IoT

MDPI

Article

IoT Network Security: Threats, Risks, and a Data-Driven Defense Framework

Charles Wheelus * and Xingquan Zhu

Department of Computer and Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL 33431, USA; xzhu3@fau.edu

* Correspondence: cwheelus@my.fau.edu

Received: 25 August 2020; Accepted: 13 October 2020; Published: 19 October 2020

Abstract: The recent surge in Internet of Things (IoT) deployment has increased the pace of integration and extended the reach of the Internet from computers, tablets and phones to a myriad of devices in our physical world. Driven by the IoT, with each passing day, the Internet becomes more integrated with everyday life. While IoT devices provide endless new capabilities and make life more convenient, they also vastly increase the opportunity for nefarious individuals, criminal organizations and even state actors to spy on, and interfere with, unsuspecting users of IoT systems. As this looming crisis continues to grow, calls for data science approaches to address these problems have increased, and current research shows that predictive models trained with machine learning algorithms hold great potential to mitigate some of these issues. In this paper, we first carry out an analytics approach to review security risks associated with IoT systems, and then propose a machine learning-based solution to characterize and detect IoT attacks. We use a real-world IoT system with secured gate access as a platform, and introduce the IoT system in detail, including features to capture security threats/attacks to the system. By using data collected from a nine month period as our testbed, we evaluate the efficacy of predictive models trained by means of machine learning, and propose design principles and a loose framework for implementing secure IoT systems.

Keywords: internet of things; network security; threats; machine learning; security analytics

1. Introduction

As both cloud and on-premise information services have become fundamental components of modern life, a new class of services, Internet of Things (IoT), is coming to fruition, increasing our dependence on networked technology to a far greater extent than might have been imaginable just a decade ago. IoT services enable communication between everyday devices such as household appliances, consumer devices, industrial controls, sensors, and just about any device imaginable. Interacting with more traditional Internet connected devices like servers and routers, these IoT systems and services provide a greater degree of automation and functionality than was previously possible. While still in its infancy, the IoT market is expected to reach $520 billion by 2021, up from $235 billion in 2017 [1]. Other estimates expect to exceed 1.5 trillion dollars by 2025 [2].

While IoT products and services clearly promise to provide great advantages, it should be clear that the security risks that come with Internet connectivity are a cause for concern. It is commonly known that many threats exist on the Internet, and extending this connectivity to everyday devices expands the reach of such threats as well [3]. Furthermore, above and beyond exposure to the nefarious activities of bad actors, the possibility that IoT services may become ineffectual or inoperable under poor Internet conditions, whether resulting from network segment failures or other technical difficulties, presents additional concerns for those that may depend on IoT products and services. During 2017 alone, attacks against IoT devices increased 600 percent [4], highlighting both the increased

interest by attackers in compromising IoT systems as well as the need for increased efforts in securing IoT systems.

Intuitively, Internet security is not a new topic. For longer than the Internet has been commercially available, security incidents have been widely reported. In 1971, Bob Thomas is credited with writing the first computer worm [5]. It was a non malicious proof of concept whose methodology would bring down the Internet over a decade later. In 1989, Robert Morris, a Cornell graduate student at the time, released his computer worm which, according to Morris, was an experiment gone awry, making the Internet largely inoperable for days [6]. Since this time, countless attacks have occurred on the Internet. Unfortunately as we become increasingly dependent on the Internet in our daily lives, these attacks continue to pose an ever-growing threat to the reliability and fidelity of the services that operate on the Internet. The advent of IoT raises the bar in terms threat potential. Not only is commerce and critical infrastructure a target, but increasingly, so are everyday items such as appliances, sensors, door locks, video cameras, and a whole host of devices that are now being connected to the Internet. While the security problems of the early Internet may have been problematic, in a sense, they were isolated to the network. Today, these challenges have a much greater real world presence than in the past.

Though estimates vary widely [2,7], the number of IoT devices connected to the Internet is conservatively projected to increase to approximately 25 billion by 2025 [2]. It is worth noting that current IoT devices already outnumber all human Internet connections, such as computers and mobile phones [7]. This trend is expected to result in an IoT market size of over 2 trillion dollars by 2023. This, however, also implies a significantly greater opportunity for those with ill intent that seek to exploit this burgeoning market. The reasons for these inherent vulnerabilities range from carelessness by vendors rushing to get products to the marketplace, to sophisticated hacking schemes that are designed to thwart traditional security methods. Computing power has enabled ever more effective brute force attack types, and the propensity of criminal enterprise to commandeer network connected computing resources has spawned a new generation of botnet armies with a wide ranging set of options for creating ill gotten wealth. At the same time, security concerns are usually an afterthought. These factors are a serious cause for concern to the security personnel tasked with protecting the digital infrastructure upon which we depend.

IoT Attack Examples: Of the many high-profile IoT attacks that have been documented, probably the most famous is known as Stuxnet [8]. Believed to have been created by United States and/or Israeli intelligence, in 2010, Stuxnet succeeded at severely damaging equipment used by Iran for enriching nuclear material. In 2015, a pair of hackers demonstrated the ability to remotely hack the control systems in a Jeep Cherokee [9], resulting in a massive recall to patch the vulnerability that they pointed out to the industry. In late 2016, the founders of a distributed denial of service (DDoS) protection company released Mirai, a virus that targets a specific brand of microprocessors and exploits default credentials [10]. In 2017, the FDA issued a voluntary recall of just under half a million pacemakers due to concerns about vulnerabilities that existed in the devices that could allow the devices to be adjusted by hackers [11]. These are just a few examples of IoT security issues that have been widely reported.

Network and cyber security techniques and methodologies have been developed and utilized for some time. Not only are IoT systems vulnerable to most if not all of the existing manner of threats, but also that they pose new security concerns due to several factors. Here, we briefly summarize three main challenges for IoT systems:

- Limited Device Capability: IoT devices and systems have entered areas that have traditionally been the domain of physical control devices. Such devices are often required to be simple and efficient for dedicated functionalities. As a result, they are designed/equipped/deployed with limited computing and networking capability. Converting these to IoT systems requires significant thought, planning and design, but the rush to market can short circuit this process and imposes severe security risks to the systems.

- Gigantic Scale and Volume: The sheer scale of IoT deployments creates very tempting attack targets for cyber criminals. Discovering and exploiting a vulnerability can quickly create a massive army of attackers with which to perpetrate further attacks.
- Vulnerable Environments: IoT devices tend to be placed in unprotected environments easier for attacks to access, comparing to firewall-protected networks. Perhaps most concerning is that low-cost devices are less likely to be patched and maintained in the same manner as traditional physical devices might be, creating an economic disincentive to maintain the software that operates IoT devices.

In light of these concerns, considerable thought and effort has been expended to better understand and define the challenges posed by this emerging paradigm, with the hopes that these efforts will result in a more standardized way of considering and addressing the issues that are presented by IoT. This laudable goal may prove to be challenging given the wide variety of IoT-enabled devices and systems that continue to proliferate rapidly. This challenge is exacerbated by our increased reliance upon these IoT systems and the threats posed by the aforementioned factors. Given this, it is clear that security deployment for IoT must be given careful consideration.

In this paper, we propose to analyze IoT network security and design a data-driven defense framework for IoT systems. Our essential goal is twofold: (1) provide practical insight into IoT network threats and risks, so researchers and practitioners can understand the commonalities and differences of IoT network security in comparison to general network security; (2) design a data-driven reference framework for the detection of attacks and security breaches of real-world IoT systems. For the former, we review IoT system and network characteristics and summarize IoT threats and risks. For the latter, we propose a generalized principles for implementing, deploying, and managing IoT services, and a data-driven framework for implementing IoT systems. We evaluate the network traffic collected from an IoT based company that provides a smartphone enabled secure access solution for commercial buildings, gated communities, parking garages, storage facilities and other related secure access services. We captured and analyzed 100 gigabytes of raw packet data and our findings are presented herein.

The main contributions of this paper include the following four aspects:

- We review state-of-the-art IoT design and deployments and summarize the IoT network threats and risks. We also discuss the security characteristics of the case study that could be improved upon by revising some of their features.
- Based upon our review of the IoT landscape in general, and specifically from our evaluation of the case study, we propose an IoT framework for implementing secure IoT systems.
- The design features in the case study are evaluated and validated by means of predictive machine learning models.
- The datasets used for validation are publicly released (in Supplementary Materials) concurrently with the publication of this paper and are likely a welcome addition to the relatively few security datasets that are freely available.

The remainder of the paper is structured as follows. Section 2 reviews and summarizes IoT network threats and risks. In Section 3, we propose design principles and a data-driven reference framework to detect attacks and security breaches of real-world IoT systems, followed by a case study reported in Section 4. We conclude the paper and discuss future works in Section 5.

2. IoT Network Threats and Risks

In this section, we first review the IoT reference model and octave, which provide a complete picture of the IoT eco-system. After that, we outline the challenges of IoT system security, and summarize the risks and attacks in IoT systems, with respect to the IoT reference models.

2.1. IoT Reference Model, Octave, and Common Pitfalls

2.1.1. The Cisco Seven-Layer IoT Reference Model

In 2014, Cisco Systems, a leading manufacturer of network equipment, proposed a seven-layer reference model to define IoT deployments and their components [12]. While earlier models were proposed, [13,14], the model proposed by Cisco appears to be the most complete and would seem to allow for a broader set of use cases, so we will use this model for the evaluation of our case study.

Cisco's IoT reference model, as seen in Table 1, begins with layer 1, known as "Edge" which is comprised of physical devices and controllers. Layer 2, known as "Connectivity", is the sum of all hardware and protocols that comprise all of the network communications that occur in the IoT system. These include all communications with level 1 devices, switching and routing, protocols and translations between protocols, network level security, and everything else comprising the communication and assuring the reliability of the network.

Table 1. Cisco seven-layer Internet of Things (IoT) reference model.

Layer	Definition
1 Edge	Physical devices like sensors and controllers
2 Communication	Inter-layer and intra-layer communication
3 Edge Computing	Data processing near edge
4 Data Accumulation	Data in motion to data at rest
5 Data Abstraction	normalization and data stores
6 Applications	Information interpretation and coordination between layers
7 Users and Centers	User interaction with the system

Layer 3, known as "Edge" or "Fog" computing, includes various functions related to data handling and network security such as data encryption and other means by which security is achieved from end to end. In this layer, other data processing functions may occur as needed, including filtering and scrubbing of data, and possibly event generation. Layer 4, known as "Data Accumulation" or "Storage" is the layer where data in motion becomes data at rest by persisting in long term storage. This may include transforming some of the network data into records in a database, allowing for data query, and various accumulation and filtering strategies for optimizing the data for future use.

