ASSESSING RISK IN IOT DEVICES

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Computer Science

by

David Levitsky

December 2018
COMMITTEE MEMBERSHIP

TITLE: Assessing Risk in IoT Devices

AUTHOR: David Levitsky

DATE SUBMITTED: December 2018

COMMITTEE CHAIR: Zachary Peterson, Ph.D.
Professor of Computer Science

COMMITTEE MEMBER: Bruce DeBruhl, Ph.D.
Professor of Computer Science

COMMITTEE MEMBER: Foaad Khosmood, Ph.D.
Professor of Computer Science
ABSTRACT
Assessing Risk in IoT Devices

David Levitsky

The explosive growth of the Internet of Things ecosystem has thrust these devices into the center of our lives. Unfortunately, the desire to create these devices has been stronger than the one to secure them. Recent attacks have shown us ignoring security in Internet of Things devices can cause severe harm in both a digital and physical sense. This thesis outlines a framework for developers and managers to assess the risk of IoT devices using a weighted scoring system across five different categories. Our case studies suggest that devices with higher security considerations have a better security posture and lower risk than those without.
ACKNOWLEDGMENTS

Thanks to:

- My parents, for their unconditional support.

- Calvin Laverty and Michael Hehman, for being excellent research partners on a previous project and contributing to the development of the risk assessment and case studies portion of this paper.

- Andrew Guenther, for uploading this template.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>CHAPTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>ix</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Current Issues</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Proposed Solution</td>
<td>3</td>
</tr>
<tr>
<td>1.2.1 Risk Factors</td>
<td>3</td>
</tr>
<tr>
<td>1.2.2 Risk Assessments</td>
<td>3</td>
</tr>
<tr>
<td>1.2.3 Case Studies</td>
<td>4</td>
</tr>
<tr>
<td>2 Background</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Definitions</td>
<td>5</td>
</tr>
<tr>
<td>2.1.1 Internet of Things</td>
<td>5</td>
</tr>
<tr>
<td>2.1.2 Risk Assessment</td>
<td>5</td>
</tr>
<tr>
<td>2.1.3 Zigbee</td>
<td>5</td>
</tr>
<tr>
<td>2.1.4 Z-Wave</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Related Works</td>
<td>6</td>
</tr>
<tr>
<td>2.2.1 OWASP</td>
<td>6</td>
</tr>
<tr>
<td>2.2.2 NIST Draft 8200</td>
<td>7</td>
</tr>
<tr>
<td>2.2.3 Secure IoT Project</td>
<td>8</td>
</tr>
<tr>
<td>2.2.4 Cloud Security Alliance</td>
<td>8</td>
</tr>
<tr>
<td>2.2.5 IoT Village - DEF CON</td>
<td>8</td>
</tr>
<tr>
<td>2.2.6 Microsoft</td>
<td>9</td>
</tr>
<tr>
<td>2.2.7 Threat Modeling: Designing for Security</td>
<td>9</td>
</tr>
<tr>
<td>3 Surface Attack Vectors</td>
<td>11</td>
</tr>
<tr>
<td>3.1 Device</td>
<td>11</td>
</tr>
<tr>
<td>3.2 Network Traffic and Services</td>
<td>14</td>
</tr>
<tr>
<td>3.2.1 Protocol</td>
<td>14</td>
</tr>
<tr>
<td>3.2.2 Transport Encryption</td>
<td>15</td>
</tr>
<tr>
<td>3.2.3 Ports</td>
<td>15</td>
</tr>
</tbody>
</table>
BIBLIOGRAPHY ................................................. 52

APPENDICES
## LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Example of OWASP Vulnerability Description.</td>
<td>7</td>
</tr>
<tr>
<td>3.1</td>
<td>Surface Attack Vectors For Device Category.</td>
<td>13</td>
</tr>
<tr>
<td>3.2</td>
<td>Surface Attack Vectors For Network Traffic Category.</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>Surface Attack Vectors For Mobile Category.</td>
<td>20</td>
</tr>
<tr>
<td>3.4</td>
<td>Surface Attack Vectors For Web Category.</td>
<td>23</td>
</tr>
<tr>
<td>4.1</td>
<td>Classification of Attack Categories of an IoT Device.</td>
<td>24</td>
</tr>
<tr>
<td>4.2</td>
<td>Visual Representation of Different Risk Scores.</td>
<td>25</td>
</tr>
<tr>
<td>4.3</td>
<td>Risk Scoring Components of the Device Attack Category.</td>
<td>27</td>
</tr>
<tr>
<td>4.4</td>
<td>Risk Scoring Components of the Network Traffic Attack Category.</td>
<td>30</td>
</tr>
<tr>
<td>4.5</td>
<td>Risk Scoring Components of the Mobile Attack Category.</td>
<td>32</td>
</tr>
<tr>
<td>4.6</td>
<td>Risk Scoring Components of the Web Attack Category.</td>
<td>35</td>
</tr>
</tbody>
</table>
Over the past few years, the Internet of Things (IoT) ecosystem has been growing exponentially. These devices range from devices with high computing power, such as cameras and voice assistants, to devices with low computing power such as light bulbs or light switches. By 2020, it is estimated that there will be more than 20 billion of these devices in use by consumers [24]. Our interactions with these devices have become increasingly intimate. While laptops and televisions have existed in our homes for a long time, adding locks, temperature sensors, and even heart sensors to that environment has really increased the impact that these devices have on our lives. As manufacturers race to develop the next hot device to captivate the market, time to market becomes vital. This emphasis on development speed, combined with a lack of security experience for most IoT companies, leads to a lack of considerations for security of the product [43]. However, recent attacks have shown us ignoring security in Internet of Things devices can cause severe consequences. In 2015, HP published a report that surveyed the ten most popular internet-connected devices on the market and discovered that 70% of them had some sort of security or privacy concern [24].

1.1 Current Issues

Like computers and laptops, IoT devices are vulnerable to different network and software attacks. The large number of devices makes them a very attractive target for cybercrime. In 2016, the Mirai botnet brought down a significant portion of the Internet when it self-propagated itself in such a high quantity that its distributed denial of service attack overwhelmed a major DNS provider on the East Coast and
shut down websites such as Netflix [33]. This botnet exploited Internet of Things devices, primarily cameras that had weak default passwords [41]. In 2014, WeMo had such severe vulnerabilities discovered in its smart lightbulbs that security researchers recommended completely unplugging these devices as the only safe and secure course of action [27]. In 2015, hackers at DEF CON demonstrated how to take advantage of a vulnerable communication protocol called ZigBee to compromise a home network by using an Internet of Things device as an easy entry vector [34].

Although IoT devices can be exploited to harm consumers, the use of these devices extends to matters of national security as well. In 2010, a worm by the name of StuxNet was discovered to specifically target embedded controllers in nuclear centrifuges and break them by causing them to spin too quickly. The worm was so destructive that it managed to temporarily halt the development of Iran’s uranium enrichment infrastructure, causing over 1,000 machines to physically degrade [25]. Stuxnet targeted SCADA systems, or supervisory control and data acquisition systems that are commonly seen in assembly lines or power plants. The creation of poorly secured IoT devices to be used in industrial control systems can have a devastating impact on any country’s infrastructure. IoT devices are extremely prevalent in our lives, and impacts from exploiting them have already been demonstrated [24]. Therefore, preventative measures against threats towards Internet of Things devices are no longer a sign of good will, but rather a requirement in order to protect participants in the Internet ecosystem and in a physical sense.

