have running. When focusing in on specific IP addresses, I gathered the IP address, MAC and operating system as presented in Table 7.

Table 7

Kali MAC/OS

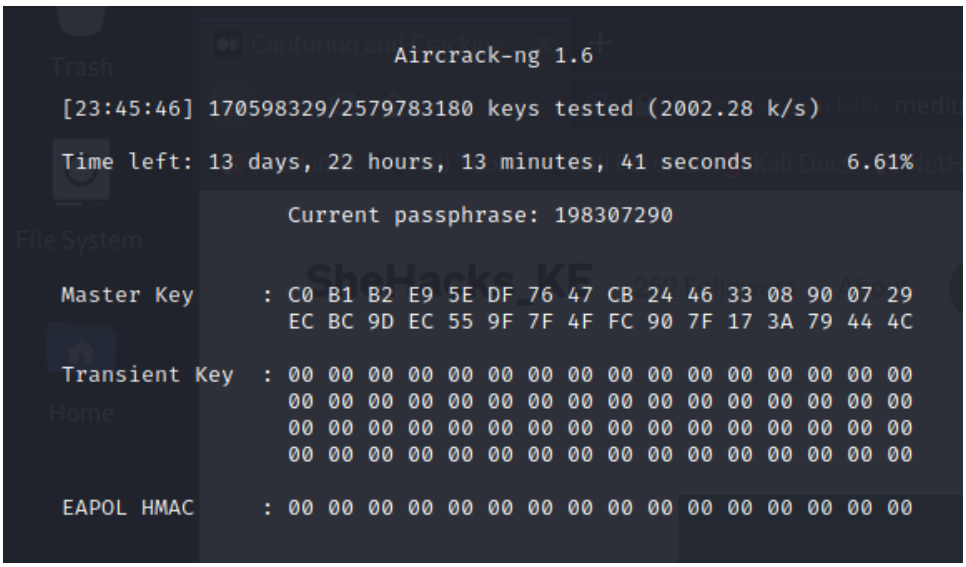
Device	IP	MAC	OS
Google nest speaker	192.168.50.67	F8:0F:F9:7B:D5:28	Linux 4.x
Kasa plug	192.168.50.94	D8:47:32:1A:E7:A1	Lightify ZigBee gateway
Kasa outlet	192.168.50.110	98:DA:C4:50:19:FA	Head Digital embedded
Arlo station	192.168.50.118	CC:40:D0:14:B2:DD	Linux 2.6.x
MyQ	192.168.50.160	64:52:99:AD:9B:BF	iHome embedded
Roku ultra	192.168.50.219	D8:31:34:5A:BC:F4	Linux 4.x
Kasa switch	192.168.50.239	D8:47:32:44:5C:9F	Lightify ZigBee gateway
Bowflex	192.168.50.240	D4:12:43:5A:55:2E	Android 5.x

Note. MAC and OS information provided by Kali Linux.

The information showed in the previous tables is remarkably beneficial for an attacker. The table shows the device, the device's MAC address, and the operating system the device is running. I was then able to examine what the exploits for each device may be possible. One important finding was when using Aircrack in an attempt to brute force the VMB4000. After capturing the Pairwise Master Key Identifier or the PMKID, I then was able to launch the brute force attack shown in figure 6.

Figure 6

Aircrack attempt



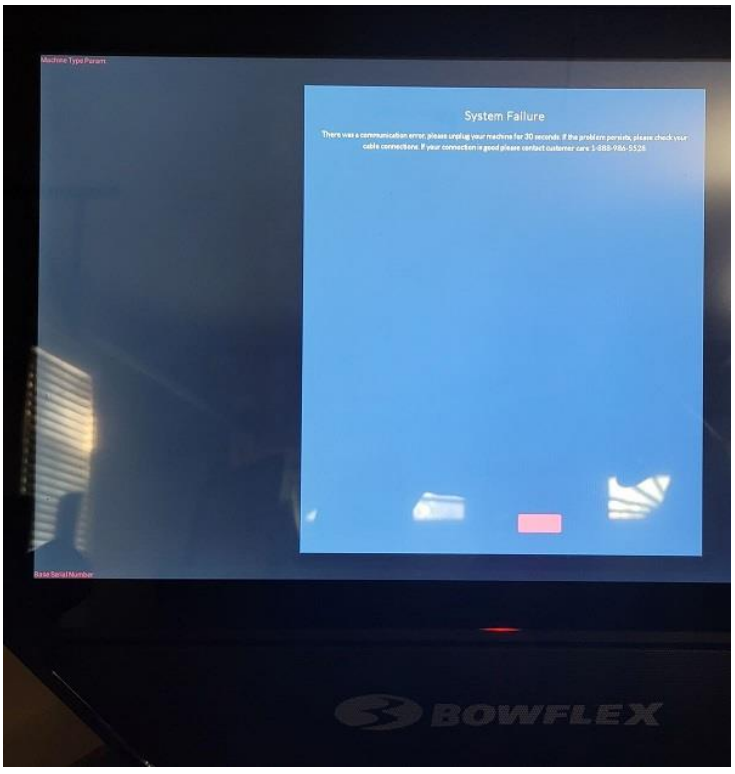
Note. This shows the aircrack working on cracking the password.