Layer 5, known as "Data Abstraction" is concerned with activity that assures quality and completeness of data, for the purpose of use with applications in Layer 6. This may include a variety of ETL (Extract, Transform, Load) functions, and may also include various manners of data processing, comparison, reconciliation, and other various types of data manipulation. Layer 6, known as "Application", is where data interpretation occurs. In this layer, software interacts with data at rest in layer 5, and therefore does not need to operate at network speeds. Common functions such as reporting, analytics, and system control implemented in this layer. It is worth noting that each industry and use case will likely have considerable differences between each other in this layer. Layer 7, known as "Collaboration" and "Processes" is the layer where user interaction is coordinated with all other aspects of the system. From here on, this model will simply be referred to as the "seven-layer model".

Researchers have used the Cisco reference model in evaluating IoT security issues. In 2018, Bakhshi et al. surveyed numerous publications referencing differing reference models and focused mainly on data ingestion and abstraction [15]. Peter Aufner reviews various threat frameworks in light of the Cisco model and concludes that gaps exist between the frameworks, IoT and security research [16].

2.1.2. The IAS Octave

In addition to the seven-layer model, many have also evaluated IoT threats in terms of the Information Assurance and Security (IAS) Octave. For many years the, Confidentiality,

Integrity, Availability (CIA) Triad has been a prevalent concept when evaluating different manners of cyber-security policies and implementations [17]. These three factors are critical when considering the security posture of any information system. The ability to control the confidentiality, or who has access to the data, is of primary concern when considering data and system security. Integrity is concerned with the assurance that the system data remains complete, accurate, and has not been manipulated. The concept of availability assures that all authorized users are able to interact with the data when needed.

In 2013 Cherdantseva and Hilton proposed and extension of the CIA Triad, later known as the IAS Octave [18]. The IAS Octave adds the concepts of accountability, auditability, trustworthiness, non-repudiation, and privacy. Accountability is the concept of having the ability to hold users of the system accountable for their actions. Auditability is the ability of the system to monitor all actions of the system and report on them as necessary. Trustworthiness is the ability to verify the identity and establish trust in a third party. Non-repudiation is the ability of the system to confirm or deny the occurrence of an action within the system. Privacy is the ability to manage the data in such a way that privacy policies are maintained throughout the system, and possibly also to allow users to manage their own personal data.

In 2019, Mahmood et al. considered a variety of attacks based upon the eight facets of the IAS octave, and determined that most of these attacks are reasonably addressed by means trusted secure certificates [19], proposing a trusted certificate authority which they refer to as a "fog" node. Akram et al. consider many dozen manners of IoT attacks in light of these attacks' goals in relation to the IAS Octave [20] and conclude that, generally speaking, the industry lacks any standard methodology of identifying IoT attacks, and that the lack of adequate countermeasures is a serious threat to the IoT paradigm.

2.1.3. Common IoT Pitfalls

Every other year, the Open Web Application Security Project (OWASP) publishes the "IoT Top 10", a of the list the ten most common pitfalls in IoT system design [21]. The project team that publishes this list is composed of industry veterans from numerous disciplines and considers inputs from the larger design and manufacturing community when compiling the list. While the list has ten separate categories, they generally fall under one of the following:

- Weak Measurements of IoT Components and Environment: The lack of rigorous effort to secure one or more of the layers in the reference model is one of the top concerns of IoT vulnerabilities. Such measures are crucial for all aspects of IoT systems, such as network services, Web, API, or mobile interfaces, updating mechanisms, data at rest and data in motion etc. Additionally, the use of outdated or insecure components adds risk that could be mitigated by selecting more suitable IoT components.
- Weak Physical Security: A strong focus on electronic security falls short when considering IoT systems. While traditional Internet services are typically located in secure data centers, with armed guards, segregated physical access, bio-metrics, and a range of other physical security measures, a significant portion of IoT components are within the physical reach of not only consumers, but also individuals with nefarious intentions. A concerted effort is required to decrease the likelihood that an IoT system can be compromised due to this physical availability.
- Weak Authentication and Privacy Design: Careful design effort must be put forth with regard to the authentication mechanisms of the system as well as the privacy needs of its users. Since it is known that some portion of the IoT system necessarily must be within physical access to the public, authentication measures must account for the likely attempt to manipulate components in an attempt to gain access or influence the system. Additionally, measures must be undertaken to segregate and protect user data and perhaps even obscure the usage of the system by individuals in cases where just the acknowledgement of system usage might present other risks for system users.

- Weak Monitoring and System Management: Regardless of the initial efforts to design and implement a secure IoT system, a concerted effort must be undertaken to continually monitor, evaluate, and upgrade the system. The IoT system will change over time in response to consumer demand for new features, changing market conditions, and the discovery of flaws in standard Internet protocols upon which the system relies, along with other factors also resulting in the modification and upgrading of the IoT system. Inadequate effort in system management and maintenance will almost certainly result in negative consequences, such as degradation or even complete failure of the system, exposure of sensitive customer information, or even loss of system control to a bad actor with the aim of using the system for nefarious purposes.

2.2. Network Security in General

Network security has been a challenge for almost as long as networks left test labs and entered the real world. Recent estimates indicate that, in the United States alone, the costs of malicious cyber activity may exceed 100 billion dollars per year by some estimates [22]. As the years have gone by, network attacks have appeared, and then are usually pushed back as mitigation strategies are created and circulated amongst the Internet community. Unfortunately, these attacks are often either modified to escape detection or entirely new ones arise to take their place.

While it may not be obvious to the average user, the effort of attackers to compromise Internet connected systems presents a continuous and difficult challenge to the operators of these systems. The Ponemon Institute's 2018 Cost of a Data Breach Study revealed that the average time to detect a data breach was 197 days [23], and, while it was better than past years, it was only an incremental improvement. In 2019, the time to detect a network breach increased to 206 days [24], with an additional 73 days on average to contain the breach. Network and cyber security continue to be a very serious problem [25], and machine learning has been commonly used to detect attacks and threats from numerous platforms, including PCs and smartphones [26]. While malicious cyber activity is already a significant problem, these problems are compounded for IoT systems that rely heavily not only on connectivity, but also low network latency.

2.3. IoT Threats and Security Challenges

IoT systems present new challenges in an already complex security landscape on the Internet. As previously discussed, just the sheer increase in the number of connected IoT devices presents a significant challenge in and of itself. In addition, the surge in IoT systems brings connectivity to new classes of devices on a regular basis, each of which may present unique challenges which will need to be dealt with. While some may be obvious, others are less so.

IoT is pushing the edge of the Internet further into our daily lives as new devices continue to come online that had only been strictly offline before. The accelerating pace with which these are introduced can mean that testing and true security design take a back seat to the motivation of getting new products into the marketplace before the competition. As if all of this was not enough, IoT devices, unlike their predecessors, do not usually exist within the relatively safe confines of secure data centers, but instead are within the easy reach of those that might seek, by means of physical access [27], to exploit any vulnerabilities that may exist within that IoT system. Beyond these concerns, the very fact that IoT devices are often within the confines of our homes, offices, and publicly accessible areas means that these devices, when compromised, can be used to spy on unsuspecting individuals within close physical proximity to the devices [28].

Many of the attacks mentioned earlier, though sufficiently studied and understood, are not necessarily easy to prevent in the future. It has been speculated that Stuxnet initially infected Iranian servers by means of a USB thumb drive. Controlling physical access, even in secured areas, can prove challenging, while ever greater interconnectivity obviates the need for physical access. The remote hack of Jeep in 2015 is a considerable cause for concern. The desire to make cars connect to consumer

devices such as cellular phones opens up new avenues of attack and it should be clear the results could be of great consequence.

The fact that the Mirai attack was accomplished by means of simply logging in to the underlying processor using the default username and password is beyond concerning and illustrates how manufacturers and integrators must exercise greater diligence to protect the end consumer by taking sensible precautions when configuring the devices they deploy. Perhaps the greatest concern of all is the pacemaker recall by the FDA in 2017: a product recall by the manufacturer would be of little consolation to a family member who lost a loved one due to a compromised pacemaker.

IoT system vulnerabilities vary significantly based upon their location in the seven-layer reference model. It is clear that physical devices are vulnerable to many possible manners of attack due their physical availability to attackers. An attacker can gain physical access, tamper with, and possibly reverse engineer an edge device with the aim of gaining access to the entire system. The communications layer is also an obvious attack surface as the Internet is the communications medium used and is known to be vulnerable to many types of attacks on protocols, ranging from well-known attacks to unknown zero-day attacks. Less common are attack methodologies that might seek to infiltrate or disrupt edge computing, data Accumulation, or data abstraction, but these layers still suffer from software bugs that may be lurking within third party code and devices. Poorly written software can still be the target of SQL injections and other attack types that have been used to infiltrate data stores in the past. Applications suffer from the same problems and must be thoroughly tested in an attempt to mitigate attacks that take advantage of poorly designed software, firmware, and hardware. Users are also an often-overlooked flaw in system design. The best designs are ones that do not rely on common sense or particularly keen human interaction in terms of security.

2.4. Existing Work on IoT Network Security

In 2018, Xiao et al. expounded upon the various problems that are faced by IoT system designers, especially the computational weaknesses of edge devices in general [29] and further discussed possible approaches to address likely attack methods such as the denial of service attacks, man-in-the-middle attacks, spoofing and others by means of various machine learning methods. They concluded that machine learning held the possibility of being effective at dealing with some of these attacks, and should be explored in greater depth.

Meidan et al. proposed a whitelist model training approach and used random forest in an effort to later predict unauthorized devices on the network [30]. Doshi et al. evaluated numerous predictive models trained by means of several common machine learning algorithms, and found a high degree of efficacy using what they referred to as "stateless" features which they define as flow-independent characteristics of individual packets. These were found to perform well when compared to "stateful" features, defined as features that capture how network traffic evolves over time [31]. Even though IoT is still a relatively new market segment, many have demonstrated that machine learning techniques can be effectively used to address IoT security issues [29,31,32].

For years, research has demonstrated the effectiveness of using machine learning techniques to address network security concerns. In 2007, Moskovitch et al. demonstrated an accuracy of over ninety percent in the prediction of worm activity [33] and was awarded a patent in 2013 for a system detecting malicious behavioral patterns in a computer using machine learning [34]. Since Stuxnet, various approaches have been proposed for dealing with such attacks. Ponomarev and Atkison proposed a telemetry-based machine learning approach and demonstrated accuracy for some models in excess of ninety percent [35]. Nath and Mehtrel concluded that the advanced multistage nature of newer, more sophisticated attacks calls for machine learning approaches [36].