IoT devices are very unique due to their merging of the digital and physical worlds. Cheap processors provide the ability to add digital intelligence to almost any physical device and add it to the IoT ecosystem, from a stuffed animal to a refrigerator. However, few IoT devices give much thought to even security basics, like encrypting data in transit and at rest. Many devices also lack the capability to be patched, which means they are permanently at risk even after flaws are discovered [42]. Adding these
devices to the Internet while not implementing security mechanisms is negligent and can have severe consequences - such as a power plant being shut off or a nuclear plant failing.

1.2 Proposed Solution

Although both consumer and industrial IoT devices are being integrated into homes and businesses, there have been limited frameworks to assess the security risks of these devices. In this thesis, we enumerate the common risks of IoT devices and propose a model to quantify them.

1.2.1 Risk Factors

We characterize the different types of risks to IoT devices using four attack categories, which helps outline threats in different components of the IoT device. We choose to have four main categories: physical device, network traffic, mobile interfaces, and web interfaces. Additionally, we create a constant to model unknown risk for a device in order to cover threats that cannot be predicted until they are discovered, such as zero day vulnerabilities. Because of this, no device is without risk.

1.2.2 Risk Assessments

We propose two separate formulas in order to help determine the risk of a given IoT device. The first formula has been developed to model risk aggressively and highlight any single points of failures in a device. The formula works by looking at the risk scores assigned to each attack category in the device. The risk of the entire device becomes equal to the risk of the most vulnerable component of the device. Although we split an IoT device into five separate attack categories, we focus solely on the risk of
individual components to identify the weakest point. This calculation helps emphasize
the fact that an attacker only needs to successfully exploit one vulnerability in order
to compromise a device. The second formula serves to encompass the overall coverage
of surface attack vectors. This model is similar to unit tests in a software program
and helps gauge the number of exposed attack vectors in an IoT device. In this case,
the user of the formula can supply weights to the five different attack categories of
an IoT device in order to prioritize certain aspects of the device based on what is
important to them. The sum of all the weights must equal one. The risk scores of
the subcomponents of each category are summed, normalized, and multiplied by the
weight supplied by the user of the formula. The sum of the scores of each category
provide the user with a percentage of coverage of surface attack vectors.

1.2.3 Case Studies

We analyze three different IoT devices using the two formulas described: the Amazon
Echo, the Belkin WeMo light switch, and the Apple watch. Through a combination
of reading technical manuals, product specifications online, and reports published by
security researchers, we gather enough information about each device to accurately
score the components in each attack category. Our formulas find that the Apple
watch has the lowest risk, with the Amazon Echo a close second. On the other
hand, our formulas find that the WeMo device is high risk, which is consistent with
the recommendations of security researchers to unplug the device as the only secure
course of action [27]. Our case studies suggest that a higher amount of security
considerations in a product result in a lower risk to the device generated by our
formulas.
Chapter 2

BACKGROUND

2.1 Definitions

2.1.1 Internet of Things

The Internet of Things (IoT) is “the network of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, and connectivity which enables these things to connect, collect, and exchange data” [51]. These devices allow for direct integration of the physical world with software. An IoT device refers to ”a physical device that wouldn’t usually be expected to have an Internet connection, and that communicate with the network independently of human action” [42]. This is a key difference from a smartphone or a laptop.

2.1.2 Risk Assessment

Risk assessment is “a procedure for optimizing application security by identifying objectives and vulnerabilities, and then defining countermeasures to prevent or mitigate the effects of those threats” [45]. We will be outlining a framework for risk assessment of IoT devices in this report.

2.1.3 Zigbee

According to IEEE, “Zigbee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios, such as for home automation, medical device data collection, and other low-power low-bandwidth needs, designed for small scale projects which
need wireless connection. Hence, Zigbee is a low-power, low data rate, and close proximity (i.e., personal area) wireless ad hoc network." [3]

2.1.4 Z-Wave

Forbes describes Z-Wave as “a wireless communications protocol used primarily for home automation. It is a mesh network using low-energy radio waves to communicate from appliance to appliance.” [49]

2.2 Related Works

The concept of assessing risk in a product or application is not new. However, due to the novelty of IoT devices, it is often skipped. With the nature of IoT designed with the emphasis of being first to market, spending extra time and money on security can be seen as a negative. However, there are several resources which describe how to develop a threat model and offer a list of common threats to various applications.

2.2.1 OWASP

The Open Web Application Security Project (OWASP) is focused on improving the security of software and publishes the top ten vulnerabilities in web security annually [37]. Due to the growth of Internet of Things devices, OWASP has expanded their list to include IoT devices. Its members have created an IoT Project designed to help manufacturers, developers, and consumers better understand the security issues associated with the Internet of Things, and to enable users in any context to make better security decisions when building, deploying, or assessing IoT technologies. The project looks to define a structure for various IoT sub-projects such as attack surface areas, testing guides, and top vulnerabilities [39]. Currently, OWASP has developed
en key categories for manufacturer and developer IoT security guidance, ranging from physical security to web interfaces. OWASP includes a threat model for each category, which helps outline potential business impacts, and identifies suggestions on how to minimize the size of these attack categories. This list considers aspects such as physical security, insecure user interfaces, privacy concerns, and insecure network services.

![Figure 2.1: Example of OWASP Vulnerability Description.](image)

### 2.2.2 NIST Draft 8200

Draft 8200 is a working document by the National Institute of Standards and Technology to create a set of standards regarding the security of IoT devices [44]. It is an initial attempt by the United States government to assess the landscape of the growing IoT ecosystem and provide some guidelines for its security. The document outlines different risks to various types of industries that IoT devices are involved. It also determines whether a technological standard already exists to address a threat or whether a new one is necessary. However, the document is still in its early stages and does not provide a framework to prioritize or score risks.
2.2.3 Secure IoT Project

The Secure IoT project is a joint project between the universities of Stanford, Berkeley, and Michigan. Its goal is to analyze how sensing and analytics systems can preserve and protect user security, and whether the creation of new hardware or software systems can make securing an IoT application as easy as it is to secure a web one [6]. This project has created some proof of concept chips that provide ample space for both hardware accelerators and cryptographic engines to allow every IoT device the ability to utilize encryption, no matter how low power it is. This project focuses on creating technological solutions for IoT devices.

2.2.4 Cloud Security Alliance

The Cloud Security Alliance (CSA) IoT Working Group has published several documents providing guidance for securing individual products that make up an IoT system. The hope of the document is to provide a framework for organizations who wish to transform their existing products into IoT-enabled devices [28]. The latest document created by the CSA, titled *Future Proofing the Connected World*, is a reference guide with thirteen steps to create a secure IoT device.

2.2.5 IoT Village - DEF CON

DEF CON is one of the world’s largest hacker conventions held annually. A popular event at DEF CON is the IoT village, which hosts talks by security researchers as well as contests to find vulnerabilities in IoT devices. The IoT village has been responsible in uncovering 113 new vulnerabilities in these kinds of devices [14] and is a good resource for tracking common exploits.

Additionally, at DEF CON 23, information security specialist Daniel Miessler pro-
vided a testing methodology for IoT devices. Using the aforementioned OWASP IoT project, he outlines the complexity of the IoT device ecosystem and suggested several differences to testing approaches while developing IoT devices. Miessler explains that it is important to consider the common surface areas that IoT systems share to make sure that the developer does not miss certain items, such as ecosystem access control, device memory, update mechanisms, and third-party backend APIs [35]. We used a combination of Miessler’s work as well as trends in IoT vulnerabilities presented gathered at DEF CON in order to identify common attack vectors.