This is a relatively straightforward process, but due to time constraints, I was not able to let the crack finish. Time permitting, I would have been able to use the password crack file to brute force this connection. The VMB4000 is susceptible to this type of attack through CVE-2019-3949 (CVE-2019-3949, 2019). Using this method, after the connection I would have then had complete access to the Arlo network. This means I have the cameras and all information traveling to and from the base station. This is a high-level issue because an attacker can upload, download, or execute arbitrary code. In addition, Arlo has urged users to update their systems to prevent this type of attack (CVE-2019-3949, 2019). Another finding is that I was able to use a Python script to connect to the individual Kasa devices; the full script for the Kasa devices is located in Appendix A. Python-Kasa is an open-source project that allows an attacker to run a simple Python script that will find and connect users to these Kasa devices without the need of going through the app, bypassing layers of security (Python-Kasa, n.d.). Supported devices from

this project include the HS103 and HS200 that I have uncovered through previous steps (Python-Kasa, n.d.). I was able to monitor these devices to see if they are being utilized or not. This would allow someone to decipher if people are home or not. Availability is a crucial piece of the CIA triad, and this was largely exploitable by using the tool hping3. This tool allowed me to successfully attack multiple devices at the same time. This attack caused the Bowflex to completely fail and need a hard reboot as shown in Figure 7.

Figure 7

Bowflex failure



Note. This is the state of the Bowflex at the time of the scan.

All the information was pulled from the IoT devices on this network. With more time, I believe more information would be gathered. Next, I will look at answering the second research question: Will certain types of devices be more at risk than others?

Devices at Risk. To answer if certain devices are more at risk, I need to look at the previous findings from this assessment. The results show that the Kasa devices are more at risk than other devices. They had known scripts to work on multiple types of devices. Every single Kasa device scanned the same, with the same open port, TCP 9999. This is the port leveraged in using a Python script to control the devices, bypassing any security placed on the network (Python-Kasa, n.d.). The final research question will be answered next: Which open-source penetration toolkit is the most effective at exploiting IoT devices?

Operating System Effectiveness. Looking through all of the data I captured utilizing Kali Linux, it has definitely proven itself an effective tool for a penetration tester. This operating system provided a plethora of tools to use for this assessment. In terms of scope, Kali Linux performed exactly as I had expected. This tool was highly effective and highly efficient; it produced a high number of results that could easily be leveraged by an attacker. Adding more pieces of equipment to Kali Linux would provide an attacker a high probability of success exploiting IoT devices.

ParrotOS

Utilizing ParrotOS, I was also able to negatively impact 80% (12 out of 15) of the IoT devices in my testing environment, as shown in the table 8.

Table 8

ParrotOS findings

Device	Threat	Tool used	CVE	Likelihood	Impact	Recommended Actions
Bowflex T22	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if

						router is capable.
Google Nest Mini	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if router is capable.
Lenovo Smart Alarm Clock	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if router is capable.
Arlo Camera System	Password	Aircrack	CVE-2019-3949	High	Privacy	Patch the system to the most up-to-date level
GE Window A/C	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if router is capable.
Kasa KP200	Scripting			Medium	Availability	Known Kasa script
	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if router is capable.
Kasa HS103	Scripting			Medium	Availability	Known Kasa script
	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if router is capable.
Kasa HS200	Scripting			Medium	Availability	Known Kasa script
	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if router is capable.
Sengled Smart Light Bulb	DOS	Smurf6		Medium	Availability	Firewall inside network or filter packets if

					router is capable.
LE LampUX Smart Light Bulb	DOS	Smurf6	Medium	Availability	Firewall inside network or filter packets if router is capable.
Chamberlain MyQ Hub	DOS	Smurf6	Medium	Availability	Firewall inside network or filter packets if router is capable.
	Jamming		Medium	Function	
Roku Ultra	DOS	Smurf6	Medium	Availability	Firewall inside network or filter packets if router is capable.
	TCP timestamps	GVM	Medium	Availability	Disable TCP timestamps

Note. This shows the findings of the ParrotOS assessment.

Again, utilizing the ISSAF Framework, I mapped out the network, found the devices and the services that were running, and then proceeded to search for any vulnerabilities or possible exploits related to the system and services. This framework provided the ability to answer the first research question: What types of information can be compromised from an IoT device?

Information Gathered. Using the ISSAF I was able to build table 9, documenting the types of information I obtained from IoT devices in this study.