IoT systems are particularly vulnerable at the edge and network. At first glance, it may appear that the physical availability of edge devices might be the greatest opportunity for attackers. While IoT systems are already incredibly diverse in nature and function, they all share a common trait: the reliance on Internet connectivity for system communication and interaction. If there exists a most likely method

of attacking any IoT system, it is difficult to argue that any other layer than the communication layer is more vulnerable. Given this, the approach taken in the case study in Section 3 is squarely aimed at the detection of attack activity by observing network activity.

The approach of "Fog" computing has been proposed and discussed as a method not only to keep the voluminous data generated by IoT systems local, but also as a method to improve security [19], but, in reality, new vulnerabilities are introduced with this approach, lessening possible security gains from this design. Indeed, while fog computing does significantly decrease traffic to and from IoT core processing, it introduces complexity that increases the difficulty authentication, securing transient data, and maintaining user privacy [37].

2.5. Categorization of IoT Attacks by Layers

While there exist a great variety of attack types, they generally fall into one or more attack categories. Worms are generally categorized by their ability to spread throughout a network or system by exploiting one or more means of replication and transmission. Denial of Service attacks, in both singular and distributed form, are generally characterized by the usurpation of resources needed by the system to operate effectively. SQL injection is accomplished by means of inserting malicious code into system inputs in an attempt to modify system behaviour. Spoofing is the process of impersonating a different entity for the purpose of subterfuge or gaining illicit access to a system. Eavesdropping is the method of intercepting and exploiting system communication. Jamming is a method of interfering with wireless communication by means of broadcasting illicit noise into the frequency band used by the system. Malware is illicit code that takes control of one or more components of a system in order to take control of the system. Brute force in characterized by the attempt to gain entry by a process of mass repetition in an attempt to secure unauthorized access. Reverse engineering is the process of studying a system in order to determine how best to exploit it, usually by means of dissecting various system components. The layers these categories seek to exploit, as well as the citations with further detail, are seen in Table 2.

Table 2. Attack categorization and reference.

Attack	Description	Layer(s) Impacted	Reference(s)
Worm	Attack capable disrupting disparate system components by means of network traversal	2	[6,8,15,28]
DDOS	Disruption of operation by usurping system resources from multiple attack origins	2	[10,15,16,28]
SQL Injection	Degradation of system integrity by means if invalid data insertion	4,5,6	[15,28]
Spoofing	System infiltration by means of disguise, subterfuge, or counterfeit credentials	1,2,3,7	[15,28]
Eavesdropping	Unauthorized interception of system data	1,2,3,7	[15,16,28]
Jamming	Saturation of communication channel to prevent proper system operation, typically in a wireless channel	1,2,3	[9,15,28]
Malware	Infiltration of malicious code to commandeer, disrupt or otherwise interfere with proper system operation	1,3,4,6,7	[15,16]
Brute Force	Repetitive attempts to infiltrate the system by means of exhaustive combinations of possible credentials	1,3,6,7	[15]
Reverse Engineering	Physical or logical system dissection in an attempt to discover actionable weaknesses or design flaws	1,3,7	[9,15,16,27,28]

3. Data-Driven Defense from IoT Attacks

In this section, we first propose a data-driven IoT defense framework, as show in Figure 1, and elaborate upon the major components of the system. This data-driven defense framework results from the consideration of factors already mentioned herein as well as the design of the real world IoT system studied in Section 4. In order to assess the effectiveness of the security design of the case study in Section 4, we trained predictive models to validate the detection of attacks found in the wild as well as to comment on the positive aspects of the case study, and identify areas for improvement in the IoT system reviewed in the case study.

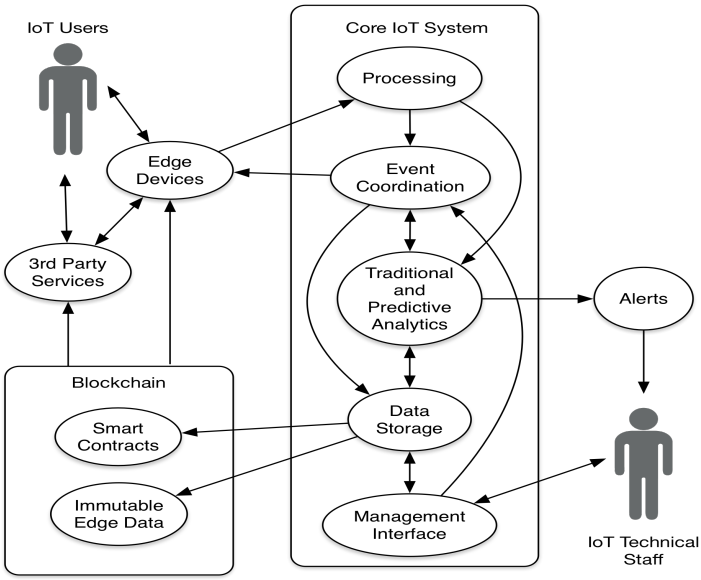

Figure 1. The proposed data-driven defense framework for IoT systems.

3.1. Data-Driven IoT Defense Framework

Effective security measures for any IoT system require a holistic perspective of the overall design and a data-driven approach in order to assure that all reasonable measures are taken to address conceivable vulnerabilities. This means considering all system components, their interdependence as a whole and the desired behavior when interacting with any outside influence. This also means designing the system to provide adequate operational data for the analysis of system integrity, performance, and errors. The architecture proposed herein delineates a generalized system from which to build a secure IoT system and is accompanied by key principles intended to promote the secure design and operation of IoT systems. It is worth noting that this framework is contemplated within the context of modern clustered computer big data architectures. Modern networks operate with great velocity and such volumes require highly parallel and concurrent processing. This is stated explicitly rather than simply implied as it is an essential design consideration. However, a reasonable treatment of clustered architectures is beyond the scope of this paper.

Processing: Input from edge devices is received in the processing subsystem, which handles validation and authorization of edge devices and governs their ability to communicate within the system. Authorized traffic is directed to the Event Coordination subsystem. The output from processing is also provided to the Analytics Engine for additional evaluation. The important step of validating edge devices and end user inputs is critical to the overall effectiveness of the system so it is imperative that this step be carefully executed to assure the trustworthiness of the data entering the system.

Event coordination: Authorized traffic from edge devices is routed here to be acted upon. Operations of a volatile nature are sent back to edge devices. Transactional and operational events are sent to the Analytics Engine and are persisted in Data Storage. This subsystem is the logistical center

of the system, providing the need interplay between all of the other subsystems, and ensuring that all transactions are completed or logged as errant.

Analytics engine: Events received from Event Coordination are analyzed in the Analytics Engine, Some of theses events are also evaluated by pre-trained predictive models (which will be addressed in the next subsection). In some cases, events and predictive models will trigger alerts for technical staff to review and possibly act upon. Models are regularly re-trained on new data to address changes or concept drifts [38]. This subsystem acts on events in the logs and KPIs in an aggregated fashion so that system personnel are provided with the necessary data, and in such a fashion to promote a better understanding of system operation so that management tasks can be well informed, in order to take appropriate measures in the day-to-day operation of the system.

Data storage: All of the necessary data of the system are stored in a storage subsystem, which provides the core components, edge devices, and third party services data as needed to ensure operation. Data within the system are encrypted at rest to further assure data security. Edge and third party services do not communicate directly with the Data Storage subsystem, but rather by means of either a series of transactions with the Processing and Event Coordination subsystems, or by means of a publicly accessible Blockchain subsystem.

Management interface: The Management subsystem provides an interface for technical staff to interact with the system for the purpose of configuration changes, threat mitigation, or other operational duties needed to assure continued operation and system fidelity. This subsystem is likely to be tightly coupled with the Analytics Engine, empowering system personnel to take action based upon system dashboards, reports, and alarms generated in the Analytics Engine.

Blockchain: The Blockchain subsystem provides a publicly accessible data set for edge devices and third parties, enforcing the design and configuration, and controlling interaction with third party systems and users of the IoT system. The immutable nature of the Blockchain makes it an ideal repository for IoT system data that must be publicly available to edge devices. This immutable quality also allows for the creation of smart contracts which can oversee and govern the behavior of edge devices and third party services in a safe and equitable fashion. Smart contracts may even be used to automatically address events that are driven by Service Level Agreements with third party vendors. Additionally, license management of edge devices is a good candidate for management by means of smart contracts.

Edge devices: Edge devices are the interface between the customer and the Core IoT System. Edge devices also provide some manner of utility or service to the IoT user. The physical availability of edge devices makes them vulnerable to tampering and reverse engineering, so additional care is taken to implement secure communication protocols and immutable behavior is provided by means of smart contracts encoded in the Blockchain ledger. Edge devices also utilize immutable data encoded in the Blockchain ledger.

Third party services: Services outside the control of the IoT provider are provided by third parties. These services interact with the edge devices and their behavior and configuration is encoded in the Blockchain ledger as immutable data, or in the context of a smart contract, and is accessed as needed. Examples of third party services might include a commercial domain name service that provides the system information needed for the IoT system's edge devices or an Internet Service Provider providing the transit of IP traffic. Another good example of third party services is a certificate authority for third party verification of any data types that might be signed by a secure certificate, such as software and other digital assets.

In summary, we believe that the principles of securing IoT systems must necessarily include the following key actions and best practices:

- Secure all components: All hardware, software, and network components used in the system must be reviewed and analyzed for vulnerabilities. Any potential flaws or issues discovered must be documented and addressed. This includes all third party components as well as embedded

firmware. Updating firmware is advisable only in the event that a vulnerability has been identified in the existing firmware or is absolutely necessary to support new system features.

