IoTAegis: A Scalable Framework to Secure the Internet of Things

Zhiyuan Zheng*, Allen Webb*, A. L. Narasimha Reddy* and Riccardo Bettati†
* Department of Electrical & Computer Engineering, Texas A & M University
†Department of Computer Science & Engineering, Texas A & M University
Email: { zhiyuanbj, allenwebbtx }@gmail.com, { reddy, bettati }@tamu.edu

***Abstract*—The infamous Mirai attack which hijacked nearly half a million Internet connected devices demonstrated the widespread security vulnerabilities of the Internet-of-Things (IoT). This study employs a set of active and passive observation methods to discover the security vulnerabilities of IoT devices within a university campus. We show that (a) the number of non-compute devices dominates the number of compute devices with open ports in a campus network; (b) 58.9% or more devices do not keep up-to-date firmware and 51.3% or more do not have a user defined password; and (c) the number of devices together with the diversity of device ages and vendors make the protection of IoT devices a difficult problem. We further develop IoTAegis framework which offers device-level protection to automatically manage device configurations and security updates. Our solution is shown to be effective, scalable, lightweight, and deployable in different forms and network types.**

I. Introduction

Network connected sensors and devices are being used in application including but not limited to inventory tracking and smart automation of buildings. Connected printers, smart lightbulbs, VoIP phones, web cameras, and smart appliances (televisions, refrigerators, washers, etc.) are commonly seen IoT devices. Other common examples include smart meters, gas pumps, medical equipment, and industrial devices. These devices employ different types of technologies such as Bluetooth [1], WiFi, Near Field Communication (NFC) [2], or Ethernet to improve their functionality and performance.

However, the lack of adequate security awareness in IoT deployment has led to widespread cybersecurity vulnerabilities that threaten to undermine not only individuals and companies, but also national infrastructures. In recent years, the IoT has become a ripe target of hackers. For example, lightbulbs have been shown to be hackable via a drone [3], smart TVs could be eavesdropping on people's conversations [4], and vulnerable Internet-connected printers could leak sensitive documents from print jobs [5]. In addition, IoT devices could be leveraged to launch disreputable attacks against other infrastructures. The recent DDoS attack against DynDNS [6] was launched from nearly half a million Mirai-powered IoT devices (mostly IP cameras and DVRs).

While computer security has received significant attention, the security of non-compute devices, or IoT devices, is only beginning to receive a similar level of attention. A number of studies [7]–[10] disclose the security vulnerabilities of specific IoT devices. Authors in [11] employ SHODAN [12] to reveal the scale of vulnerable IoT devices and identify ones with default passwords. Another work [13] provides details of an IoT honeypot and sandbox to analyze Telnet-based attacks against various IoT devices. These studies focus on narrow domains of IoT devices with limited services, but a complete vulnerability disclosure of IoT devices is still unknown.

In addition, the protection of IoT devices is a challenging research topic because of the diversity of IoT device types, communication media, protocols, and network topologies. A number of solutions [14]–[21] already exist for securing IoT devices, but the expertise and management burden of applying these fixes has led to this mainstream problem. These solutions can be classified into two categories: device level and network level. At the network level, the Norton Core [22] identifies unpatched or unsecured IoT devices in home network. In [17], an SDN platform is applied to identify suspicious network behavior, and dynamic security rules are enforced on IoT devices. An anomaly detector is developed in [20] to identify suspicious network traffic in Building Automation Networks. At device level, authors in [18] propose a secure authentication algorithm to verify the identities of clients and servers in a CoAP-based IoT environment. Studies in [14] and [15] focus on the security of IoT apps.