Table 9

ParrotOS assessment

Device	IP ADDR	Hostname	Services
Kasa plug	192.168.50.20	Hs103.lan	TCP 9999
LampUX	192.168.50.34	Esp_cd460e.lan	TCP 6668
Lenovo smart clock	192.168.50.43	Audiocast.lan	TCP 8008, 8009, 8012, 8443, 9000,

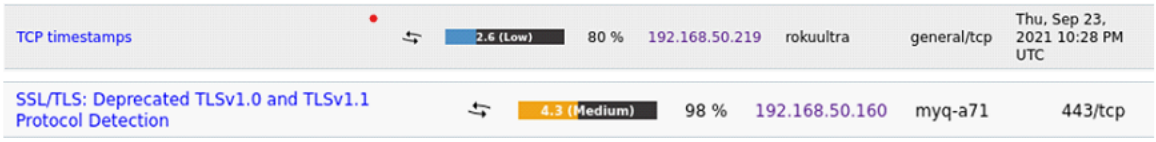
			10001, 10007, 10101
Google nest speaker	192.168.50.67	Google-nest-mini.lan	TCP 8008, 8009, 8443, 9000, 10001
LampUX	192.168.50.91	Esp_cd935d.lan	TCP 6668
Kasa plug	192.168.50.94	Hs103.lan	TCP 9999
Kasa outlet	192.168.50.110	Kp200.lan	TCP 9999
Arlo base	192.168.50.118	Vmb4000.lan	TCP 554, 5061, 8100
MyQ	192.168.50.160	Myq-a71.lan	TCP 80, 443
LampUX	192.168.50.168	Esp_d63983.lan	TCP 6668
Kasa plug	192.168.50.183	Hs103.lan	TCP 9999
Sengled bulb	192.168.50.202	Sengled_wifia1960_white.lan	
Roku ultra	192.168.50.219	Rokuultra.lan	TCP 7000, 8060, 8080, 55846
Kasa outlet	192.168.50.224	Kp200.lan	TCP 9999
Lenovo smart clock	192.168.50.227	Audiocast.lan	TCP 8008, 8009, 8012, 8443, 9000, 10001, 10007, 10101
Kasa switch	192.168.50.239	Hs200.lan	TCP 9999
Bowflex	192.168.50.240	Bowflex-t22.lan	TCP 5555, 9999
Kasa plug	192.168.50.254	Hs103.lan	TCP 9999

Note. This shows the results from the ParrotOS assessment.

Using the OpenVAS (Greenbone Vulnerability Manager) I was able to discover a TCP timestamp vulnerability and a deprecated SSL/TLS vulnerability, both shown in figure 8.

Figure 8

GVM vulnerabilities

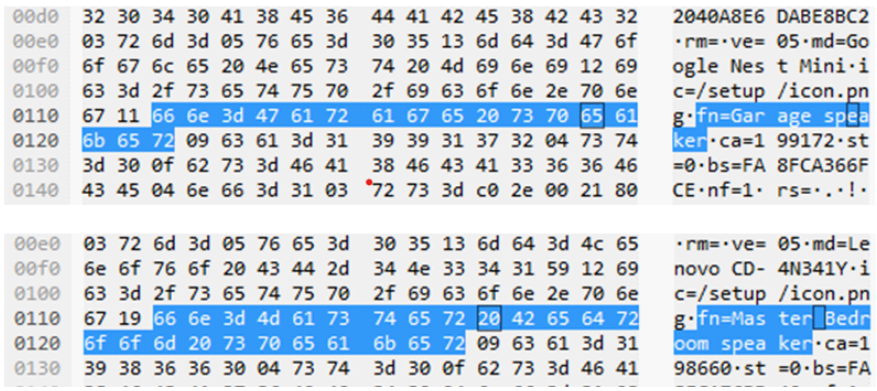


Note. This is the vulnerabilities revealed through GVM.

Both findings, the TCP timestamps and deprecated SSL/TLS, are fairly minor, but they do open up another attack vector someone could exploit. More problematic information came from using Wireshark. I was able to see the locations of both the Lenovo smart speaker and Google nest mini as shown in figure 9.

Figure 9

Locations

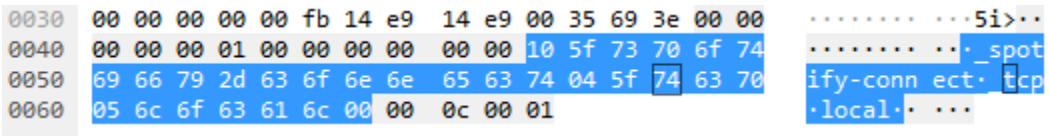


Note. This shows the location data from the two devices.

In Figure 9, you can see the locations of garage and bedroom. This opened another vector for exploitation if the attacker would be persistent. This went even further, when one of the speakers started playing a service; I was even able to see which service it was utilizing, seen in figure 10.

Figure 10

Services



Note. This shows the service running on the device.

After seeing this, I was a little shocked at how much information could be pulled from Wireshark with the majority of the connections being encrypted. When looking through the results, these devices are in a fairly secure state. I was able to utilize Python-Kasa, which was the same Python script utilized for the Kali assessment. I was also able to successfully deny access, or DoS, the same 13 devices from the Kali assessment with a tool named smurf6. The MyQ and Roku Ultra are susceptible to hash attacks through the TCP timestamp and SSL/TLS findings from OpenVAS. The Google nest mini and Lenovo smart speaker devices are giving away a little too much information, showcasing their locations within the home. The next step is to answer the second question: will certain types of devices be more at risk than others?