- Secure digital environment: Use best practices when communicating between system components. Consider the nature of the communication between each component and select appropriate protocols for each type of communication. Encrypt all data in motion and at rest and use either an internal or third party certificate authority to digitally sign all communications so that they can be verified for authenticity as needed. Firewalls, DMZs and other mechanisms should be implemented to keep systems secure. They should be configured to output data into the Analytics Engine and be part of system alarming. Evaluate known weaknesses of system protocols used in the system and take reasonable measures to address or work around all perceived weaknesses. As much of the functionality as feasible should reside in the system core. Data that must be exposed should be encoded in the Blockchain by means of smart contracts when possible. Explicitly limit functionality on edge devices that are physically accessible to outsiders and consumers to the greatest degree possible. All communications between core system components should take place on private networks and limit external communications to only those necessary for system operation. Sanitize all intended communications between interfaces, both within the system and with third party systems to prevent injection attempts. Consider known weaknesses that may exist in frameworks and third party systems upon which the IoT system may rely, and take prophylactic measures to mitigate issues before they occur.
- Service Level Agreements: All services, systems, and components provided by a third party vendor or partner should be governed by a well-conceived Service Level Agreement that documents and specifies actions, availability, corrective measures, response times, escalation procedures and other issues surrounding the relationship and nature of the agreement and the expectations of service performance by the third party in relation to the IoT system.
- Geo-redundancy/private lines: In order to maintain resilient core function, if feasible, build two or more geo-redundant core nodes, in diverse locations, to maintain system operation in the event of a natural or other disaster that could cause a core node to become unavailable. When possible, geo-redundant core nodes should be connected using private, redundant leased lines, rather than the Internet.
- Secure physical environment: The core system should reside in one or more facilities that are designed for production grade Internet server hosting. Such facilities have backup power, multiple network carriers, diverse entry points, armed security, bio-metrics, and a wide variety of controls in place to ensure the highest level of both physical and digital security. All edge devices should be designed using tamper-resistant methods and mechanisms to alert the system regarding misuse or tampering. Automatic expulsion should be performed to protect the IoT system. Careful consideration should be given to authentication and the minimal set of permissions should be assigned to the edge.
- Privacy and data governance: Data should be segregated in such a manner as to assure that only data that is necessary and proper should be available to users of the system. Each account type within the system should be given only enough trust to accomplish the necessary function within the system. The data within the system should be partitioned in such a fashion so that no user can obtain information about other users of the system. This means that consumers can see only their own data, and not be able to access data for any other user. Protections should be in place to assure that system managers and IoT providers know and acknowledge the correct and proper use of the system and only have access to the data needed to perform their duties in maintaining the system, maintaining user privacy to the greatest degree possible. Every reasonable effort to maintain the privacy of system users should be undertaken and audited regularly. All applications distributed for use with the IoT system should be digitally signed with a certificate and registered with the appropriate third party distributor. All upgrades of system components, and especially edge devices, should be executed by means of secure mechanisms.

- Analytics: Every system transaction, event, and error should be logged with all relevant data ingested into the Analytics Engine for evaluation. Traditional analysis should be regularly performed as well as predictive analysis by means of machine learning. Alerts should be generated by means of established rule sets and as a result of anomalous behavior for further review by technical staff. Methods for the automatic detection and the expulsion of errant edge devices should be established to mitigate undesired system behavior, identifying devices that may have been reverse engineered, tampered with, or damaged.

These principles are not unique to IoT, but are well-recognized best practices that have been around for some time. They are included again here due to the fact that many of the high profile IoT breaches detailed in Section 2.3 could have been avoided had these best practices been adhered to carefully. For example, The Mirai attack could have been prevented had the components and environment been carefully secured prior to release. Indeed, it is pointless to spend any serious effort on security without rigorously undertaking the aforementioned actions and best practices.

Predictive model deployment: Any system of predictive models deployed with the intention of augmenting the security of an IoT system must be designed explicitly with the purpose of optimizing system fidelity. To these ends, at the most basic level, the system shall be designed with the goal of minimizing noise (undesired "attack" traffic and anomalies) while at the same time maximizing the signal (desired system behavior). Given that every IoT project provides unique functionality it would be unreasonable to propose a specific general design beyond mere guidelines. In Algorithm 1, we propose a basic framework to serve as a conceptual starting point for such a design.

- Datastream $\mathcal{T}$: A datastream of network packet traffic consisting of IoT system traffic ("signal") and unwanted illicit traffic ("noise"). This raw traffic is pre-processed and organized into sessions, based upon packet commonalities such as source and destination IP addresses and port, and also temporal characteristics.
- Predictive model corpus $\mathcal{C}$: A set of predictive models trained by means of machine learning attributes for the purpose of predicting IoT traffic classes. This set is initially empty and gains members as predictive models are successfully trained and added.
- Set $\mathcal{H}$ of machine learning algorithms: A selection of machine learning algorithms for training predictive models: $\{h_1(), ..., h_n()\}$.
- Labeling function $\mathcal{L}()$: A labeling function is used to label all sessions based upon the ground truth.
- Attribute extraction functions $\mathcal{A}()$: A set of functions for extracting machine learning attributes/features from characteristics derived from the traffic. A set of features used in our experiments are summarized in Table 3.
- Performance threshold $\mathcal{Z}$: An established threshold of one or more training metrics to determine model efficacy and eligibility for promotion to membership in $\mathcal{C}$.

Algorithm 1: Model training for IoT attack detection.

INPUT: $\mathcal{T}$ `// IoT network packet traffic`
OUTPUT: $P_1, \ldots, P_n$ `// predictive models`

$\mathcal{S} \leftarrow \mathcal{A}(\mathcal{T})$ `// Extract and map attributes/features`
$\mathcal{I} \leftarrow \mathcal{L}(\mathcal{S})$ `// Label network session instances`
forall *ML algorithms h in* $\mathcal{H}$ **do**
  $P_{model}, P_{metrics} \leftarrow h(\mathcal{I})$ `// train model`
  if $P_{metrics} \geq \mathcal{Z}$ **then**
    $\mathcal{C} \leftarrow P$ `// Promote model`
  else
    discard P

Table 3. A summary of created features using session based IoT network traffic data.

Attribute	Name(s) in Datasets	Definition
Session repetition	`in_rep`, `out_rep`	The packet count, per session, of all packets that are of the most common packet size
Session periodicity	`in_prdcty`, `out_prdcty`	The measure of periodicity within a session, given by the variance of timestamp differences between packets
Session convergence	`in_conv`, `out_conv`	Self-similarity of the packets in the session, determined by examining the variance in size of the packets
Packets per second	`invel_pps`, `outvel_pps`	Velocity of the traffic measured in packets per second
Bits per second	`invel_bps`, `outvel_bps`	Velocity of the traffic measured in bits per second
Bytes per packet	`invel_bpp`, `outvel_bpp`	Velocity of the traffic measured in bytes per packet
RIOT packets	riotp	Ratio of Inbound to Outbound Traffic measured in packets (inbound and outbound combined)
RIOT bytes	riotb	Ratio of Inbound to Outbound Traffic measured in bytes (inbound and outbound combined)
Duration	duration	The total elapsed time of the session (inbound and outbound combined)
Byte count	`orig_bytes`, `resp_bytes`	Session traffic size in bytes
Packet count	`orig_packets`, `resp_packets`	Session traffic size in packets

Event detection: Once a predictive model has gained membership in $\mathcal{C}$, it is used to make class predictions for new sessions, as shown in Algorithm 2. As network traffic variety and volume varies significantly due to a number of factors, the design and operational considerations of the model corpus must be able to scale horizontally and vertically in order to accommodate such variance. In addition to the aforementioned requirements, these additional requirements are contemplated below.

- Key performance indicators $\mathcal{K}$: A set of numerical metrics produced on a regular basis that measure the performance of various parts of the system. The design of these indicators is intended to measure the health of the system and reveal changes over time to conditions within the system by means of time series analysis. Such key performance indicators are a commonly used method for detecting anomalous system conditions that may require intervention to correct problems within the system.
- Mapping function $\mathcal{C}_{map}$: A mapping function that assures correct assignment of pre-processed and extracted sessions from $\mathcal{S}$ to attribute them to the appropriate models in $\mathcal{C}$ for prediction.
- Aggregation function $\mathcal{F}()$: A function for aggregating and reconciling predicted classes for sessions.
- A noise detection function $\mathcal{N}()$: A function for detecting system anomalies and/or attacks found by one or more predictive models.
- A mitigation strategy $\mathcal{M}$: The mitigation strategy is defined outside the scope of the attack or anomaly detection and receives inputs from model prediction and anomaly detection. The mitigation strategy will depend on the desired manner of addressing anomalies and attacks and is therefore subject to design decisions and the context of the particular IoT system for which it is designed. Given this fact, herein we simply consider that a mitigation strategy is a black box that receives inputs from the framework.

Algorithm 2: IoT attack detection.

INPUT: $\mathcal{T}$ // IoT network packet traffic

$\mathcal{C} \leftarrow \mathcal{C}_{map}(\mathcal{T})$ // Map sessions to models

$\mathcal{S} \leftarrow \mathcal{A}(\mathcal{T})$ // Extract attributes

forall *mapped models p in* $\mathcal{C}$ **do**

  $\mathcal{R} \leftarrow \mathrm{p}(\mathcal{S})$ // predict class

  $\mathcal{J} \leftarrow \mathcal{F}(\mathcal{R})$ // aggregate predictions

$\mathcal{M} \leftarrow \mathcal{N}(\mathcal{K}, \mathcal{J})$ // Detect anomalies and initiate mitigation

3.2. Case Study: Production IoT System

In this section, we review and evaluate an operational IoT service provider as the case study. The studied system is a secure access system that can be activated by means of a smartphone or similar mobile device. This system is in the "Community" domain, as described by Gubbi et al. [13] and the "Smart Environments" domain, as described by Aztori et al. [14]. Herein, we consider the design of this case study in light of aforementioned IoT considerations, and review the design within this context. Additionally, we use raw network traffic generated by the system in the course of normal operation to consider how the addition of predictive models trained by means of machine learning might be useful in future revisions. Specifically, we analyzed the data to identify different types of attacks that were attempted against the system over a period of several months. The system studied herein is a secured access system that allows pre-approved users to gain access by means of a revocable smartphone-enabled invitation. (Figure 2) The invitations sent by the users of the system are designed in such a manner as to only allow usage by the original recipient, and cannot therefore be forwarded to another smart device for usage. Additionally, all transactions in the system are recorded. Access to this data allows for easy labeling of the resulting dataset used for machine learning. These labels are categorized by class according to their position in the seven-layer model, allowing for multi-class evaluation.

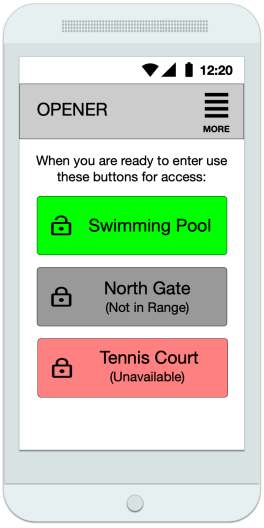

Figure 2. Smartphone application for secure community access. The invitation used by guests is a web application displayed on the smartphone or tablet of the invited guest. Once an invitation has been received by a guest it is locked to the guests device by cryptographic means and can only operate the specified gate or door when within the range specified by the community (usually just a few meters). Invitations can be sent via text message or email, and can be revoked at any time by the inviting resident.

3.2.1. System Architecture

The design of the system in this case study places a high level of importance on the overall security, integrity and fidelity of the service. The system (Figure 3) provides secure access to communities or facilities by means of a mobile device such as a smartphone that is connected to the Internet. The guest receives an invitation via email or SMS which becomes locked to the device on which it was opened. When the guest is within range of the gate, the invitation allows for the guest to open the gate and enter.