This paper discovers security vulnerabilities of various IoT device types. In contrast to previous study [11] that uses SHODAN search engine [12] to identify Internet-connected IoT devices, we employ a set of active and passive observation methods to discover a list as complete as possible of IoT devices within a university campus. Our datasets include not only IoT devices that can be directly visited by external hosts but also devices that are only visible to hosts within their subnet. Next, we evaluate the security of these devices, mainly focusing on default or weak credentials and unpatched firmware or software. Finally, we develop a framework to effectively secure IoT devices from the device level: it checks the unsecured IoT devices in the network and updates their firmwares, passwords, and configurations. Norton Core focuses on securing IoT devices at the network level, leaving firmware and configuration vulnerabilities on the end devices. By focusing on the problem at the device level, we aim to have an approach that works for any size network, ranging from small home networks to large campus or enterprise networks. The framework is easy to scale and applicable to different types of IoT devices. It can be the basis for a standalone application or a system service. The service would be deployed on a dedicated device or integrated with network routers. We developed a prototype and showed that our framework can not only automatically update firmwares for HP printers but also apply a non-default password. The contribution of this paper can be summarized as follows:

- We provide a methodology to passively and actively discover IoT devices that employ different services, such as mDNS/DNS-SD, WiFi, UPnP, NetBios;
- We provide a vulnerability disclosure for IoT devices in a university campus and compare the scale and security protections of IoT devices with computing devices;
- We develop a framework that is able to automatically check security vulnerabilities of IoT devices and take remedies accordingly;
- Our prototype of HP printer firmware update and password configuration shows that IoTAegis is effective, scalable, and applicable to different network topologies.

Our approach can be seen as a *device hygiene* approach, and our solution is implemented centrally within an administrative network (campus or home). While the current solution works as a software running on a workstation within a campus network, a future version of this software can run as an app within a home network or as a service on a consumer router.

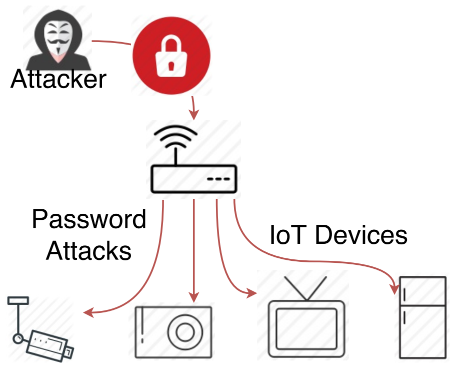
The structure of the paper is as follows: Section 2 provides a survey on network-connected IoT devices. In Section 3, we present the attacker model with different attack goals. The solution framework is presented in Section 4. Section 5 shows that our framework can automatically update firmwares of HP printers. Related work is presented in Section 6. Conclusion and future work is discussed in Section 7.

II. Attacker Model

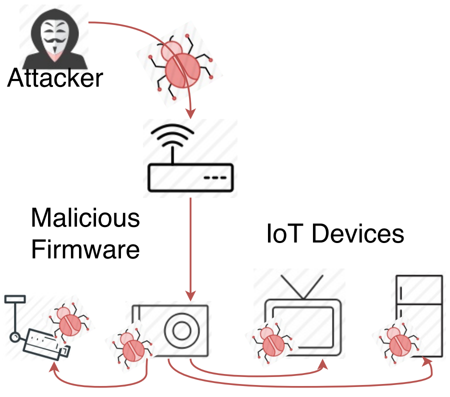
In this paper we try to prevent the compromise of IoT devices instead of reacting to various network-level attacks. We are interested in developing a solution to prevent attackers from gaining unauthorized control of the devices of interest. Thus, we consider two different attacks in the exploitation stage: *configuration attacks* and *firmware attacks*, see Figure 1.

Assumptions. Without loss of generality, we focus on IoT devices that communicate through a wired or wireless network connection. Media such as Bluetooth [1], ZigBee [23], Z-Wave [24], and 6LowPAN [25] are not considered in this paper. However, they could easily be addressed when the device supports these protocols in addition to an IP-based network connection. We assume attackers do not have physical access to IoT devices, but can access the devices over the local network within any firewalls.