Devices at Risk. To answer if certain devices are more at risk I, again, need to look at the previous findings from this assessment. The results show that the Kasa devices are more at risk than other devices. These devices had known scripts to work on multiple types of devices. Every single Kasa device scanned the same, with the same open port, TCP 9999. This is the port leveraged in using a Python script to control the devices, bypassing any security placed on the network (Python-Kasa, n.d.). The final question will be looked at next: Which open-source penetration toolkit is the most effective at exploiting IoT devices?

Operating System Effectiveness. While completing the ParrotOS assessment, I was able to uncover enough information to start building an attack. This operating system provided a plethora of tools to use for this assessment, and had drivers already installed for add-on dongles. This tool proved itself very capable, but it did have some downfalls. This operating system was a little more complicated to update and had a few hiccups when it came to “out-of-box” databases that would not work without further configuration. ParrotOS did include a number of resources that were different from the previous assessment, but these resources are also open source.

Commando VM

When completing the final assessment using Commando VM, I was only able to negatively impact 33% (5 out of 15) of the IoT devices in my testing environment as shown in table 10.

Table 10

Commando VM findings

Device	**Threat**	**Tool used**	**CVE**	**Likelihood**	**Impact**	**Recommended actions**
Bowflex T22	DOS	Nmap		Medium	Treadmill will not function	A simple reboot fixes the issue
Google nest mini	SWEET32	Nmap	CVE-2016-6329	Medium	Clear-text data	Network segregation
	Privacy	Nmap		Medium	Location privacy	Use generic naming
Kasa HS103	Scripting			Medium	Availability	Known Kasa script
Kasa HS200	Scripting			Medium	Availability	Known Kasa script
Kasa KP200	Scripting			Medium	Availability	Known Kasa script

Note. This shows the findings from the Commando VM assessment.

Information Gathered. Using the ISSAF, I was able to gather data through information gathering, network mapping, vulnerability identification and penetration. Through this set of steps, I was able to build Table 11, providing what types of information could be compromised from an IoT device using Commando VM.

Table 11

Commando VM assessment

Device	IP ADDR	Hostname	Services
Kasa plug	192.168.50.20	Hs103.lan	TCP 9999
LampUX	192.168.50.34	Esp_cd460e.lan	TCP 6668
Lenovo smart clock	192.168.50.43	Audiocast.lan	TCP 8008, 8009, 8012, 8443, 9000, 10001, 10007, 10101
Google nest speaker	192.168.50.67	Google-nest-mini.lan	TCP 8008, 8009, 8443, 9000, 10001
LampUX	192.168.50.91	Esp_cd935d.lan	TCP 6668
Kasa plug	192.168.50.94	Hs103.lan	TCP 9999
Kasa outlet	192.168.50.110	Kp200.lan	TCP 9999
Arlo base	192.168.50.118	Vmb4000.lan	TCP 554, 5061, 8100
MyQ	192.168.50.160	Myq-a71.lan	TCP 80, 443
LampUX	192.168.50.168	Esp_d63983.lan	TCP 6668
Kasa plug	192.168.50.183	Hs103.lan	TCP 9999
Sengled bulb	192.168.50.202	Sengled_wifia1960_white.lan	
Roku ultra	192.168.50.219	Rokuultra.lan	TCP 7000, 8060, 8080, 55846
Kasa outlet	192.168.50.224	Kp200.lan	TCP 9999
Lenovo smart clock	192.168.50.227	Audiocast.lan	TCP 8008, 8009, 8012, 8443, 9000, 10001, 10007, 10101
Kasa switch	192.168.50.239	Hs200.lan	TCP 9999
Bowflex	192.168.50.240	Bowflex-t22.lan	TCP 5555, 9999
Kasa plug	192.168.50.254	Hs103.lan	TCP 9999

Note. This shows the results from the Commando VM assessment.

In this assessment, every step was performed using Nmap. There was no real “exploitation phase” as all the tools geared towards exploitation were for Active Directory. However, using Nmap’s script scanning, I gathered more information than before. This occurred even though I was using similar tools with the other toolkits. For example, I was able to obtain the multicast address for systems, the services running on systems, their serial numbers, names of the products and even their locations in the house. I was able to show locations, IPv4 and IPv6 addresses, serial numbers, and model numbers shown in figure 11.

Figure 11

IPv4, IPv6, serial, models

Note. This shows the addresses and serials for the devices.

Now I have a little more information, than in previous assessments. This would allow an attacker to determine where they would need to be for further exploitation. In addition, while working my way through testing the vulnerability scripts, I noticed the connection resetting on .240, which

was the treadmill. Upon this, I walked out to the treadmill, and found that the system had completely malfunctioned as shown in figure 12.

Figure 12

Failed Bowflex

Note. This shows the disconnected state of the Bowflex.

By running the Nmap script, I disconnected the Bowflex from the onboard OS causing the need for a reboot. This was where the assessment hit a stopping point. The exploit capabilities of Commando VM are largely tied to Windows Active Directory. With no windows devices in play, I could not do more. With all the exploitation tools being for Windows devices, this is where I was forced into an early stopping point. I was able to uncover a lot of information that could easily be transitioned into other tools for exploitation, but that is outside the scope of this

assessment. This leads into the second research question: Will certain types of devices be more at risk than others?