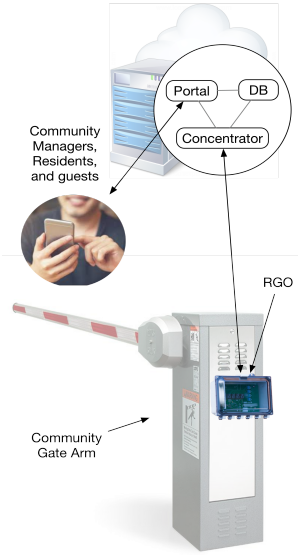

Figure 3. IoT secure access system design: Connected to the community gate arm is a Remote Gate Opener ("RGO") that can open the gate when signaled by the system. It is operated by the system concentrator which coordinates all gate access as directed by invitations contained in the database. The portal handles all input from Residents and Community Managers received from their smart devices or computer systems and communicates directly with the concentrator. Invited visitors are able to utilize invitations to communicate with the concentrator when within range of the RGO, opening the gate from a smartphone or tablet. All communication between components happens over encrypted Internet tunnels, with the exception of inter-system communication between the concentrator, database, and portal, which takes place on a private network.

Attached directly to the gate is a Remote Gate Opener (Figure 4) or "RGO", which is also connected to the Internet. The invitation used to open the gate does not communicate directly with the RGO, but, instead, with the concentrator. The concentrator validates the invitation and directs the RGO to open the gate. The RGOs communicate with the concentrator by means of a cryptographic handshake protocol. When an RGO comes online or restarts, It uses a predetermined set of DNS requests to determine the valid concentrator(s) with which to authenticate.

Residents and community managers send invitations through the portal, as well as schedule recurring invitations. The portal communicates through a secure, private channel with the concentrator. Residents and community managers may also delete invitations through the portal, and community managers can add and delete residents from the portal for that community. All log records for the system are maintained for a period of time, even after a resident account has been deleted.

The RGO is strictly an edge device. The mechanism it uses to open the gate is a simple electronic relay. It communicates with the concentrator by means of the internet over a secure encrypted tunnel. Upon power on or restart, the RGO determines the correct concentrator address and establishes the secure channel by means of a proprietary protocol upon being authenticated by the concentrator.

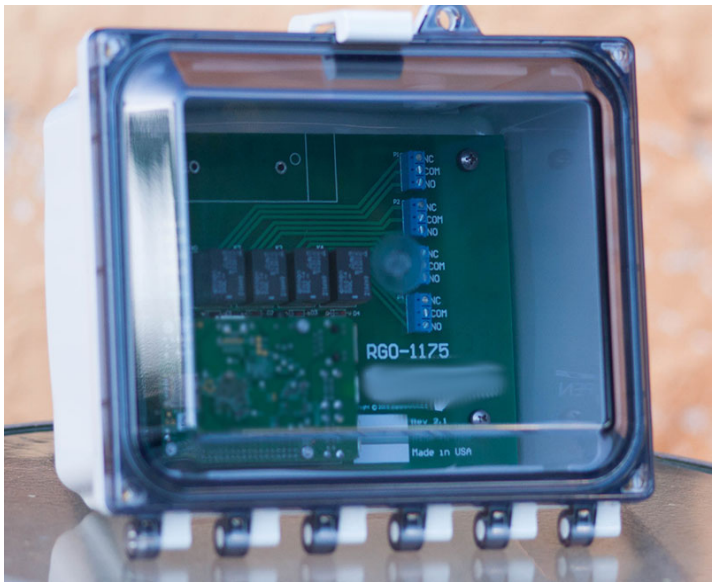

Figure 4. Remote Gate Opener (RGO) which is placed at the gate entrance access point. The RGO is connected directly to the gate or door it controls and communicates with the system concentrator on an encrypted channel over the Internet, opening the gate only for verified signals received from the concentrator.

The portal and concentrator are two separate applications that communicate with each other via a secure and proprietary protocol. Both applications communicate with a database where all data at rest is maintained. All data in the system are accumulated and abstracted as necessary within the database. It is worth noting that intra-layer communication between the concentrator and the portal happens within the same computing cluster, further securing such communication. While the concentrator only reads from the database, the portal also writes to and deletes from the database. The database itself also exists at the application layer due to all of the embedded routines and the API used to communicate with the concentrator and portal.

All mobile and computing devices for both residents and community managers communicate with the portal over the internet, for the purpose of adding, scheduling and deleting invitations, with community managers able to add, modify and delete users from the portal as well. Guest users are limited to the mobile device that opened the invitation. While it may appear to the guest that their invitation is on their phone, the interface is actually a webpage and the invitation is served from the portal, making the guests' mobile device strictly an edge device.

With regard to the communication layer, all communications, with the exception of the communication between the concentrator, portal, and database take place over encrypted Internet sessions. The portal is the interface between all of the human users and the system by means of encrypted web based micro-services, a function the occupies both Layers 6 and 7 of the seven-layer model. When user interaction at this level triggers an action within the system, the concentrator manages these interactions and operates the appropriate RGO on behalf of the user. Communications between the concentrator and portal take place on a private encrypted local area network by means of the database and inter-process communication between the portal and the concentrator.

In considering the concepts of the IAS Octave, the IoT system studied herein accomplishes confidentiality of user data by several means, including encryption both at rest and in motion, logical separation of different communities in the database schema, and an encrypted token requiring that only the original device that received the invitation is able to open the gate, the latter foiling any attempt to steal and misuse any guest invitations by unauthorized users. These measures and a well-conceived proprietary communication protocol also serve to assure the integrity of the system data. Due to this, attempted intrusion and malfeasance is easier to detect and mitigate. While the

system is vulnerable to Internet and power outages, and equipment failures; the concentrator and portal are deployed in a redundant and clustered fashion in order to assure highly available system operation, even in the event of server failure within the system. Facilities can be fitted with redundant RGOs, so, in the event of a hardware failure in one RGO, the others can continue to operate other gates as needed until the failed unit can be replaced.

The database and supporting systems track all user and administrative activity, providing accountability on a per transaction basis, and additionally all system activity and interactions are logged and can be remotely audited at any time. The digital tokens used throughout the system provide for trustworthiness of the underlying data, which assures that all third parties of the system are uniquely identified and tracked throughout their use of the system. The transactional nature of the aforementioned system allows the company to track and evaluate system usage, providing the characteristic of non-repudiation as may be required for situations such as law enforcement activity or possible evidence for any civil proceeding. Finally, individuals privacy is a paramount consideration in the system design, and beyond the measures already listed earlier, the system provides for facility administrators to reallocate system usage, but automatically does not allow new tenants to have any access to any prior tenant data.

In the analysis and evaluation of the collected network traffic data, in combination with the system logs and transactional information provided by the company, an aggregate dataset was curated, transforming the characteristics of the network traffic by means of the SANTA attributes. Subsequent to the transformation and labeling of each instance of the data, based upon the criteria subject from the logs and transactional data, the resulting dataset was used to train predictive models by means of machine learning algorithms.

3.2.2. IoT Attack Types

Three types of attacks are studied in our study (the results are reported in the Experiments section). The attacks are referred to herein as query cache (“qc”), zone transfer (“zt”), and no shared secret (“nss”). The system in the case study uses DNS entries for the bootstrapping of new or rebooted systems to load data pertinent to that edge device. Both the query cache and the zone transfer are attempts to infiltrate the system by retrieving data contained in DNS records. The No Shared Secret attack is an attack that seeks to gain access to the system by means of illicit authorization. It should be noted that all of these attacks failed in their attempts to gain access to the studied system. These attacks are all attempts to compromise the Integrity of the system by gaining unauthorized access. They are also a possible affront to the trustworthiness of the system as an interloper that gained access could possibly render the system inoperable. By recognizing these attacks, the studied system clearly demonstrates accountability, auditability, and non-repudiation. As pointed out in recent research [19] related to fog computing trust based IoT devices (the deployment of secure communications by means of secure certificates), the studied system is nearly complete in its encryption in motion, significantly raising the level of difficulty in breaching the system.

3.3. Machine Learning to Detect IoT Attacks

Based on the IoT system introduced in the above subsection, we propose a machine learning framework used to detect IoT attacks. In this subsection, we first introduce features used to characterize IoT network traffic, then discuss machine learning methods used in our study.

3.3.1. IoT Network Traffic Features

The network session-based SANTA attributes, proposed in our previous research [39], are used to characterize the network traffic and evaluate the efficacy of predictive models trained using machine learning algorithms. The attributes used are listed in Table 3. The attributes generally take advantage of the repetitive nature of many attack types and are organized into groups of packets called sessions; defined by matching source and destination addresses and ports and temporal proximity. Sessions are

full duplex. The attribute Session repetition is the number of packets that are of the most common size in that session. Session periodicity is the variance in inter-arrival times between the packets belonging to that session. Session convergence is a measure of self similarity in terms of packet size. Traffic velocity measurements of Packets per second, bits per second and bytes per packet are also employed. Additionally The ratio of inbound to outbound traffic per session are measured in bytes and packet count, and finally the byte and packet count per session are also included.

3.3.2. Machine Learning Algorithms

In evaluating the usage of machine learning for predicting the classification of attack types, the classification algorithms selected include Naive Bayes, Logistic Regression, Multilayer Perceptron, C4.5 Decision Tree, and Random Forest. These algorithms are chosen because they all support the WeightedInstancesHandler methodology [40] used in the Class Balancer pre-processing step, are well known, and often prove to be predictive. The exception in terms of predictive power is Naive Bayes; which is selected for its fast training time and for use as a baseline for comparison. Naive Bayes also supports WeightedInstancesHandler.

Naive Bayes is based upon Bayes Theorem and assumes that all attributes are conditionally independent and unrelated. The probability distribution for each attribute is determined and used for prediction. Logistic Regression uses the sigmoid function to establish the likelihood of class membership. A multilayer perceptron is a neural network, and uses gradient decent to converge to a decision boundary during the training process, for the purpose of predicting class membership. The C4.5 Decision Tree algorithm builds a tree with each node being a decision. The attributes for each node in the tree are selected by means of information gain—the spitting of the group at that node into subgroups using the most enriched class. At the stage of predicting new instances, the resulting tree is traversed until the instance is classified. Random Forest works by building a group of decision trees, each of which are trained using a random sampling of both attributes and instances, and then selecting the mode of predictions as the final prediction.

3.4. Operational Considerations

While the proposed system presented herein focuses on system design, the need is clearly implied for zealous ongoing system management. The absolute necessity of diligent management of even the simplest IoT system cannot be overstated. While a thorough discussion of systems management is beyond the scope of this paper, it is equal in importance to the system design and should be given careful consideration. Issues common to other software development efforts and Internet service management should be addressed. These issues range from reasonable systems life-cycle management, strong peer review, and rigorous testing, to best practices for hiring personnel, background checks of technical staff, due diligence in vendor selection, well-conceived security policies and procedures, including system audits. Additional efforts like penetration testing and security drills are also strongly recommended.