2.2.6 Microsoft

Microsoft has published their secure development lifecycle (SDL) process to the public with detailed documentation available on their website. Their SDL contains an introductory security training phase, followed by five individual phases, and concludes with a response phase that requires product development teams to act on security vulnerabilities discovered post-release [21]. It is a great resource to use for building large-scale applications, but we find it hard to see IoT developers taking the time to integrate Microsoft’s detailed process into their rapid development, especially in the case of smaller manufacturers. Therefore, we instead reference the basic guiding principles of Microsoft’s SDL to create a compressed risk threat model specifically geared towards IoT devices.

2.2.7 Threat Modeling: Designing for Security

Adam Shostack, an expert in security design analysis techniques, is the author of a book that outlines how to use threat modeling to assess software and system design processes. He provides a structured and methodical framework that can be applied to every software program. Although Shostack’s book does not directly deal with
IoT devices, we lean heavily on his method of breaking up applications into various categories and analyzing each subcomponent.
Chapter 3
SURFACE ATTACK VECTORS

The ecosystem of Internet of Things devices is difficult to capture and define. Because these devices are embedded, connect to the Internet, interact with a hub, and communicate with other IoT devices, creating a one size fits all risk assessment is difficult to do. In order to identify the different categories of risk for a single Internet-of-things device, we split up the risk factors of the IoT ecosystem into five different categories: the physical device, network traffic, mobile clients, web and cloud interfaces, and the unknown risk category. Each of these areas comes with its own threat model. All of the five categories mentioned above are interconnected, and the security failures of one can result in the security failures of a users entire IoT ecosystem. For instance, if a device does not properly verify that the cloud server it is connecting to receive an update is the server approved by its manufacturer, attackers can spoof a server and ship malware to this device which has happened before in Belkin WeMo devices [20]. This represents a single point of failure of the devices security, and even if other aspects of the device have been engineered with a high standard of security, they are now meaningless.

3.1 Device

The physical device itself has a variety of different attack vectors that can be exploited by a malicious third party. Consider what is present in the devices memory. When a user authenticates to the device, it receives a set of credentials. If these credentials are stored in plaintext, this poses a vulnerability because any access to the devices memory will now leak usernames and passwords. Although it is against best security
practices, most users will likely not put extra thought into creating an overly secure password and will probably use a common set of credentials that they use for other accounts. Therefore, storing unencrypted credentials could affect users in a scope outside of the IoT device.

In general, devices need access to servers to receive updates, report usage statistics, or perform searches on queries. For instance, the Amazon Alexa sends voice commands spoken by a user to a server that processes the command and then instructs Alexa to respond. The IoT device will need some way to access these servers. If this information is hardcoded in memory in plaintext, such as a URL, an attacker will be able to identify these servers and potentially access them. Hopefully, these servers will use some sort of authentication. If a device has server authentication credentials hardcoded in its source code, attackers will again have the ability to access these servers. Because IoT devices are usually fairly inexpensive, such as a light switch or a light bulb, an attacker does not require a high amount of financial resources in order to search for hardcoded credentials for a variety of different manufacturers.

The storage of encryption keys can compromise the security of a device. Key management is a difficult problem, and in previous cases, manufacturers have simply shipped an encryption key in the firmware of a device in order to encrypt data. In 2013, Belkin shipped a firmware signing key with its devices in order to use to verify that any update a device was receiving was actually coming from Belkin servers. Researchers were easily able to dump the firmware, extract this signing key, and then create their own programs that they could now sign and send to the device [23]. If the device is using a hardcoded key to encrypt user credentials, sensitive URLs, or other information that should be hidden from an attacker, it is almost the same as not even encrypting in the first place.

Some IoT devices require an even more stringent threat model based on their
purpose, such as IoT devices in classified government facilities. If these devices are able to be accessed by employees, such as centrifuge readings in a nuclear facility, then there are several factors that should be considered. If there is a command-line interface, can any user escalate their privilege and access any data stored on the device? Can users extract the firmware of the device or tamper with it? Are there additional checks in place to ensure that data cannot be permanently deleted on purpose or by accident? Although these questions are not as important for common household devices, they need to be considered by manufacturers creating IoT devices for more important purposes.

The model of threats to a device can be seen in Figure 3.1

![Figure 3.1: Surface Attack Vectors For Device Category.](image-url)
3.2 Network Traffic and Services

IoT devices can communicate with different types of devices and over different channels. For instance, an IoT lightbulb can communicate with other lightbulbs over a specific low-powered protocol, such as ZigBee or Z-Wave. Both of these protocols are limited to a distance range, so the lightbulb cannot communicate with other devices outside of the local ZigBee network. In order to receive updates, the lightbulb must also be connected to a hub. The hub is a more powerful device, similar to a router, that will then connect to servers to receive updates and transfer it over to the lightbulb. In order to receive instructions to turn on or off, a user can utilize a mobile application that will communicate with the hub over WiFi. The hub will then send instructions to the lightbulb over the low powered protocol, ZigBee, to tell it to change colors. In this very simple example of turning an IoT lightbulb on or off, there are many different channels over which information is transferred: lightbulb to hub, hub to mobile device, hub to servers. Each channel poses an attack vector for a malicious party to compromise a device.

3.2.1 Protocol

As quickly mentioned above, there are different protocols that IoT devices can utilize in order to communicate. Higher powered devices, such as voice assistants, can turn to WiFi. Other devices, such as light switches, may rely on low powered protocols created specifically for low powered devices. Two of the most prevalent ones are ZigBee and Z-Wave. However, ZigBee has been shown to be much riskier than WiFi and has been the source of several large IoT vulnerabilities [53]. The ZigBee protocol allows for combinations of certain features that renders the network vulnerable [46]. Since it is a newer standard, there are also many more opportunities for developers to make mistakes. Several prominent issues include insecure key transportation, where
a new device joining the ZigBee network receives its network key to encrypt traffic over the air in plaintext. There are two opportunities in the protocol that allow for reusing an initialization vector, allowing for an attack to recover the plaintext of the encrypted messages [47]. Therefore, the choice of a protocol can increase the surface attack area of an IoT device.

3.2.2 Transport Encryption

Regardless of what communication protocol is used, data should be encrypted when it is sent from one device to another. A few years ago, Belkin created an IoT baby monitor that allowed users to view the video feed of their homes. The password to view the video feed was transmitted over plaintext to a server at a hardcoded IP address, which means that any attacker able to view that traffic would have access to a users video feeds of monitors installed in their home [26]. When a user needs to authenticate to their IoT device, these credentials should be encrypted when the authentication request is created and sent. If using something such as SSL/TLS, the manufacturer of the IoT device must ensure that the implementations are properly configured and that the most current version is being used. Custom encryption protocols should never be used under any circumstances, and if using AES encryption, ECB mode should never be used either [22].

3.2.3 Ports

IoT devices, just like every other type of device, should operate with a minimal number of network ports active. If a port does not need to be used, leaving it open creates an attack vector for a malicious actor. There are many free tools, such as NMAP, which can quickly run a port scan on a device to determine which ports are open, which are closed, and what kind of service each port is attempting
to host. Additionally, devices should not make ports available to the Internet via protocols such as Universal Plug and Play (UPnP). This protocol does not require authentication and can expose devices to Internet access from outside of the bounds of the local network [48].