Devices at Risk. To answer if certain devices are more at risk I, again, need to look at the previous findings from this assessment. The results from Commando VM are the same as determined from Kali Linux and ParrotOS. Kasa is more susceptible across the board than other devices. Every single Kasa device scanned the same, with the same open port, TCP 9999. This is the port leveraged in using a Python script to control the devices, bypassing any security placed on the network (Python-Kasa, n.d.). The final question will be looked at next: Which open-source penetration toolkit is the most effective at exploiting IoT devices?

Operating System Effectiveness. Commando VM was not built for this type of IoT 'out of the box' assessment. This operating system was designed with scanning windows devices in mind. The majority of the tools are geared for attacking an active directory forest. This tool would be fun to set loose on a corporate environment to see what someone would be capable of doing. But in terms of this assessment, Commando VM was least successful in providing tools I could utilize to breach IoT devices.

Chapter IV Summary

In this study I answered three research questions: 1. What types of information can be compromised from an IoT device? 2. Will certain types of devices be more at risk than others? and 3. Which open-source penetration toolkit is the most effective at exploiting IoT devices? I was able to determine the IP address and hostnames of all 15 devices. On 47% (7 of 15) of the IoT devices, I was able to obtain the location of the rooms these devices were in. On 80% (12 of 15) of the IoT devices, I was able to render useless with a DoS attack. Finally, I was able to

uncover one high-risk vulnerability with known exploits of the VMB4000 Arlo device CVE-2019-3949. Kasa devices are more susceptible to an attack via scripting. The scripting attack is known for all the Kasa devices I owned. This attack transcended the “type-model” series and was applicable to multiple devices. As for the third question, Kali Linux, by far, was the most “ready” to go and offered the best tools for someone who understands cyber security. This tool provided the most functionality upon boot up. Kali Linux included the ability to install any of the tools not included with Kali onto the system to cover any tool one might prefer from ParrotOS or Commando VM. ParrotOS offered more drivers for add-on services. For example, ParrotOS comes with ubertooth drivers and software already installed where Kali does not. ParrotOS also has the ability to scan via Low-Power Wireless Personal Area Networks (6LoWPAN). Both features were outside of scope but would be beneficial to a hacker. ParrotOS was a little harder to work with than Kali. I needed to make a couple of changes to ParrotOS files to update the system. This was due to some of the mirrors no longer being active. Therefore, this OS required more research than Kali just to update. Commando VM, is best used for scanning Active Directory and Windows systems. This toolkit was not intuitive or useful for accessing IoT devices. In Chapter five, I discuss key findings and how to build from what is known to ensure these devices are devices we can trust in our everyday lives.

Chapter V: Summary and Conclusions

IoT devices inherently are not secure with an out-of-box setup. As a result, users are not aware of this vulnerability. I was able to determine the IP address and hostnames of all 15 devices. On 47% (7 of 15), I was able to obtain the location of the rooms these devices were in. On 80% (12 of 15), I was able to render them useless with a DoS attack. Finally, I was able to uncover one high-risk vulnerability, CVE-2019-3949 on the VMB4000 Arlo device. Kali Linux provided the best platform for examining IoT devices in this study. ParrotOS worked well also but the user had to configure programs before any scanning could occur. The majority of the findings were privacy related. For example, giving out location information with some of the devices. In the rest of Chapter V, I will analyze the findings of this assessment, go over some of the limitations, give my honest recommendations, and discuss what all of this means for the IT community. First, I will go over the findings.

Analysis of Findings

The findings from this assessment confirm the current knowledge on IoT device security. Through this assessment, I showed that IoT devices give up more information than what should be acceptable in terms of security. This confirms the literature showing privacy is an issue when it comes to IoT devices (Kumar & Mallick, 2018). Showing this in a formalized study may educate users on how to combat some of the security vulnerabilities these devices currently carry. The findings from this assessment also show the need for a user to continually update their systems when an update is available; the literature already shows the human element being an issue when it comes to keeping systems patched (Sadique et al., 2018). This was highlighted in the Arlo base station finding, CVE-2019-3949. This study improved the knowledge in this field

by identifying outdated TLS settings on the MyQ and Roku Ultra devices. This study also showed the ability to DoS IoT devices, again confirming the literature on this topic (Jose & Vijyalakshmi, 2018). All the findings from this assessment will benefit the IT community's awareness of IoT security. If the community can be more aware of the vulnerabilities and ways that an attacker can disrupt or exploit the services these devices run on, then the community can be more proactive at finding ways to prevent these attacks.