4. Experiments

4.1. Network Traffic Characterization

For our case studies, we collected 100 Gigabytes of data over a period of about nine months. The data include all network traffic to every device involved in the IoT company ecosystem, as well as the devices of individuals interacting with the system. More specifically, all traffic in the system traverses an aggregated, redundant network interface, so raw packet captures were collected at this point for later transformation and analysis. Tables 4 and 5 provide a more detailed breakdown of the traffic examined. Volume is the relative proportion of that IoT traffic message types divided by the sum of all traffic message types, rather than volume in bytes. While it may be initially unexpected that attack traffic significantly outnumbers legitimate traffic in terms of message volume, further examination

reveals that successful IoT system communication is more concise than brute force attempts to infiltrate the system.

Table 4. Statistics of benchmark data.

IoT Traffic Type	Volume	Features
Normal	11.7%	Norm Compliant Traffic
net (attacks)	86.0%	Protocol Violations
app (attacks)	2.3%	Invalid Messaging Attempts

Table 5. Statistics of application layer normal traffics and attacks.

IoT Traffic TZype	Volume	Features
Normal	83.6%	Norm Compliant Traffic
qc (query cache)	3.6%	Invalid cache query attempted
zt (zone transfer)	2.1%	Invalid sequence attempted
nss (no shared secret)	10.7%	Invalid credentials presented

In the period studied, just under 96% of all system traffic by volume was between the concentrator and Internet connected devices, with the portal accounting for just over 4%. During this period, 312,258 unique IP addresses attempted contact with the system, over 93% of which were illicit, unwanted traffic (a.k.a "attack"). Over 36% of traffic by volume that contacted the portal was legitimate, while just over 61% contacting the concentrator was legitimate.

With the exception of just over 0.003% of traffic, none of which was reciprocated by the IoT system; all traffic used one of three protocols: TCP, UDP, and ICMP. The breakdown of traffic by protocol is reported in Table 6. All manner of traffic herein is contemplated in terms of sessions, of which there were a total of 4,317,595, including traffic from system components, edge devices, legitimate system users, administrative functions, and unwanted attack traffic. Many services were attempted as generic lines of attack including snmp, sip, krb, irc and ftp. As the system does not use any of these common Internet services, none of these attempts were engaged by the IoT system. Among services used within the IoT system, such as dns, ssl, ssh, and https, the connections were reciprocated, and the attack attempts were logged within the system for analysis.

Table 6. Raw network traffic—byte and packet count.

		Bytes	Packets	TCP Bytes	TCP Packets	UDP Bytes	UDP Packets	ICMP Bytes	ICMP Packets
Portal	In	912 M	8 M	899 M	8 M	12 M	43 k	2 M	18 k
	Out	3.3 G	8 M	3.3 G	8 M	262 k	3 k	77 k	548
	Total	4.2 G	16 M	4.2 G	16 M	12 M	46 k	2 M	19 k
Concentrator	In	7.5 G	65 M	7.4 G	64 M	70 M	465 k	4 M	32 k
	Out	90.4 G	46 M	90.4 G	46 M	70 M	425 k	152 k	1351
	Total	97.9 G	111 M	97.7 G	110 M	140 M	890 k	5 M	34 k
System	In	8.4 G	73 M	8.3 G	72 M	70 M	507 k	6 M	51 k
	Out	93.7 G	54 M	102 G	126 M	152 M	936 k	6 M	53 k
	Total	102 G	127 M	110.2 G	198 M	222 M	1.4 M	13 M	103 k

4.2. Data Transformation

Numerous transformations are made to produce the final dataset for training the predictive models. These transformations are made by using proprietary and third party software, including Zeek and Tshark. Zeek (FKA Bro) is a software package that is designed for a multitude of packet and network traffic analysis techniques and functions [41]. Zeek is used to group the packets into sessions as well as identify network features and protocols, and is the initial stage of pre-processing

performed on the raw packet captures. Subsequently, Tshark processing is performed on the raw packet data. Tshark [42] is the command line based equivalent of the popular packet analysis software wireshark [43]. Tshark is used to identify specific characteristics of the packets that are captured, including features that can be used identify the packets that are grouped into sessions. Subsequently, proprietary software is used to calculate all of the SANTA attributes used for the training of predictive models. It is worth noting that, with the exception of duration, RIOT measured both in bytes and packets, all other attributes are calculated in a half duplex fashion and each therefore measures half of a session, and are stated as either "in" and "out" or "orig" and "resp". The attribute names, as used in the datasets, with the exception of class label, can be seen in Table 3.

For the period studied, every transaction from the system is included from the system event logs, allowing for the clear labeling of the data-set. Log entries that corresponded to various erroneous system use are categorized into attack type, based both upon its position in the seven-layer model (Case Study 1) and also more specific individual attack types (Case Study 2). The parameters of each event used to define sessions (IP addresses, ports, timestamp, duration, etc.) are also used to label the sessions in accordance with the events in the system logs. Events that corresponded to normal system expected system behaviour are labeled normal, with all others being labeled in accordance with the manner of anomalous behavior that had been previously identified by means of manual traffic inspection. IP addresses that has no association with any corresponding normal system behavior are initially tagged as unknown anomaly until such time as they are identified and given a more descriptive label.

4.3. Experimental Settings

The event logs are highly detailed allowing for programmatic labeling of all data used for model training. The training of the predictive models is done via the WEKA data mining package [44]. All of the aforementioned data iss used to construct the arff files used by WEKA for model training. In evaluating the efficacy of the predictive models, the widely recognized metrics of Precision, Recall, F-measure, and the area under the Receiver Operating Characteristic curve ("ROC area") are used. Given the high degree of class imbalance in both case studies, a pre-processing step is performed using Class Balancer in order to bring balance to the classes to avoid bias and over-fitting.

All of the predictive models are trained using tenfold cross validation and using the default parameters available in WEKA for each machine learning algorithm. The classes in both case studies are chosen based upon the real world attack data present in the logs. We consider the individual attack types in a multi-class problem (case 2), but also classes broken down by their location in the seven-layer model (case 1).

When evaluating the efficacy of predictive models, we are concerned with whether or not a target class instance is correctly predicted, known as a true positive; or is incorrectly classified as a non target class, known as a false negative. Conversely, we are also concerned with whether the non target class instance is correctly classified as such, known as a true negative; or is incorrectly classified as a member of the target class, known as a false positive. In this context, precision is defined as the count of all true positives divided by the sum of all instances predicted to be positive, regardless of whether or not they are correctly classified. Recall is defined as the count of all true positives divided by the sum of all true positives and false negatives. F-measure is the harmonic mean of precision and recall. The ROC area is the value between zero and one that is determined by plotting the true positive rate by the false positive rate at various intervals. A simple way to express this a cost benefit ratio, with numbers closer to one expressing the greatest benefit (and thus the best model performance).

4.4. Network vs. Application Layer Attack Detection Results

In the first experiment, traffic is segregated into three classes, based upon its layer in the seven-layer model. These classes are normal, network layer attack ("net"), and application layer attack ("app"). The network layer attack takes place in Layer 2 of the seven-layer model, whereas the

application layer attack occurs in Layer 6. In Table 7 we see all five predictive models compared in terms of the aforementioned metrics. The best performer is Random Forest, with and ROC area of 0.946. The decision tree also performs well in comparison, while being significantly less computationally expensive than Random Forest. The worst performing model is Naive Bayes. The confusion matrix for Random Forest is reported in Table 8.

Table 7. Network layer vs. application layer attack detection results (three types of IoT traffic including normal, network layer attacks, and application layer attacks).

	Precision	Recall	F-Measure	ROC Area
Naive Bayes	0.736	0.663	0.619	0.836
MLP	0.708	0.704	0.686	0.826
J48	0.830	0.829	0.828	0.919
Logistic Regression	0.741	0.738	0.734	0.885
Random Forest	0.842	0.802	0.792	0.946

Table 8. Network layer vs. application layer attack detection results—confusion matrix: Random Forest.

a	b	c	← Classified
498,734.29	8209.82	173.23	a = normal
12,020.28	471,530.86	23,566.2	b = net
3582.3	253,662.2	249,872.75	c = app

The performance by class for each model, as measured by the area under the ROC curve, is reported in Figure 5. The results show that Random Forest consistently outperforms all other classifiers. J48 decision trees exhibit performance only slightly below Random Forest for the normal and net class, but has significantly inferior performance in detecting application attacks. The multilayer perceptron has the worst performance. As seen in the pruned version of the J48 decision tree Figure 6, the `orig_bytes` attribute is the most predictive feature. The attributes duration, riotp, riotb, `in_conv`, and `invel_bps` are also highly predictive.

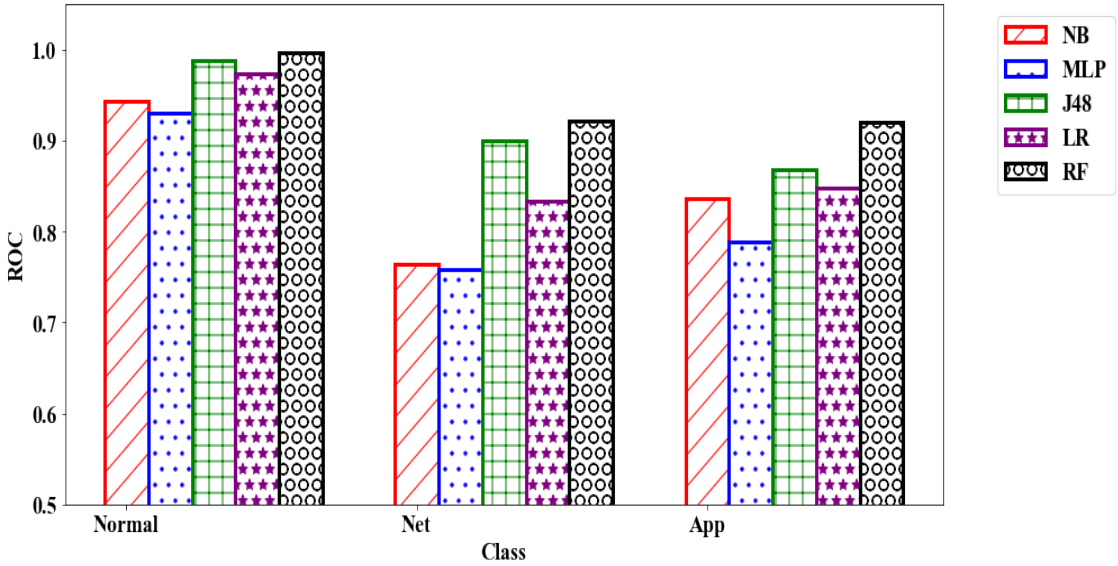

Figure 5. Area under the Receiver Operating Characteristic (ROC) curve per class for network vs. application attacks. The *x*-axis shows different classes, and the *y*-axis shows the area under the ROC curve.