### 3.2.4 Updates

IoT devices should be receiving regular updates in order to enhance features, address bugs, and fix vulnerabilities. While updates are supposed to be a good thing, they also add another attack vector due to the fact that new software is being installed on the device. Updates should be encrypted when they are sent to a device, otherwise attackers will have full knowledge of how an IoT device works by viewing the source code and can craft more powerful attacks. Furthermore, it is extremely important that updates are signed and the receiving device verifies this signature. Otherwise, this presents a great opportunity for attackers to either tamper with the update, or create their own piece of malware to ship to a device. An additional point to consider is whether the device is performing SSL verification. Ideally, a device should be pinning certificates owned by its manufacturer so that when it connects to a server to receive an update, it knows that the connection is valid and trusted [50]. Otherwise, servers can be spoofed.
3.3 Mobile Client

A large appeal of IoT devices is the ability to control them remotely with only your fingertips through the use of a mobile application. For example, if I am coming home from work during winter, I can tell my IoT thermostat to begin warming up the house prior to my arrival. While this is a large convenience factor, it brings a whole suite of surface attack vectors with it.

3.3.1 Authentication

Because mobile clients can control devices inside of a user’s home, it is very important that authentication be performed properly. As with any authentication scheme, there are several details to keep in mind. Weak passwords should not be allowed. The definition of a weak password can vary from manufacturer to manufacturer, but in general, passwords shorter than 8 characters [19] or in known password lists should
not be allowed if possible. Username enumeration should be prevented by displaying general error messages if a username and password combination does not work. More specifically, rather reporting an error of either the username being invalid or the password being invalid, the application should report saying the combination together is not correct. This prevents attackers from gathering a list of known usernames associated with IoT devices. Mobile applications should also have an account lockout mechanism to prevent brute force attempts. While this can create the potential for a denial of service attack, malicious parties should not have the freedom to brute force username and password credentials for user devices.

Mobile applications should have the ability for users to enable two-factor authentication, which can be as simple as using the Touch ID or Face ID features present on Apple iPhones. In the case that an attacker is able to guess a user’s account credentials, this will allow users to add an additional layer of protection if they wish to. Mobile clients should enforce password expiration after a reasonable time period in order to ensure that users rotate their passwords. An easy way for attackers to brute force credentials and access a high number of IoT devices from an associated mobile application is if default usernames and passwords are used. If it is the case that an IoT device comes with a default username and password, the mobile application should force the user to change these credentials upon first use. A great example of this is Mirai, which assembled a botnet of over 400,000 devices to create one of the largest distributed denial of service attacks that brought down a DNS provider on the east coast by compromising IoT devices that were still using default usernames and passwords [32].
3.3.2 Communication

Mobile devices will communicate with either IoT devices directly, or will communicate through a hub that a low powered device is connected to. Any information that is sent between the device and the mobile application should be encrypted, especially when passing along user credentials with an authentication request.

3.3.3 Injection

Because there are authentication fields for users to fill out when logging into their mobile application, this adds the risk of malicious parties injecting commands or attempting to access data that they should not be accessing, such as with a SQL injection. Usernames and passwords will probably be stored in a database, so mobile applications should ensure that user input is not trusted and is sanitized prior to executing any queries.
3.4 Web Interface

Some IoT devices will come with a web application that can be used to control settings remotely, rather than being limited to a mobile application. This is often the case in industrial environments where a large number of devices need to be kept track of and modifications can be made concurrently. This creates a number of different surface attack vectors for malicious actors to experiment with.

3.4.1 Authentication

Similar to the points presented in the authentication section for mobile applications, the web interface should use proper authentication. Weak passwords, like those un-
der 8 characters, should not be used. Username enumeration should be prevented by providing general error message to invalid authentication requests, rather than specifying whether the username or password is incorrect. Web interfaces should implement an account lockout mechanism to prevent attackers from having the ability to brute force different username and password combinations without an issue. Web interfaces should force users to rotate their passwords after a certain period of time, such as six months. If the web interface has an account with default credentials associated with an IoT device, the user should be required to immediately change these credentials when using the web application for the first time. To add an additional layer of security, web interfaces should also provide users with the option to enable two factor authentication when logging in.

3.4.2 Communication

Web interfaces should be encrypting any communications associated during user interactions with their IoT device. The web interface should have the ability to use HTTPS in order to encrypt user traffic. In order to prevent unnecessary and unwanted traffic or attempted connections, the web application should have a firewall to protect any web interfaces exposed to the Internet. If the web interface uses third-party APIs or services in order to assist with accomplishing a task, data sent to these services should be encrypted if possible.

3.4.3 Processing User Data

Year after year, two of the top 10 risks of web applications assessed by the Open Web Application Security Project stem from implicitly trusting user input to an application [40]. In 2017, the most common web vulnerability was injection flaws, such as a SQL injection [40]. This occurs when untrusted data is sent as part of
a command or a query. Attackers can craft special queries to either do immediate damage, by attempting to delete or modify a database, or by attempting to gain sensitive information without proper authorization. Cross-site scripting attacks are a common vulnerability in web applications as well. These flaws occur whenever an application includes untrusted data in a new web page without proper validation, or updates an existing web page with user-supplied data that can create HTML or JavaScript. This vulnerability, typically referred to by the acronym XSS, allows attackers to execute scripts in a victim's browser that can steal user credentials from users who are logged in, deface websites, or redirect the user to a malicious site that they are not aware of.

3.4.4 Updating Software Components

Libraries, frameworks, and other modules used by a web application can run with the same privilege that the application has. If a vulnerability is discovered, web interfaces should be patched by updating the library or framework with the latest possible version that contains a security fix. In 2017, Equifax’s massive data breach that resulted in credit data and social security numbers being stolen from hundreds of millions of consumers occurred due to a vulnerability in a software module that was running Apache. The vulnerability was disclosed a few months prior to the hack, but Equifax did not take the time to update their web interface [18]. Attackers were then able to exploit the vulnerability and compromise Equifax systems. Therefore, any libraries or frameworks utilized by the web application should be updated as often as possible. If a security vulnerability is discovered, patches should be applied immediately to secure the web interface.
Figure 3.4: Surface Attack Vectors For Web Category.
Because threat modeling requires detailing many different components, it can be easy for developers to miss some security aspects under the pressure of meeting deadlines. In order to help facilitate this process, this section outlines a model that can be used to determine the risk of a device based on security considerations made by developers.

Figure 4.1: Classification of Attack Categories of an IoT Device.

4.1 Classifying Risk Scores

As discussed in detail in Chapter 3, there are five different attack categories in an IoT device. These categories will be individually assessed to determine risk for each
component, and then these risks will be combined to create an overall calculation of the risk of a device. The score will be between 0 and 1, which can be easily correlated to a percentage. A score of 0 is idealistic and indicates that there are no risks posed for a device. A score under 0.5 indicates a minor risk, since no system is fully secure, but is not a cause for much concern. A score of 0.5 indicates that there is a medium risk posed, but the attacker will require either close proximity or high skill to exploit the vulnerability. A score of 0.7 indicates a more exploitable vulnerability, but does not guarantee that an attacker will take advantage. A score of 1 indicates that the risk is maximized and the device will be, without much uncertainty, compromised. The scores will be visually represented by the colors green, yellow, orange, and red, respectively. For each attack category, the possible options for scores of individual subcomponents are detailed below.