The current literature speaks on the privacy leaks from IoT devices, this was shown through being able to uncover locations of the Lenovo speaker, Google nest mini, and MyQ devices (Kumar & Mallick, 2018). After having this information an attacker will be able to leverage what they know in terms of locations and other services to build their attack further. Understanding what is capable from the information gathered ties into solving the issues of IoT security. The findings I highlighted in this assessment all lead to gathering more and more information. The more information an attacker can leverage, the more likely the attacker will succeed in compromising the system or device. Theoretically, an attacker would take all the information gathered from this assessment and build upon that to further an attack. Utilizing the proper tools and equipment would drastically increase the level of confidence in an attack on these devices. This assessment highlighted how simple it is to start building a database of information to increase the chances of success. Most of the information from this assessment could be leveraged with plug and play devices to exploit less secure forms of communication. This is one area that IoT struggles in (Biham & Neumann, 2020). The community knows that wireless transmissions can be secure, but some of the other forms of communication are made more for convenience than security. This study also showed Kali Linux to be the most user-friendly. Kali Linux had the least number of issues updating the system or ensuring the databases

for scanning were functional. This system was also ready to scan all types of operating systems, instead of being geared for Windows devices. Looking at the second research question, Kasa showed more devices that were susceptible to being attacked than that of the other brands owned.

Limitations

The first and main limitation of this study was the number of devices I attempted to breach. I only scanned the devices I had possession of at a singular point in time. This could be built upon, and should be built upon, by using additional IoT devices that are currently available to the public. Google makes everything from locks to cameras. Amazon has a long list of devices such as the echo and show, the list goes on. I only followed one methodology throughout this assessment. This limitation required me to work within a certain boundary of rule sets; using more than one methodology might be another way to improve the results from this assessment. Another limitation of this assessment is that every device owned by a different person may be configured differently on a differently configured network. When it came to the devices I was using, I knew their state of configuration. If this assessment were to be performed without that type of knowledge, with devices being fully patched, passwords changed, services disabled; the findings may have been different.

Recommendations

Based off the findings in this study, I recommend further study on fully patched systems. In this study, I looked at factory default systems. Researching fully patched systems may give knowledge on what holes were opened or closed through patching. Testing the security of all Kasa devices is another possible area for study. Assessing the Kasa family could provide

valuable information about what security features these devices may benefit from using. Another area of further study is utilizing tools that breach Bluetooth Low Energy or BLE and IPv6 over Low-Power Wireless Personal Area Networks or 6LoWPAN. These tools did not fit the scope of this assessment, but I believe would have been beneficial at several stages of exploitation. Additional information about how susceptible these devices are to the vulnerabilities in Bluetooth, or the information that can be easily obtained using 6LoWPAN would be beneficial. Finally, I think utilizing more brands of IoT devices and examining entire brand suites of devices would provide useful information. This study shows multiple devices carry different problems, so adding to the study a higher number of devices and brands would also provide beneficial data.

Implications

The need for secure IoT devices has grown significantly over that past years. During my study, I discovered that some of the IoT systems are still utilizing passwords that are capable of being brute force attacked. These systems need to come ‘out of the box’ with randomized string passwords to help prevent exploitation. Additional security measures need to happen at the manufacturer level. For example, one IoT device in my study, Ecobee, is heading in the right direction with their systems. This company takes the security patches away from the end-user. If there were more companies that chose to take this approach, the end-user would benefit by not having to remember to patch their systems. This would have prevented me from finding CVE-2019-3949 on the Arlo system. Also, as an end-user, a good plan of action would be to segregate the IoT devices on the network with VLANs. Most users are not of aware of how to setup a VLAN. By using this approach, I would not have been able to successfully DoS the systems if they were not attached to the specific network I was scanning from. This assessment also shows

the need for the IT community to begin working on standards and metrics when it comes to IoT devices (Kumar & Mallick, 2018). When devices can be made from multiple countries, following multiple standards, there will be varying levels of security. An overarching standard would help in preventing some of these issues.

Conclusion

In conclusion, I determined the types of information accessible from IoT devices using three different types of Open-Source Software. The types of information were privacy-related, service-related, and system-related. Another finding is by utilizing Kali Linux, I was able to exploit more devices than using the other two types of software in the study. I also found that the Kasa family of devices were more vulnerable to exploitation than others. One way to build on this study would be to use an ubertooth device to provide an ability to attack Bluetooth. There are 1000's of different types of IoT devices available today. Researchers can build off this study by conducting more assessments while utilizing other tools that have shown to be highly capable for attacking a wide variety of devices. Through this study, I was able to show how easy and quick a threat actor can download one of these open-source operating systems and grab information on a network. The more the IT community can highlight the issues with these devices, suggest mitigation, and secure their IoT devices, the more secure users' privacy will be.

References

Attacks on IoT devices continue to escalate. Help Net Security. (2020, October 22). https://www.helpnetsecurity.com/2020/10/28/attacks-on-IoT-devices-continue-to-escalate/.

Barteaux, J. (2019, March 29). *Commando VM: The first of its kind windows offensive distribution*. FireEye. https://www.fireeye.com/blog/threat-research/2019/03/commando-vm-windows-offensive-distribution.html.

Bello, O., & Zeadally, S. (2016). Intelligent device-to-device communication in the internet of Things. *IEEE Systems Journal*, *10*(3), 1172–1182. https://doi.org/10.1109/jsyst.2014.2298837

Biham, E., & Neumann, L. (2020). Breaking the bluetooth pairing – the fixed coordinate invalid curve attack. *Lecture Notes in Computer Science*, 250–273. https://doi.org/10.1007/978-3-030-38471-5_11

Buckley, J. J. (2006). *From rfid to the internet of things: Pervasive networked systems*. Office for publications of the European commission.