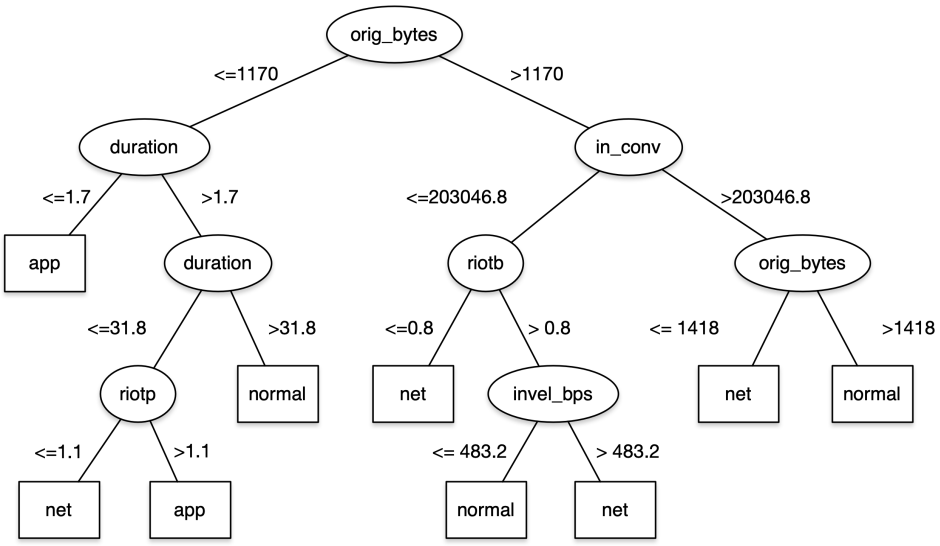

Figure 6. Pruned J48 decision tree for Case Study 1.

4.5. Application Layer Attack Detection Results

In the second experiment, the application layer based attacks from case 1 are separated into different classes for case study. In this case the classes are labeled: normal, query cache ("qc"), zone transfer ("zt"), and no shared secret ("nss"). The IoT system makes use of a proprietary protocol and employs DNS as a means of exchanging some basic startup information when an RGO comes online, either for the first time, or subsequently after restarting. The RGO requests a cache of specific information, but only after following a specific sequence of steps. If the sequence is not followed in the correct order, the request for the cached data is denied and logged. This is the query cache ("qc") error and is an indication of an unauthorized request for system data and is considered an attack. This particular attack is an attempt discover system information by coaxing the DNS servers to disclose cached information. The configuration of the DNS servers prevents this and logs it into the event log.

The zone transfer is another example of example of an attack that is an attempt to scan the components of the IoT system. This attack is also focused on DNS servers. The perpetrator is attempting to get a copy of zone information in order to gain system information contained in DNS zone records. Since the system provides certain information during the bootstrap process of the RGOs, a successful attempt could provide an attacker with information about the bootstrap process. As with the query cache attack, the configuration of the DNS servers prevents this and logs it into the event log.

During the authorization of an RGO a shared secret is exchanged. In the event the secret is expired or is invalid, it is logged as an unauthorized attempt to join the IoT network; this is the no shared secret ("nss") attack. As mentioned herein, computational resources at the edge are in short supply, and the need for encrypted channels requires a computationally expensive encrypted handshake. The violation of a shared secret allows for early termination before the expensive handshake, saving resources, and making a denial of service attack against an edge device less likely.

As seen in Table 9, Random Forest achieves the best performance, and C4.5 is only slightly inferior to the Random Forest in terms of precision, recall, and F-measure, but has a slightly worse ROC than Logistic Regression. In this case 2, the most predictive feature in the decision tree is that of outbound velocity as measured in packets per second. Subsequently, the second most predictive feature is the session size in bytes of the origination when packets per second is less 69.8 bytes on average; but when greater, the second most predictive feature is the session size of the response. Among other highly predictive features, in this decision tree, are session duration, inbound session convergence, and the ratio of inbound to outbound traffic measured both in terms of bytes and packets. The confusion matrix with these attack types are reported in Table 10.

Table 9. Application layer attack detection results (four types of IoT traffic including normal and three application layer attack types: query cache ("qc"), zone transfer ("zt"), and no shared secret ("nss")).

	Precision	Recall	F-Measure	ROC Area
Naive Bayes	0.674	0.700	0.681	0.903
MLP	0.645	0.597	0.567	0.782
J48	0.892	0.886	0.884	0.944
Logistic Regression	0.804	0.803	0.801	0.946
Random Forest	0.905	0.894	0.891	0.976

Table 10. Application layer attack detection results—confusion matrix: Random Forest.

a	b	c	d	← Classified
53,087.23	41.71	0.6	78.96	a = normal
863.55	34,796.4	0	17,548.55	b = qc
24.55	0	53,183.95	0	c = zt
619.4	3392.85	0	49,196.25	d = nss

The performance by class for each model, as measured by the Area under the ROC curve is reported in Figure 7. Random Forest consistently outperforms all other classifiers. J48's performance is only slightly below Random Forest for the normal and zt classes, but exhibits significantly weaker performance in detecting nss attack. In the case of qc attack, logistic regression outperforms J48 and performs nearly as well for classes zt and normal. The multilayer perceptron has considerably poor performance in detecting the zt class.

In addition, as shown in the pruned version of the J48 decision tree in Figure 8, the `outvel_pps` attribute is the most predictive feature. The attributes duration, riotp, riotb, `in_conv`, `resp_bytes`, and `orig_bytes` are also highly predictive.

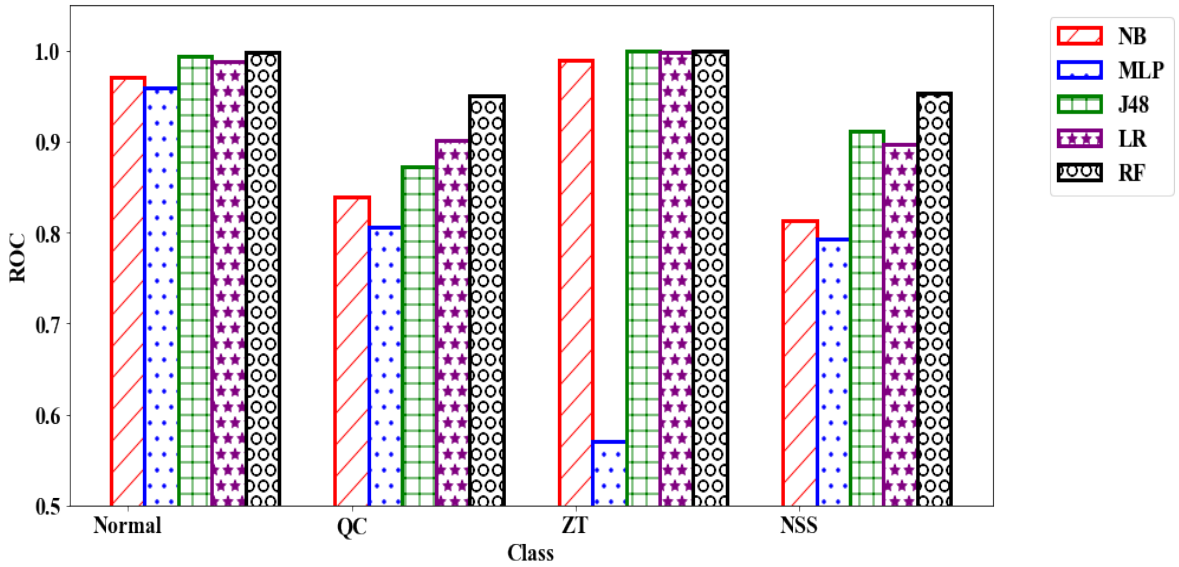

Figure 7. Area under the ROC per class for application layer attacks. The *x*-axis shows different classes, and the *y*-axis shows the area under the ROC curve.

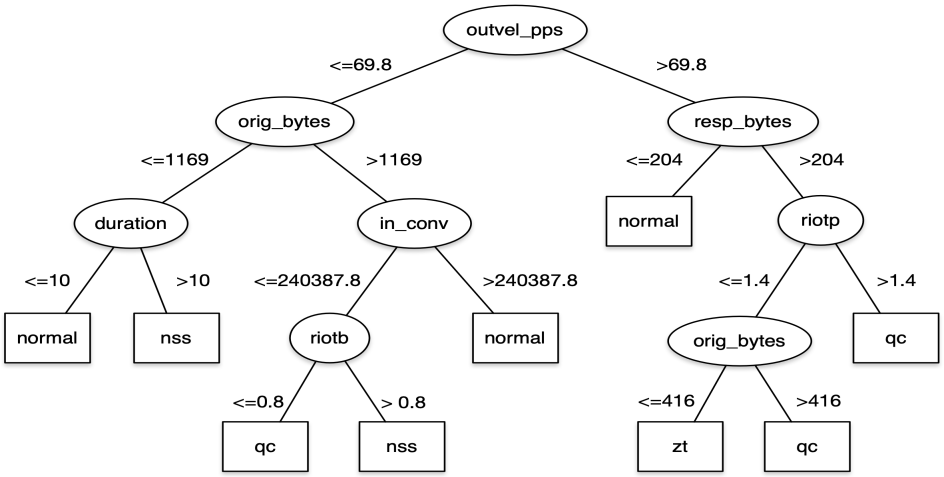

Figure 8. Pruned J48 decision tree for Case Study 2.

5. Conclusions and Future Remarks

In this paper, we first reviewed issues relevant to IoT security, the security posture of IoT systems, in terms of the IAS Octave, the Cisco seven-layer IoT reference model, and the OWASP top ten IoT vulnerabilities. Then we proposed a data-driven framework to defend against IoT attacks, and presented a real-world IoT system—a secured access system allowing pre-approved users to gain access by means of a revocable smartphone enabled invitation. Experiments, using several months of captured network traffic, illustrate the importance of various aspects of the proposed framework, and also validate the ability of machine learning models to accurately detect network layer and application layer attacks from normal traffic. It can also differentiate different types of network layer attacks, including query cache, zone transfer, and no shared secret.

Future work may include penetration testing attacks for the evaluation of attacks not discovered organically, as well as the design and evaluation of attributes contemplated to identify specific characteristics of various attack types. Attempts to better evaluate machine learning approaches for predicting IoT application layer attacks not found herein are also necessary for future study. Additionally, the attributes considered in this study are "stateful" from the perspective of [31]. It would be interesting to evaluate the performance of "stateless" attributes in detecting attacks. It would also be interesting to evaluate the performance of other machine learning approaches, such as deep learning. Perhaps a method of transfer learning similar to that contemplated in [45] might be achievable, and would be worthy of exploration.