![Figure 4.2: Visual Representation of Different Risk Scores.](image)

4.1.1 Device

The subcomponents visually portrayed in Figure 3.1 that make up the Device attack category are listed below with assigned scores.

**Memory Storage**

- Data Encrypted: Is data encrypted? Data is either encrypted or not, so we assign a score of 1 for unencrypted data to signify a guarantee of compromise.

- Encryption keys hardcoded: Are encryption keys hardcoded? If hardcoded
cryptographic keys are used, it is almost certain that attackers will be able to gain access to the original unencrypted data. OWASP rates the likelihood of this exploit as high [38], and we escalate it to be a guarantee of compromise. Therefore, we assign a score of 1.

Physical Access

- Tamper resistance: Is the device tamper resistant? Early detection of tampering will help system administrators identify compromised devices and prevent damages [15]. We consider the risk of a device without tamper resistance mechanisms to be at a medium risk, since a lack of tamper resistance does not substantially increase the risk of the device being compromised itself. Scoring is 0.5.

- Firmware extraction: Can a non-administrative user extract firmware? If any user is able to extract firmware, binaries present in the firmware can be exploited [5]. This requires time and skill, so the risk is considered to be medium and thus scored as 0.5.

- Privilege escalation: Can a user gain root privileges through a command-line interface? If so, a user can access resources and perform actions that they should not be able to, effectively bypassing all access control mechanisms present on the device. This requires skill and time, so the risk is considered to be medium. Its score is 0.5.

Figure 4.3 displays a visualization of these subcomponents colored according to their risk score.
4.1.2 Network Traffic

The subcomponents visually portrayed in Figure 3.2 that make up the Network Traffic attack category are listed below with assigned scores.

Protocol

- **ZigBee**: ZigBee has been proven to be exploitable repeatedly [53] and has a number of security risks in the standard itself [46]. Therefore, ZigBee risk is considered high and is scored as 0.7.

- **Z-Wave**: Z-Wave has been exploited multiple times in the past, but recent
versions support better encryption standards and have implemented security fixes [31]. However, the protocol is still new and has not been subject to as much attention and testing as other protocols such as Wi-Fi. We define risk is medium, and its score is 0.5.

- WiFi: WiFi has been studied and researched for many years. Although the recent Key Reinstallation Attack surfaced late last year, security updates have become available and the attack itself requires close proximity of the attacker in order to be able to execute it [52]. We consider the risk of WiFi to be low, and its score is 0.1.

*Transport Encryption*

- Updated library version: Are encryption libraries updated? Outdated libraries are filled with vulnerabilities that can be exploited. For instance, OpenSSL had 12 different documented vulnerabilities in the year 2017 alone [11]. We consider this risk to be high and score it as 0.7.

- Custom protocol: Is a custom encryption protocol being used? Custom encryption that has not been vetted by cryptographers does not work and can be easily broken by attackers. We consider this risk to be a guaranteed compromise and score it as 1.

- Re-using IV: If applicable, is the initialization vector of the cryptographic protocol being reused? If so, the encryption scheme is compromised [2]. We consider this to be a high risk and score it as 0.7.

*Ports*

- Unused ports: Are any unused ports left open? Malware can be distributed through open ports, services running on open ports can contain bugs, and ap-
Applications can be misconfigured [4]. If ports are open and unused, this is probably a mistake and there will likely be no security monitoring conducted on those ports. Leaving unused ports open greatly increases the surface area for attackers. We consider this to be a guarantee of compromise and score it as 1.

- **UPnP protocol**: Are protocols in place that expose the device to extraneous connections? Protocols such as UPnP were not designed with a priority for security and have many exploits [36]. IoT devices should not be attempting to connect to any services or devices that are not necessary for its functionality. We consider this to be a guarantee of compromise and score it as 1.

**Updates**

- **Updates encrypted**: Are updates encrypted when shipped? In 2013, WeMo devices received firmware updates over an unencrypted rich site summary (RSS) mechanism, allowing attackers to easily push firmware updates to WeMo users by spoofing the feed with correctly signed firmware [23]. We rate this risk as a guarantee of compromise and score it as 1.

- **Updates signed**: Are updates signed and verified when distributed? If not, an attacker can spoof an update and distribute malware to a device [20]. We consider this to be a guarantee of compromise and score it as 1.

- **SSL Verification**: Does the device (or its hub) perform SSL verification when connecting to a server for an update? If the device does not validate its connection, it could be spoofed by an attacker and all communications can be observed by an attacker [10]. We consider this to be a high risk and score it as 1.

Figure 4.4 displays a visualization of these subcomponents colored according to their risk score.
Figure 4.4: Risk Scoring Components of the Network Traffic Attack Category.

4.1.3 Mobile

The subcomponents visually portrayed in Figure 3.3 that make up the Mobile attack category are listed below with assigned scores.

Authentication

- Account lockout mechanism: Will too many failed attempts trigger an account lockout? There are tools such as Hydra that can perform rapid dictionary attacks to brute force passwords and compromise accounts [7]. We consider this risk to be medium and score it as 0.5.

- Weak passwords: Are weak passwords allowed? It takes about 15 minutes for a regular laptop to break an 8 character password that only consists of numbers, and about 0.001 seconds on a supercomputer [8]. In a 2010 password leak of RockYou.com, 30% of users picked a password less than six characters in length,
with the most common being “123456” [30]. We define this risk as high and score it as 0.7.

- Password in known list: Is the users password in a known password list? Special hardware can be used to quickly run through a leaked password list and guess credentials. We consider this to be a high risk and score it as 0.7.

- Two-factor authentication: Does the device enable two-factor authentication? This measure will complicate the attacker’s ability to access the device even if the password is correctly guessed, but does not add a significant level of vulnerability to the device. We consider the risk to be low and score it as 0.1.

**Communication**

- Data encrypted in transit: Is data encrypted when sent from device? Uncrypted data can easily be captured and read by attackers. We classify this risk as guarantee of compromise and score it as 1.

- Encryption libraries updated: Are encryption libraries up to date? The explanation is the same as mentioned in the Transport Encryption section under the Network Traffic risk category. We define this risk to be high and score it as 0.7.

- IV re-used: If applicable, is the IV being re-used when encrypting data? If so, the encryption scheme is compromised [2]. We consider this to be a guarantee of compromise and score it as 1.

**Injection**

- Data sanitization: Is any user data received being sanitized properly? Malicious parties can execute a command injection attack rather trivially that will force the device to run any command that is entered [12]. We consider this risk to be high and score it as 0.7.
Figure 4.5 displays a visualization of these subcomponents colored according to their risk score.

![Risk Scoring Components of the Mobile Attack Category](image)

**Figure 4.5: Risk Scoring Components of the Mobile Attack Category.**

### 4.1.4 Web

The subcomponents visually portrayed in Figure 3.4 that make up the Web attack category are listed below with assigned scores.

**Authentication**

- Weak passwords allowed: Are weak passwords allowed? As mentioned in the Authentication section of the Mobile risk category, weak passwords can be brute forced in 15 minutes by a regular laptop. We define this risk as high and score it as 0.7.
• Default credentials allowed: Are default credentials allowed to be used? Many devices will ship with default credentials to be used by the user that remain unchanged. This can lead to a botnet such as Mirai, which exploited default credentials in devices to take over 400,000 devices and generate one of the largest Distributed Denial of Service attacks ever seen [41]. We define this risk to be a guarantee of compromise and score it as 1.