Cambridge Dictionary: Find definitions, meanings & translations. (n.d.). https://dictionary.cambridge.org/us/.

CVE-2019-3949. CVE. (2019, January 3). Retrieved October 14, 2021, from https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-3949.

Dempsey, K. , Johnson, L. , Scholl, M. , Stine, K. , Clay, A. , Orebaugh, A. , Chawla, N. and Johnston, R. (2011), Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations, Special Publication (NIST SP), National Institute of Standards and Technology, Gaithersburg, MD, [online], https://doi.org/10.6028/NIST.SP.800-137 (Accessed April 25, 2021)

Dragomir, D., Gheorghe, L., Costea, S., & Radovici, A. (2016). A survey on secure communication protocols for IoT systems. *2016 International Workshop on Secure Internet of Things (SIoT)*. https://doi.org/10.1109/sIoT.2016.012

Editor, C. S. R. C. C. (n.d.). *CVE - Glossary*. CSRC. https://csrc.nist.gov/glossary/term/CVE.

Farhan, L., Kharel, R., Kaiwartya, O., Quiroz-Castellanos, M., Alissa, A., & Abdulsalam, M. (2018). A concise review on Internet of Things (IoT) problems, challenges and opportunities. *2018 11th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP)*. https://doi.org/10.1109/csndsp.2018.8471762

Grassi, P. A., Garcia, M. E., & Fenton, J. L. (2017, June). *threat - glossary*. CSRC. https://csrc.nist.gov/glossary/term/threat.

Heiser, E., & Ryan, C. (2019). Four steps to IoT security [white paper]. UBlox. https://uploads-ssl.webflow.com/5fa429174cc2b89c3d4b6bd4/606332b37b3302037d2e72e0_Download-White-Paper-4-Steps-to-IoT-Security-ublox.pdf.

Herzog, P. (2010). The open source security testing methodology manual: Contemporary security testing and analysis. *ISECOM*.

How IoT Works - 4 main components of IoT system. DataFlair. (2018, September 15). https://data-flair.training/blogs/how-IoT-works/.

Jose, D. V., & Vijyalakshmi, A. (2018). An overview of security in internet of things. *Procedia Computer Science*, *143*, 744–748. https://doi.org/10.1016/j.procs.2018.10.439

Kali docs: Kali Linux documentation. Kali Linux. (2021, February 19). http://docs.kali.org/.

Kumar, N. M., & Mallick, P. K. (2018). Blockchain technology for security issues and challenges in IoT. *Procedia Computer Science*, *132*, 1815–1823. https://doi.org/10.1016/j.procs.2018.05.140

Kumar, R. P., & Smys, S. (2018). A novel report on architecture, protocols and applications in Internet of Things (IoT). *2018 2nd International Conference on Inventive Systems and Control (ICISC)*. https://doi.org/10.1109/icisc.2018.8398986

Lack of security in internet of things devices. (2014). *Network Security*, *2014*(8), 2. https://doi.org/10.1016/s1353-4858(14)70075-3

Lodico, M., Spaulding, D., & Voegtle, K. (2010). Methods in educational research: From theory to practice. San Francisco, CA: Jossey-Bass.

Mahmoud, R., Yousuf, T., Aloul, F., & Zualkernan, I. (2015). Internet of things (IoT) security: Current status, challenges and prospective measures. *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*. https://doi.org/10.1109/icitst.2015.7412116

Marshall, C., & Rossman, G. (2011). *Designing qualitative research.* Thousand Oaks, CA: Sage

Mehta, R., Sahni, J., & Khanna, K. (2018). Internet of Things: Vision, applications and challenges. *Procedia Computer Science*, *132*, 1263–1269. https://doi.org/10.1016/j.procs.2018.05.042

Miller, C. (2019). Lessons learned from hacking a car. *IEEE Design & Test*, *36*(6), 7–9. https://doi.org/10.1109/mdat.2018.2863106

Miloslavskaya, N., & Tolstoy, A. (2018). Internet of Things: information security challenges and solutions. *Cluster Computing*, *22*(1), 103–119. https://doi.org/10.1007/s10586-018-2823-6

Minerva, R., Biru, A., & Rotondi, D. (2015). Towards a definition of the Internet of Things (IoT). *IEEE Internet Initiative*.

News Release. News release | IHS markit online newsroom. (2017, October 24). https://news.ihsmarkit.com/prviewer/release_only/slug/number-connected-IoT-devices-will-surge-125-billion-2030-ihs-markit-says.

Neuman, W. (2009). *Social research methods: qualitative and quantitative approaches.* Boston, MA: Allyn and Bacon.

O'Neill, M. (2016). Insecurity by design: today's IoT device security problem. *Engineering*, *2*(1), 48–49. https://doi.org/10.1016/j.eng.2016.01.014

OWASP Internet of Things security team. (2018). https://owasp.org/www-project-internet-of-things/.

Python-Kasa/Python-Kasa: Python API for TP-link Kasa smarthome products. GitHub. (n.d.). Retrieved October 14, 2021, from https://github.com/Python-Kasa/Python-Kasa.