While the proposed framework is intended to be a starting point for secure IoT system design, future work could also be realized in the implementation of an operational system. The secure access IoT system evaluated herein could definitely be improved upon by implementing this framework. DNS is a weakness that could be addressed by using a Blockchain to implement edge device bootstrap and licensing. Additionally, edge devices could be improved to resist tampering, and analytics could be introduced to evaluate edge device misuse and also physical tampering. The implementation of machine learning, trained for recognizing known attacks, may augment the system by providing a certain level of protection against zero-day attacks on the system.

Supplementary Materials: The training datasets for the case studies contemplated herein are available for download here: https://www.kaggle.com/charleswheelus/iotdatadrivendefense.

Author Contributions: Conceptualization, supervision, validation, visualization, writing—review and editing: C.W. and X.Z. Data curation, formal analysis, funding acquisition, investigation, methodology, resources, software, writing—original draft preparation: C.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research is funded by HarmonyLogic Corporation.

Conflicts of Interest: The authors declare there are no conflicts of interest.

References

1. Columbus, L. Forbes: IoT Market Predicted To Double By 2021, Reaching $520B. Available online: https://www.forbes.com/sites/louiscolumbus/2018/08/16/IoT-market-predicted-to-double-by-2021-reaching-520b (accessed on 23 May 2020).
2. O'Dea, S. Forecast Number of IoT Connected Objects Worldwide from 2018 to 2025, by Type. Available online: https://www.statista.com/statistics/976079/number-of-IoT-connected-objects-worldwide-by-type/ (accessed on 1 October 2020).
3. Robinson, G. securitytoday: Why Cybersecurity Has Never Been More Important. Available online: https://securitytoday.com/Articles/2018/02/19/Why-Cybersecurity-Has-Never-Been-More-Important.aspx?Page=2 (accessed on 23 May 2020).
4. Symantic Corporation. Internet Security Threat Report. Available online: https://www.symantec.com/content/dam/symantec/docs/reports/istr-23-2018-en.pdf (accessed on 23 May 2020).
5. Chen, T.M. Robert, J.M. The evolution of viruses and worms. *Stat. Methods Comput. Secur.* **2004**, *1*, 1–16
6. Federal Bureau of Investigation. The Morris Worm—30 Years Since First Major Attack on the Internet. Available online: https://www.fbi.gov/news/stories/morris-worm-30-years-since-first-major-attack-on-internet-110218 (accessed on 23 May 2020).
7. Maayan, G.D. The IoT Rundown For 2020: Stats, Risks, and Solutions. Available online: https://securitytoday.com/Articles/2020/01/13/The-IoT-Rundown-for-2020.aspx?p=1 (accessed on 23 May 2020).
8. McAfee. What is Stuxnet? Available online: https://www.mcafee.com/enterprise/en-us/security-awareness/ransomware/what-is-stuxnet.html (accessed on 23 May 2020).
9. Wired Magazine: Greenberg A. Hackers Remotely Kill a Jeep on the Highway—With Me in It. Available online: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/ (accessed on 23 May 2020).
10. Cloudflare. What is the Mirai Botnet? Available online: https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/ (accessed on 23 May 2020).
11. Shah, S. FDA Recalls Close to Half-a-Million Pacemakers Over Hacking Fears. Available online: https://www.engadget.com/2017-08-31-fda-pacemakers-abbott-hacking.html (accessed on 23 May 2020).
12. Cisco Systems, Inc. The Internet of Things Reference Model. Available online: http://cdn.IoTwf.com/resources/71/IoT_Reference_Model_White_Paper_June_4_2014.pdf (accessed on 20 May 2020).
13. Jayavardhana, G.; Rajkumar, B.; Slaven, M.; Marimuthu, P. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Gener. Comput. Syst.* **2013**, *29*, 1645–1660.
14. Atzori, L.; Iera, A.; Morabito, G. The internet of things: A survey. *Comput. Networks* **2010**, *54*, 2787–2805.
15. Bakhshi, Z.; Balador, A.; Mustafa, J. Industrial IoT security threats and concerns by considering Cisco and Microsoft IoT reference models. In Proceedings of the 2018 IEEE Wireless Communications and Networking Conference Workshops (WCNCW), Barcelona, Spain, 15–18 April 2018; pp. 173–178.
16. Aufner, P. The IoT security gap: A look down into the valley between threat models and their implementation. *Int. J. Inf. Secur.* **2019** , *19*, 3–14.
17. Sattarova, F.Y.; Kim, T.H. IT security review: Privacy, protection, access control, assurance and system security. *Int. J. Multimed. Ubiquitous Eng.* **2007**, *2*, 17–32.
18. Cherdantseva, Y.; Hilton, J. A reference model of information assurance & security. In Proceedings of the 2013 International Conference on Availability, Reliability and Security, Regensburg, Germany, 2–6 September 2013; pp. 546–555.
19. Mahmood, S.; Ullah, A.; Kayani, A.K. Fog Computing Trust based Architecture for Internet of Things Devices. *Int. J. Comput. Commun. Networks* **2019**, *1*, 18–25.

20. Abdul-Ghani, H.A.; Konstantas, D.; Mahyoub, M. A comprehensive IoT attacks survey based on a building-blocked reference model. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 3.
21. OWASP Foundation. The Open Web Application Security Project IoT Top 10. Available online: https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project (accessed on 23 May 2020).
22. The Council of Economic Advisers. The Cost of Malicious Cyber Activity to the U.S. Economy. Available online: https://www.whitehouse.gov/wp-content/uploads/2018/03/The-Cost-of-Malicious-Cyber-Activity-to-the-U.S.-Economy.pdf (accessed on 23 May 2020).
23. Murphy, R. Breach Discovery: How Long Does Detection Take? Available online: https://www.blackstratus.com/breach-discovery-how-long-does-detection-take/ (accessed on 23 May 2020).
24. Ponemon, L. Cost of a Data Breach Report 2019. Available online: https://securityintelligence.com/posts/whats-new-in-the-2019-cost-of-a-data-breach-report/ accessed on 1 October 2020
25. Ding, D.; Han, Q.L.; Xiang, Y.; Ge, X.; Zhang, X.M. A survey on security control and attack detection for industrial cyber-physical systems. *Neurocomputing* **2018**, *275*, 1674–1683.
26. Peiravian, N.; Zhu, X. Machine Learning for Android Malware Detection Using Permission and API Calls. In Proceedings of the IEEE International Conference on Tools with Artificial Intelligence, Herndon, VA, USA, 4–6 November 2013; pp. 300–305.
27. Nawir, M.; Amir, A.; Yaakob, N.; Lynn, O.B. Internet of Things (IoT): Taxonomy of security attacks. In Proceedings of the 2016 3rd International Conference on Electronic Design (ICED), Phuket, Thailand, 11–12 August 2016; pp. 321–326.
28. Al-Garadi, M.A.; Mohamed, A.; Al-Ali, A.; Du, X.; Guizani, M. A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. *arXiv* **2018**, arXiv:1807.11023.
29. Xiao, L.; Wan, X.; Lu, X.; Zhang, Y.; Wu, D. IoT security techniques based on machine learning. *arXiv* **2018**, arXiv:1801.06275.
30. Meidan, Y.; Bohadana, M.; Shabtai, A.; Ochoa, M.; Tippenhauer, N.O.; Guarnizo, J.D.; Elovici, Y. Detection of unauthorized IoT devices using machine learning techniques. *arXiv* **2017**, arXiv:1709.04647.
31. Doshi, R.; Apthorpe, N.; Feamster, N. Machine learning ddos detection for consumer internet of things devices. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), Francisco, CA, USA, 24 May 2018; pp. 29–35.
32. Miettinen, M.; Marchal, S.; Hafeez, I.; Asokan, N.; Sadeghi, A.R.; Tarkoma, S. IoT Sentinel: Automated device-type identification for security enforcement in IoT. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 2177–2184.
33. Moskovitch, R.; Nissim, N.; Stopel, D.; Feher, C.; Englert, R.; Elovici, Y. Improving the detection of unknown computer worms activity using active learning. In Proceedings of the Annual Conference on Artificial Intelligence, Hyderabad, India, 6–12 January 2007; pp. 489–493.
34. Moskovitch, R.; Stopel, D.; Boger, Z.; Shahar, Y.; Elovici, Y. Method and System for Detecting Malicious Behavioral Patterns in a Computer, Using Machine Learning. US Patent 8,516,584, 20 August 2013.
35. Ponomarev, S.; Atkison, T. Industrial control system network intrusion detection by telemetry analysis. *IEEE Trans. Dependable Secur. Comput.* **2015**, *13*, 252–260.
36. Nath, H.V.; Mehtre, B.M. Static malware analysis using machine learning methods. In Proceedings of the International Conference on Security in Computer Networks and Distributed Systems, Trivandrum, India, 13–14 March 2014; pp. 440–450.
37. Hassija, V.; Chamola, V.; Saxena, V.; Jain, D.; Goyal, P.; Sikdar, B. A survey on IoT security: Application areas, security threats, and solution architectures. *IEEE Access* **2019**, *7*, 82721–82743.
38. Zhang, P.; Zhu, X.; Shi, Y. Categorizing and mining concept drifting data streams, In Proceedings of the ACM SIGKDD Conference, Las Vegas, NV, USA, 24–27 August 2008; pp. 812–920.
39. Wheelus, C.; Khoshgoftaar, T.M.; Zuech, R.; Najafabadi, M.M. A Session Based Approach for Aggregating Network Traffic Data—The SANTA dataset. In Proceedings of the 2014 IEEE International Conference on Bioinformatics and Bioengineering (BIBE), Boca Raton, FL, USA, 10–12 November 2014; pp. 369–378.
40. Weka—Weighted Instance Handler. Available online: http://infochim.u-strasbg.fr/cgi-bin/weka-3-9-1/doc/weka/core/WeightedInstancesHandler.html (accessed on 28 September 2020).
41. Zeek (FKA Bro) Network Security Monitoring Tool. Available online:https://zeek.org/ (accessed on 22 May 2020).

42. Tshark: Terminal-Based Wireshark. Available online: https://www.wireshark.org/docs/man-pages/tshark.html (accessed on 23 May 2020).
43. Wireshark. Available online: https://www.wireshark.org/ (accessed on 22 May 2020).
44. Weka—The Workbench for Machine Learning. Available online: https://www.cs.waikato.ac.nz/ml/weka/ (accessed on 22 May 2020).
45. Chen, Y.; Qin, X.; Wang, J.; Yu, C.; Gao, W. Fedhealth: A federated transfer learning framework for wearable healthcare. In Proceedings of the IEEE Intelligent Systems, Varna, Bulgaria, 26–28 June 2020.