• Password rotation: Is password rotation required? Changing passwords helps prevent brute force attacks and refreshes credentials if a device’s password has been guessed. We consider the risk to be low and score it as 0.1.

• Username enumeration: Is username enumeration prevented? Username enumeration allows the attacker to validate their guesses of usernames and/or passwords [9]. We define this risk to be high and score it as 0.7.

• Two-factor: Does the device enable two-factor authentication? As previously mentioned, this measure will complicate the attacker’s ability to access the device even if the password is correctly guessed, but does not add a significant level of vulnerability to the device. We consider the risk to be low and score it as 0.1.

Communication

• Firewall enabled: Is there a firewall enabled for the web application? Firewalls help prevent malicious connections and access by unauthorized users. We define this risk to be medium and score it as 0.5.

• HTTPS used: Is HTTPS used? If not, traffic will not be encrypted. Information sent from web application to device, such as credentials, sensitive information, or commands can be intercepted and tampered with by attackers. We categorize this as a guarantee of compromise and score it as 1.
Processing User Data

• Injection attacks: Is data sanitized to prevent injection attacks? Malicious parties can execute a command injection attack rather trivially that will force the device to run any command that is entered [12]. We consider this risk to be high and score it as 0.7.

• XSS attacks: Is data sanitized to prevent XSS attacks? XSS attacks are consistently one of the top attacks conducted against web applications and can be highly successful [39]. We consider this risk to be high and score it as 0.7.

• XSRF attacks: Is data sanitized to prevent XSRF attacks? XSRF attacks are also one of the top ten attacks executed against web applications and can cause serious damage by executing commands when a user is already authenticated [13]. We consider this risk to be high and score it as 0.7.

External Software Components

• Libraries updated: Are any external or third-party libraries running the latest version? In March of 2017, a vulnerability was discovered in the Apache Struts web-application software. In May of 2017, Equifax leaked data of 143 million Americans due to their unpatched enterprise system that utilized Apache Struts [18]. It only took two months of the library not being patched for attackers to exploit one of the largest credit bureaus in the United States due to an unpatched bug in a third-party library. Based on this, we rate this risk as high and assign it a score of 0.7.

• Update Mechanism: Is there a mechanism to check for updates? If there is no mechanism to check for updates of external components outside of manually checking, then eventually libraries will become outdated and devices exploited. We rate this risk as medium and score it as 0.5.
Figure 4.6 displays a visualization of these subcomponents colored according to their risk score.

![Risk Scoring Components of the Web Attack Category.](image)

**Figure 4.6: Risk Scoring Components of the Web Attack Category.**

### 4.1.5 Unknown

We assign the unknown subtree as a completely vulnerable attack vector. We capture the idea that the less the company/consumer knows about the device that they are using, the less they should trust the device. This has two different perspectives, where the score for the company that is designing the IoT device in question should be set to a negligible weight, whereas a consumer could weigh this subtree much higher if they have not done significant research on how the device in question functions.
4.2 Formula

4.2.1 Calculating Risk

This formula models the risk of a device. It places an emphasis on the concept of a single point of failure, meaning that if any single attack vector of the device is exploitable, then the risk of the entire device hinges on the risk of that attack vector being compromised. In other words, a device is only as secure as its weakest link.

As noted in the previous sections, “risk scores” are assigned to each component in the four attack categories. We iterate through the risk scores assigned to the individual components and derive an overall risk for the device based on these scores.

Let $r_{ij}$ denote the risk score of the $j$th component of the $i$th attack category. As an example, this could be the component representing whether updates sent to the device are encrypted or not, which is present in the Network Traffic attack category. Then, if $R$ is the highest risk from the assessed attack categories, we have

$$R = \max\{r_{ij} : 1 \leq i \leq n, j \in J_i\}$$

where $J_i$ is some set that indexes the components in the $i$th attack category. Therefore, the risk to the entire device is the largest risk score assigned to any of the components of the device. Since each $r_{ij} \in [0, 1]$, $R$ will be a value in $[0, 1]$ as well, with scores closer to 1 representing a higher risk that the device will be exploited by attackers. Thus, $R$ represents a model for a single-point-of-failure viewpoint, where a large potential for compromise in one subcomponent represents a proportionately large risk for compromise of the entire device.

To complete the formula, we assign a constant $U$ to model the risk of an unknown attack vector. Therefore, total risk to the device $T$ becomes
\[ T = \min\{1, R + U\} \]

where the total risk is capped at 1.

### 4.2.2 Calculating Coverage

When developers are creating a new IoT device, it can also be helpful to know how many different surface attack vectors that exist in their product. When developers write tests for their program, there are tools that can quantify code coverage for these tests to ensure that every portion of the program has been validated by a formal test case. We would like to base this formula off of the same idea and quantify the overall percentage of attack vectors that are present in a device.

In order to add flexibility to this formula, we also want to provide developers or project managers with the ability to weight different aspects of the device. Since different software teams will work on different parts of the product, we want to ensure that they are able to emphasize the security coverage specifically of the portion that they are working on. Additionally, some attack categories may not be relevant to some devices. If an IoT device does not have web application associated with it, then risk from that attack category should be canceled out. Therefore, our formula takes in a weight as a parameter for each risk category, where the sum of the total weights are equal to one.

For our second formula, let \( r_i \) denote the risk for each attack category, which is determined by normalizing the sum of each of the subcomponents \( c_i \). \( S \) is the total possible score of all subcomponents in an attack category.

\[ r_i = \frac{1}{S} \sum_{i=1}^{n} c_i \]
Let $w_i$ denote the weight for each attack category. In order to ensure that our end risk is between 0 and 1, the sum of all the weights must be equal to one.

$$1 = \sum_{i=1}^{n} w_i$$

We define our attack vector coverage $C$ as a sum of the risk from each of the trees subtracted from 1.

$$C = 1 - \sum_{i=1}^{n} r_i w_i$$

where $r_i$ is the risk from each of the component trees, normalized so that $|r_i| \leq 1$ and $w_i$ is the weight that is associated with each attack category. Since the total sum of all the weights is 1, $C \in [0, 1]$.

This formula considers the surface attack coverage of each of the individual attack categories. It uses the risk scores to determine whether a developer has eliminated an attack vector or not. Higher risk scores correlate to having a higher number of vulnerabilities for an attacker to exploit. Combining the coverage from each of the four categories results in creating an overall attack vector coverage score for the entire device. Since the final score is normalized to be between 0 and 1, this translates directly into a percentage that developers can easily consult to determine how many security gaps are missing in their implementations.

We would like to point out that although the two formulas are related and seem similar, there is a distinct difference between them. The first formula calculates the immediate risk posed to a device. The second formula seeks to encompass all vulnerabilities present in a device as a measure of overall security.
In this section, we present three different case studies on a variety of IoT devices in order to obtain tangible risk scores by using the model that we have discussed in detail in Chapter 4. In order to be able to provide values to the two risk formulas, it was necessary to collect background information for each device. In each case study, we will first show the calculations of the risk formula to determine the risk score of the device. This will be followed by calculations to show the overall coverage of security mechanisms present to protect from surface attack vectors.