Configuration Attacks. Attackers may launch password attacks that take advantage of IoT devices with weak or no password protection, *e.g.* brute force attack and dictionary attack. Once they have access, attackers may further modify the devices' configurations to prevent authorized users from accessing or controlling the devices. By default IoT devices provide some services without authentication that give opportunities to the attackers.



(a) Configuration Attacks



(b) Firmware Attacks

Fig. 1: Exploitation Attacks Considered in the Paper

Firmware Attacks. Attackers may launch firmware attacks to infect IoT devices by exploiting outdated firmware with design or implementation flaws. We assume that patches already exist in the newest firmware version and that attackers cannot exploit the latest firmware with 0-day attacks. In addition, some devices (such as any HP printer before 2011) support remote firmware update without checking the signature of the firmware to be installed [26]. Attackers who have network access to the target devices may infect the devices by uploading a malicious firmware. Exploitation may also result from unsecured protocols such as Telnet or Simple Network Management Protocol (SNMP) and from flawed parsers for data-like postscript print jobs.

Once the IoT devices have been exploited, attackers may launch *actuation attacks*: change the state of an IoT device with malicious intent, *e.g.* false command injection attacks; *data acquisition attacks*: obtain a user's sensitive data such as print jobs or phone calls; *data integrity attacks*: send false sensor data or status information to a server or other IoT device; and *Botnet attacks*: leverage the infected IoT devices to attack other hosts, *e.g.* Mirai attacks.

III. Survey

First, we are interested in understanding the relative numbers of computers versus non-compute devices on a typical campus network. Second, we expect this study to reveal the status of the IoT devices in a well-managed network. Third, we expect this study to point to potential security problems from IoT devices since the security practices between IoT and compute devices may be different. For example, the operating system software (or the firmware) may not be automatically updated on these devices, and the IoT devices may not be protected by anti-virus software.

The section below covers the methodology used and our findings on the security issues of IoT devices.

A. Methodology

We employed two primary strategies for device discovery: port scanning and using service discovery protocols.

Devices that advertise through service discovery protocols are only visible within a subnet. Thus, we walked through the campus to collect data across different subnets in the university campus. Our dataset includes computing devices, IP phones, printers, NAS, network infrastructure, cameras and surveillance equipment, industrial control devices (used in electrical grid and water distribution facilities), scientific measurement equipment, and other less common equipment.

Port scanning. Nmap [27] and Zmap [28] are both open source utilities for network discovery and security auditing. In our study, Nmap was used to identify non-SCADA devices with TCP ports 80 and 443 open. Nmap was also used to identify SCADA devices with TCP port 502 (Modbus protocol), TCP port 20,000 (DNP3 protocol), and UDP port 47,808 (BACnet protocol) open. Combined with external scripts [29], Nmap can retrieve additional device information such as firmware version, device ID, and operating status. Both DNS resolution and host discovery were disabled to minimize the impact on the network. The landing pages for devices with a service running on port 80 or 443 were downloaded using curl. The resulting pages were clustered using unique strings, and each group was manually identified. Figure 2 shows the distribution of IoT devices across the different categories.

Service discovery protocols. DNS-SD (DNS Service Discovery) and UPnP (Universal Plug and Play) SSDP (Simple Service Discovery Protocol) are two different technologies both designed to allow devices to easily discover and browse other devices on the local network so that the average user does not have to deal with the complexities of IP addresses, ports, and protocols. DNS-SD [30] uses multicast [31] instead of broadcast for cross communication and leverages the DNS protocol to advertise services available on each host as part of *zeroconf* (Zero Configuration Networking). UPnP's SSDP uses HTTP and SOAP (Simple Object Access Protocol) for host and service discovery instead of DNS.