Rathore, B., Brunner, M., Dilaj, M., Herrera, O., Brunati, P., Subramaniam, R., … Chavan, U. (2006). Information Systems Security Assessment Framework (ISSAF) Draft 0.2.1. *Open Information Systems Security Group.*

Riahi, A., Challal, Y., Natalizio, E., Chtourou, Z., & Bouabdallah, A. (2013). A systemic approach for IoT security. *2013 IEEE International Conference on Distributed Computing in Sensor Systems*. https://doi.org/10.1109/dcoss.2013.78

Sadique, K. M., Rahmani, R., & Johannesson, P. (2018). Towards security on Internet of Things: Applications and challenges in technology. *Procedia Computer Science*, *141*, 199–206. https://doi.org/10.1016/j.procs.2018.10.168

Sami, S., Dai, Y., Tan, S. R., Roy, N., & Han, J. (2020). Spying with your robot vacuum cleaner. *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. https://doi.org/10.1145/3384419.3430781

Scarfone, K. A., Souppaya, M. P., Cody, A., & Orebaugh, A. D. (2008). Technical guide to information security testing and assessment. https://doi.org/10.6028/nist.sp.800-115.

SecurityCompass (2019). Internet of Things: security vulnerabilities and challenges [White paper]. Security Compass. https://resources.securitycompass.com/whitepapers/internet-of-things-security-vulnerabilities-and-challenges.

Sedrati, A., & Mezrioui, A. (2018). A survey of security challenges in Internet of Things. *Advances in Science, Technology and Engineering Systems Journal*, *3*(1), 274–280.

Sobin, C. C. (2020). A survey on architecture, protocols and challenges in IoT. *Wireless Personal Communications*, *112*(3), 1383–1429. https://doi.org/10.1007/s11277-020-07108-5

System requirements - Parrot documentation. (n.d.). https://www.parrotsec.org/docs/info/system-requirements/.

Techopedia. (2011, November 3). *What is a switch? - definition from Techopedia*. Techopedia.com. https://www.techopedia.com/definition/2306/switch-networking.

Unit42 (2020). IoT threat report [White paper]. Paloalto networks. https://start.paloaltonetworks.com/unit-42-IoT-threat-report.

Vidales, M. (2017, May 16). *802.15.4 wireless for Internet of Things developers*. Medium. https://blog.helium.com/802-15-4-wireless-for-internet-of-things-developers-1948fc313b2e.

Visoottiviseth, V., Akarasiriwong, P., Chaiyasart, S., & Chotivatunyu, S. (2017). PENTOS: Penetration testing tool for Internet of Thing devices. *TENCON 2017 - 2017 IEEE Region 10 Conference*. https://doi.org/10.1109/tencon.2017.8228241

What is trustworthiness in qualitative research? Statistics Solutions. (2021, June 23). https://www.statisticssolutions.com/what-is-trustworthiness-in-qualitative-research/.

Appendix A. Python-Kasa script

```
        """Module for smart plugs (HS100, HS110, ..)."""
import logging
from typing import Any, Dict

from kasa.smartdevice import DeviceType, SmartDevice, requires_update

_LOGGER = logging.getLogger(__name__)


class SmartPlug(SmartDevice):
    """Representation of a TP-Link Smart Switch.

    To initialize, you have to await :func:`update()` at least once.
    This will allow accessing the properties using the exposed properties.

    All changes to the device are done using awaitable methods,
    which will not change the cached values, but you must await :func:`update()` separately.

    Errors reported by the device are raised as :class:`SmartDeviceException`s,
    and should be handled by the user of the library.

    Examples:
        >>> import asyncio
        >>> plug = SmartPlug("127.0.0.1")
        >>> asyncio.run(plug.update())
        >>> plug.alias
        Kitchen

        Setting the LED state:

        >>> asyncio.run(plug.set_led(True))
        >>> asyncio.run(plug.update())
        >>> plug.led
        True

    For more examples, see the :class:`SmartDevice` class.
    """

    def __init__(self, host: str) -> None:
        super().__init__(host)
        self.emeter_type = "emeter"
        self._device_type = DeviceType.Plug

    @property  # type: ignore
```

```
@requires_update
def is_on(self) -> bool:
    """Return whether device is on."""
    sys_info = self.sys_info
    return bool(sys_info["relay_state"])

async def turn_on(self, **kwargs):
    """Turn the switch on."""
    return await self._query_helper("system", "set_relay_state", {"state": 1})

async def turn_off(self, **kwargs):
    """Turn the switch off."""
    return await self._query_helper("system", "set_relay_state", {"state": 0})

@property  # type: ignore
@requires_update
def led(self) -> bool:
    """Return the state of the led."""
    sys_info = self.sys_info
    return bool(1 - sys_info["led_off"])

async def set_led(self, state: bool):
    """Set the state of the led (night mode)."""
    return await self._query_helper(
        "system", "set_led_off", {"off": int(not state)}
    )

@property  # type: ignore
@requires_update
def state_information(self) -> Dict[str, Any]:
    """Return switch-specific state information."""
    info = {"LED state": self.led, "On since": self.on_since}
    return info
```