5.1 Amazon Echo

The Amazon Echo is a voice-enabled assistant released in 2014 that is intended for user in home or office. It is geared towards the consumer market and has sold millions of units around the world. It is one of the most popular voice assistants on the market, so we believed it was a good candidate to identify risk. Below, we will assign weights to and determine the risk scores of each category:

- Physical Device (10%): The Echo does not have a command-line interface and is designed to accept voice commands. Data is encrypted when it is stored on the device. Firmware extraction and tampering, if possible, will require a high level of skill, so we will not consider this to be a risk factor. The highest risk score for each attack vector in this category is 0.

- Network Traffic (40%): The Echo uses WiFi for its communications. Data is encrypted and validated SSL connections are used for most communications - SSL requests failed when a proxy with a fake certificate authority was used [29].
However, some firmware updates were sent over HTTP and in plaintext. It is unknown whether they are signed or not, but exposing firmware updates in the open receives a score of 0.5.

- Mobile Application (20%): We will assume that the user sets a password of 10 characters that is not in a well-known password dump online. Research suggests that the accompanying mobile application will prompt the user to reset their password if too many incorrect attempts are submitted [29]. Communication with the device occurs over WiFi and is encrypted. There is no two-factor authentication for the application after initial setup, so we assign a score of 0.1.

- Web Interface (20%): An echo account must be created by the user and does not have default credentials. Password rotation is not required, so this receives a score of 0.1. Two-factor authentication is not an option, so this also receives a score of 0.1.

- Unknown Risk (10%): We model the risk of an unknown attack vector as a static constant, 1, to signify that it exists.

Using this scoring, we can now perform our calculations. We have our first formula, which assesses the total risk of the device:

\[ R = \max\{r_{ij} : 1 \leq i \leq n, j \in J_i\} \]

\[ T = \max\{1, R + U\} \]

The highest risk component of the Echo was the transmission of potentially unsigned firmware updates in cleartext, which earned a risk score of 0.5. We add our static unknown risk constant, 0.1, which results in a risk factor of 60%. Based on our formula, this is the chance a given Echo will be compromised.
We now use our second formula to assess coverage:

\[ C = 1 - \sum_{i=1}^{n} r_i w_i \]

where \( r_i \) is the sum of each attack category and \( w_i \) is the assigned weight.

- Physical Device (10%): \( r_{\text{device}} = \frac{1}{3.5} \cdot [0 + 0] \cdot 0.1 = 0 \)
- Network Traffic (40%): \( r_{\text{network}} = \frac{1}{7.6} \cdot [0.5 + 0.5] \cdot 0.4 = 0.06 \)
- Mobile Application (20%): \( r_{\text{mobile}} = \frac{1}{5.8} \cdot [0 + 0.1] \cdot 0.2 = 0.004 \)
- Web Interface (20%): \( r_{\text{web}} = \frac{1}{7.4} \cdot [0.1 + 0] \cdot 0.2 = 0.01 \)
- Unknown (10%): \( r_{\text{unknown}} = 0.1 \)

\[
0.06 + 0.004 + 0.01 + 0.1 = 0.174 \\
(1 - 0.174) \cdot 100 = 82.6\%
\]

Thus, the surface attack coverage of the Echo Dot is 82.6%. We would like to highlight the difference between the two calculated numbers. Although the coverage of surface attack vectors is high, and security measures have been considered during the development of the product, risk still remains at 60%. The attacker only needs to find one vulnerability to exploit, and since plaintext traffic is fairly easy to observe, the calculated score captures this risk.

5.2 Belkin WeMo Switch

The Belkin WeMo switch is a smart outlet that can be used to measure the power usage of devices. It is controllable through a mobile application. These devices have been through several rounds of updates due to significant vulnerability disclosures
through the years, starting in 2014. We have chosen to look at the 2014 model as it is a great example of the public-relations nightmare that can occur if a device is designed without security. At one point, security researchers published a statement claiming that the only way for customers to secure a Belkin WeMo device was to completely unplug it [20].

- Physical Device (0%): We were unable to determine whether data stored on the device is encrypted. This is one of the challenges we consistently experienced while researching different devices and their implementations. Therefore, this section will not be scored.

- Network Traffic (60%): The Belkin WeMo switch communicates using ZigBee, which earns its protocol component a risk score of 0.7. The device previously shipped with a hard-coded key to communicate with Belkin information servers [23], resulting in a trivial way for an attacker to get the key and decrypt traffic. Updates are unsigned and unencrypted, and there is no SSL validation used by the device when communicating with an update server [20]. This earns the update component a risk score of 1.

- Mobile Application (30%): The mobile application does not provide multi-factor authentication, so we assign it a risk score of 0.1.

- Web Interface (0%): We were unable to research the risk of the web interface.

- Unknown (10%): We model the risk of an unknown attack vector as a static constant, 1, to signify that it exists.

Using this scoring, we can now perform our calculations. We have our first formula, which assesses the risk of the device:

\[
R = \max\{r_{ij} : 1 \leq i \leq n, j \in J_i\}
\]
The component with the highest risk score is the update mechanism under the network traffic attack category, which received a score of 1. Thus, $R$ is 1.

$$T = \min\{1, R + U\}$$

Since the risk score is capped at 1, total risk $T$ is also 1. The quantity and severity of vulnerabilities disclosed by security researchers agrees with this risk assessment. Again, consumers were urged to disconnect this device completely due to its exploitability.

We now use our second formula to assess coverage:

$$C = 1 - \sum_{i=1}^{n} r_i w_i$$

where $r_i$ is the sum of each attack category and $w_i$ is the assigned weight.

- Physical Device (0\%): Not Analyzed
- Network Traffic (60\%): $r_{\text{network}} = (1/7.6 \cdot [0.7 + 0.5 + 1 + 0.7]) \cdot 0.6 = 0.23$
- Mobile Application (30\%): $r_{\text{mobile}} = (1/5.8 \cdot [0 + 0.1]) \cdot 0.3 = 0.01$
- Web Interface (0\%): Not Analyzed
- Unknown (10\%): $r_{\text{unknown}} = 0.1$

$$0.23 + 0.01 + 0.1 = 0.34$$

$$1 - 0.34 \cdot 100 = 66\%$$

The surface attack coverage of the attack categories we analyzed is 66\%.
5.3 Apple Smart Watch

The Apple Watch is one of the most popular smart watches on the market and commands 61% of the market share [1]. Smart watches are very interesting devices because they are not expected to be fully independent and standalone, but they can still accomplish a large amount of different tasks. The Apple Smart Watch processes and interacts with a vast amount of personal data, including messages and health statistics.

Gathering data about the Apple Watch was a more straightforward process than researching the previous devices discussed due to Apple’s iOS white papers that are available for the public to read. They describe, in detail, security mechanisms such as encryption protocols, key exchanges, and hardware secrets. This helped increase the accuracy of the case study to the availability of the required information.

• Physical Device (30%): The Apple Watch encrypts all data at rest and uses symmetric encryption to safely store all user data on the device. It offers the creation of a passcode in order to be unlocked, preventing attackers from tampering with it. Keys used for communications between the watch and iPhone are secured using class-based protection.

• Network Traffic (40%): The Apple Watch uses Bluetooth to communicate with a paired iPhone and defaults to WiFi if the iPhone is out of range. According to Apple’s iOS 12 white paper, Bluetooth addresses are rotated at a 15 minute interval to limit the risk of network traffic being compromised [17]. All communications between Apple Watch and iPhone are encrypted using a special service called Apple Identity Service, with the encrypted Bluetooth, Wi-Fi, and Cellular links providing a secondary encryption layer. We will model WiFi as the primary mode of communication for the watch, and this warrants a risk
score of 0.1.