B. Findings

The results presented in this section are from a random sampling across the measured network. We found 1828 index pages from unique hosts with port scanning and 584 unique hosts with the service discovery technique. There were 13 hosts that were common to both datasets, so the total number of unique hosts from both datasets is 2399. The port scan data revealed that IoT devices with open services are about three times more common than compute devices such as servers, desktops, laptops and smartphones. The results of the categorized data from the port scan are shown in Figure 2. Some of the 34 different manufacturers represented in the sample spanned more than one device category. Among all hosts from both datasets, we grouped all the computing devices into one category that had 692 devices making up 28.8% of the total. The non-compute or IoT devices are represented as separate categories. IP phones comprised the largest group with 1001 unique devices and 41.7% of the total. The next largest category was printers with 194 devices followed by AV (Audio/Video) equipment with 41 devices. Uninterrupted power supplies (UPS), network infrastructure such as router and wireless access points, network attached storage (NAS), server remote management cards, private branch exchanges (PBX), keyboard video and mouse (KVM) switches, a piece of science equipment and a digital sign each had less than 30 devices.

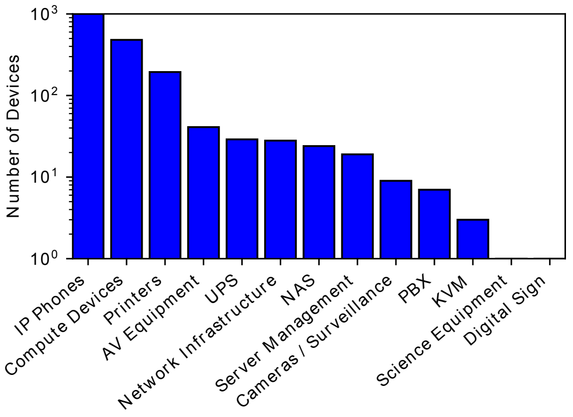


Fig. 2: Categorized Sample of Devices from the Port Scan.

While some of these devices are less critical such as KVM switches, others such as NAS, UPS and network infrastructure could lead to serious problems if compromised. Critical data could be leaked or lost, critical systems could be powered off on demand, or parts of the network could be cut off. Each brand has one or more unique administration pages for their devices, so the variety of IoT devices increases the difficulty of managing them all.

The service discovery based dataset included 78 different services. Fig. 3 shows the top ten identified services. The most common service was `_http._tcp` or web services that account for 16.00% of the unique devices identified. These web services have a wide variety of uses ranging from public web sites to administration interfaces for IoT devices such as printers, cameras, network attached storage, and more. Therefore, the presence or absence of the HTTP service is not enough to classify a device. Page description language (PDL)

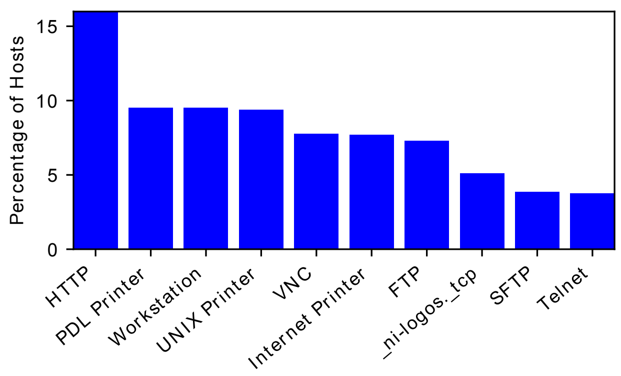


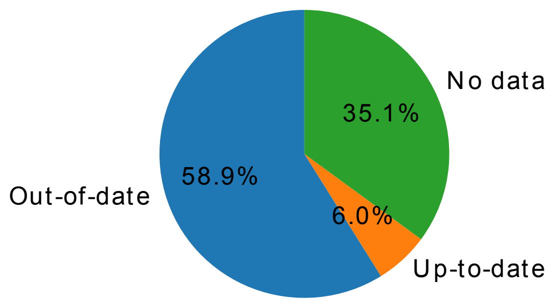
Fig. 3: Top Ten Identified Services

printers, workstations, and UNIX printers each account for about 9% of the devices; VNC, Internet printers, FTP services were identified in about 7% of the unique devices. Of these services workstation and virtual network computing (VNC) are associated with compute devices. For this dataset compute devices represent 36.1% of the included hosts.