- Mobile Application (20%): No issues were found with the mobile application.

- Web Interface (0%): A web interface is not applicable to this device.

- Unknown (10%): We model the risk of an unknown attack vector as a static constant, 1, to signify that it exists.

Using this scoring, we can now perform our calculations. We have our first formula, which assesses the risk of the device:

\[ R = \max\{r_{ij} : 1 \leq i \leq n, j \in J_i\} \]

The component with the highest risk score was a tie between a lack of multi-factor authentication of the mobile application and the use of WiFi as a communication protocol, which both received a score of 0.1.

\[ T = \min\{1, R + U\} \]

\[ T = \min\{1, (0.1 + 0.1)\} \]

Therefore, total risk \(T\) becomes 0.2. The security considerations involved in designing the Apple Watch clearly have a tangible benefit by substantially decreasing the risk of the device.

We now use our second formula to assess coverage:

\[ C = 1 - \sum_{i=1}^{n} r_i w_i \]

where \(r_i\) is the sum of each attack category and \(w_i\) is the assigned weight.

- Physical Device (30%): \(r_{device} = 0\)
• Network Traffic (40%): \( r_{\text{network}} = (1/7.6 \times [0.1]) \times 0.4 = 0.01 \)

• Mobile Application (20%): \( r_{\text{mobile}} = (1/5.8 \times [0.1]) \times 0.2 = 0.01 \)

• Web Interface (0%): Not Analyzed

• Unknown (10%): \( r_{\text{unknown}} = 0.1 \)

\[
\frac{0.01 + 0.01 + 0.1}{1 - 0.12} \times 100 = 88\%
\]

The surface attack coverage of the attack categories we analyzed is 88%.

<table>
<thead>
<tr>
<th>Table 5.1: Comparison of Calculated Risk Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Formula Used</td>
</tr>
<tr>
<td>Risk</td>
</tr>
<tr>
<td>Coverage</td>
</tr>
</tbody>
</table>

5.4 Challenges

5.4.1 Acquiring Data

The attack categories that we have proposed contain a significant number of different components that help encompass the surface attack vectors that a hacker may attempt to exploit. Even though our model is detailed, there are undoubtedly additional attack vectors that we have not considered. In order to use our outlined risk scoring model, it is necessary to obtain all information about a device to be able to fill in risk scores for different attack vectors. In some cases, such as the Apple Watch, the manufacturer has published detailed specifications that help us understand the
security mechanisms in place. In others, such as the Belkin Switch, we sourced our data from product specifications or from vulnerability disclosures provided by security researchers. However, this information is not comprehensive and results in potential gaps in our model. To provide a more detailed example, in all three of our case studies, we were unable to determine whether the encryption protocols in place were implemented correctly. For instance, when the AES cipher is used for symmetric encryption, a costly mistake is reusing the same initialization vector when encrypting data. This can lead to leaking information and is a vulnerability, even though encryption is supposedly used to protect data. Acquiring better data will lead to acquiring better results.

5.4.2 Assigning Risk Scores

In Chapter 2, Assessing Risk, we laid out a framework to classify vulnerabilities as ranking between 0 and 1, where 0 indicates that there is no risk, and 1 guarantees that the device will be compromised. We tried to take into consideration the skill required to perform an attack and the severity of the damage caused when assigning scores as low, medium, or high-risk, but this is not a foolproof method and functions as a general hypothesis when classifying the risk of different vulnerabilities. However, this kind of data is more qualitative than quantitative. Since we cannot guarantee exactly how a vulnerability can or will be exploited, it becomes difficult to assess the potential damage that can be caused and assign a corresponding risk score. In an attempt to gather more concrete data, we used historical examples such as the Mirai botnet to try to gauge how vulnerabilities were exploited in the past. In the case of Mirai, default credentials led to a massive botnet being created that caused a successful distributed denial of service attack on DNS services, resulting in portions of the Internet being unreachable [33]. However, we were unable to match each vulnerability with a real, in the wild exploit.
6.1 Expanding Classes of IoT Devices

Although this report took into consideration industrial IoT devices, a majority of the research was focused on detailing the risks of consumer IoT devices, such as smart speakers and smart watches. Information about these devices was more readily available, and the devices themselves were cheaper to access. We suggest expanding the work of this report to include IoT devices specifically in medical, industrial, and critical infrastructure applications. The consequences of a smart speaker being compromised are vastly different than of an internet connected pacemaker being hacked, and we pose the question of whether the risk model would need to be altered to account for the severe impact that comes with compromised medical devices. Autonomous vehicles are also an interesting industry to continue exploring.

6.2 Interactive Tool

In order to allow users to quickly interface with our risk model and receive an assessment of an IoT device, we propose the creation of an interactive tool. The tool will calculate the risk of different categories of an IoT device based on input from the user. We also propose that the tool come with a set of default values for different classes of IoT devices, such as the typical risk of a smart speaker versus that of a light bulb based on common technologies and implementations used specifically in those kinds of devices.
6.3 Quantifying Business Impact

We propose conducting future case studies using the risk scores generated by our model and correlating those scores with devices that have been compromised in the past in order to provide a tangible business impact associated with each risk score. For instance, if a smart speaker has a 70% chance of being exploited, and its overall attack vector coverage is 30%, we would like to correlate that with a negative cost towards the business based on empirical evidence. This work could take into account factors such as market share.
Although there have been attempts at outlining security guidelines for the IoT ecosystem, it remains mostly unsecured. The sheer quantity of these devices in our lives is overwhelming, and it will continue to grow as more industries begin to incorporate smart devices [24]. As recent years have shown us, cyberattacks on these devices can have devastating effects. In 2010, the Stuxnet virus caused real world, physical damage by destroying nuclear centrifuges after hackers took control of them [25]. The Mirai botnet, which had almost half a million devices under its command, launched the most powerful distributed denial of service attack of its time and caused portions of the Internet to stop functioning [33]. In order to decrease the likelihood and ease with which these events occur, securing these devices needs to become as important as producing them.

In our report, we use a combination of empirical evidence and related research in order to identify the most common attack vectors in IoT devices. This has been done before by other groups, but we take it a step further by also providing two different formulas for evaluating the risk of a device. The first formula calculates the chance of the device being exploited based on security design and implementations and emphasizes the fact that a device is only as strong as its weakest point. The second formula assesses overall coverage of attack vectors in an IoT device and can be a useful tool to developers and project managers to quantify how much work needs to be done in order to increase the overall security of a device. In order to facilitate this process, we create risk scores that are associated with each attack vector. These scores range from zero to one on a numerical scale, where one signifies a guarantee of compromise. The calculation of each risk score takes into account the number of times
each attack vector had been exploited in the wild, the impact it had, and the rough level of skill required to accomplish compromising a device through a given attack vector. Additionally, to add flexibility to the calculations, we include a parameter for weighting each of the attack categories that we define in this paper.

We hope that this work helps developers understand the different risks to IoT devices and the impact each attack vector can have. We also hope that our work can be expanded upon in the future to develop into an automated tool that can be used to help assess the risk of a device based on security design and implementations.
BIBLIOGRAPHY


[9] Command Injection - OWASP.

[10] Multiple vulnerabilities found in connected IoT home security device.

[12] Command Injection - OWASP.


[41] S. Ragan. Here are the 61 passwords that powered the Mirai IoT botnet. CSO Online.

[42] S. Ranger. What is the IoT? Everything you need to know about the Internet of Things right now. ZDNet.