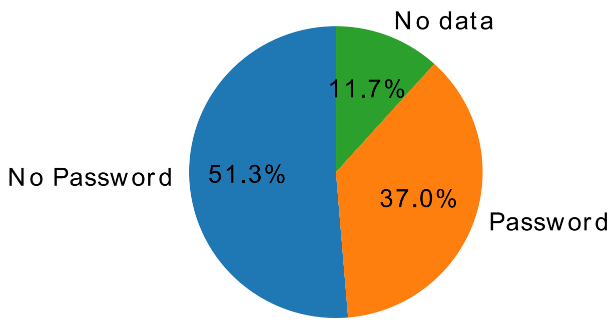
Printers. We identified printers from 11 different manufacturers. The presence of custom passwords and the firmware dates for printers were obtained using web scraping scripts. Of all the printers identified, we confirmed 51.3% had no password ever specified by the user (see Figure 4(b)). The no data percentage is different between the two plots because for some printers with a password set the firmware version is inaccessible.

Figure 4(a) shows the percentage of printers identified known to have out-of-date firmware 58.9% as well as the 6.0% that were known to have up-to-date firmware. The devices in the *no data* category timed out while trying to access the HTTP index page or had a password preventing access to the firmware date. Figure 5 is a histogram of the dates of the firmwares installed on printers we observed along with the dates of the latest firmwares available for those printers. There is a cluster of printers with firmwares made in 2010 and the first half of 2011. The newest firmware available for 12.5% of the printers was from 2014 or before. This data points to the fact that the firmware on these devices is both not being updated by the manufacturers and being updated by the users even when new version of firmware is available. This shows that even on a well managed network IoT devices like printers may be overlooked, receive minimal initial configuration, and fall behind in updates. We did observe newer devices that automatically update their firmware.

VoIP Phones. VoIP (Voice over Internet Protocol) telephones can be connected to a Private branch exchange (PBX) through a local area network (LAN). Given an VoIP phone with minimal custom configuration, the default admin password will be known [32]. Using this access the SIP configuration can be changed to launch a man in the middle attack. This enables attackers to record calls, and keep track of call log information. If the attacker manages to steal the SIP authentication data, they can place calls from the victim's phone number.



(a) Printer Firmware



(b) Printer Passwords

Fig. 4: Security Readiness of Printer Sample

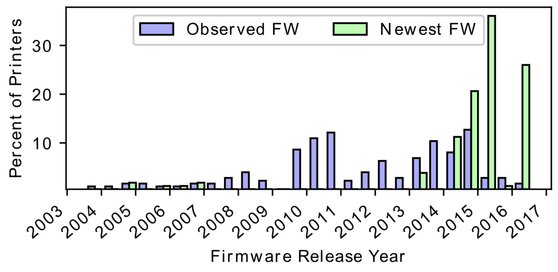


Fig. 5: Printer Firmware (FW) Release Dates

The VOIP phones dominated the network wide scan of port 80 prompting us to conduct a more in-depth scan of this subnet. We found a total of 6999 VOIP phones from the focused port scan. Only about a third of the VOIP phones were protected with a custom admin password (see: Figure 6) None of the VOIP phones had the newest firmware. One brand of VOIP phone had no password authentication for the HTTP management page and represented 20.7% of the VOIP phones. This data strengthens the point that some IoT devices do not receive the same attention to configuration and security updates as compute devices.

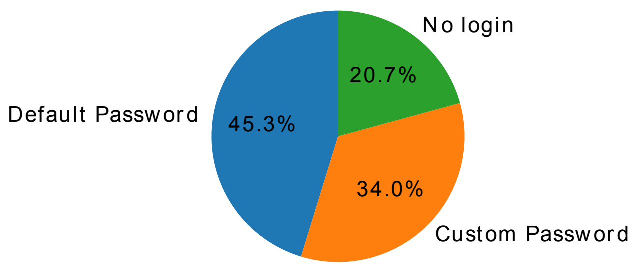


Fig. 6: Authentication for VoIP Phone Administration

SCADA devices. We have identified a total of 415 SCADA devices that are used across building automation networks, water metering system and electrical grid. This includes 197 BACnet field panels and workstations, 188 water system meters and 30 DNP3 devices. Security vulnerabilities have been identified among these devices. However, due to security reasons, we are not allowed to further discuss these vulnerabilities in this paper.

We also identified a piece of scientific measurement equipment that could be tampered with over the network interface. Adjusting the default configuration of the device with awareness of the potential problems could go a long way toward securing the scientific measurements. Ideally, the network access to the device would be restricted through a VPN to only authorized individuals.

IV. Solution

From the results of our survey, the number of IoT devices is more than that of compute devices on the measured network. While compute devices are generally protected by Antivirus and security patches are regularly updated, many IoT devices are not adequately protected. Specifically, while firmware updates may be available, 58.9% or more of the devices were not kept up-to-date. In addition, 51.3% or more devices did not have a user defined password. This problem has been recognized by others [11], [33]. The solution is made difficult by the number of devices on the network, their age range, and the number of vendors supplying these devices. Thus, we developed IoTAegis to address these identified problems directly while simultaneously reducing the management burden. The IoTAegis provides a central tool for keeping firmware up-to-date and device configurations secure.

An overview of IoTAegis is shown in Figure 7. The framework discovers hosts, identifies device types and supported services. Then, it can perform tasks needed to update the devices to the newest firmware and address security holes in the device configuration. In this way an application or system service uses the framework to manage the network devices and handle security issues as they are detected. Ideally, checks will be performed as soon as a new IoT device is connected to the network. Rather than introducing another network protocol to handle these attacks, we propose a methodology that could be

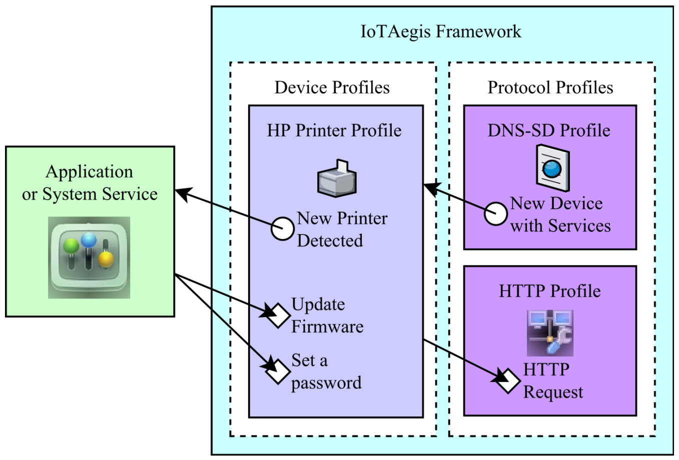


Fig. 7: An Overview of Our Solution—IoTAegis

drafted into a standardized API. Our proposed methodology consists of two types of profiles: one for handling network protocols and the other for handling devices. These profiles could be likened to device drivers used by an operating system.

Each network protocol profile fulfills two primary roles. First, it defines “requests” that are abstracted from the underlying protocol and can be used by device profiles. An example of a request would be downloading an index page over HTTP. Second, the network protocol profile defines events. These events are tied to activity at the protocol level which should be handled by listeners defined in device profiles. An example of an event would be the discovery of a new device and its services over mDNS. The DNS-SD profile includes the logic needed to discover devices using mDNS and generate events whenever a service advertisement is received.

Each device profile depends on network protocol requests or events to provide the logic required to perform a set of standard tasks. Some tasks are device specific, but here are key examples: discovering the device, identifying the device model and state, defining the password, checking what the latest available firmware is, checking the firmware version on the device, applying a secure set of configuration options, recommending restrictive firewall rules. Not all tasks will apply to every device.

IoTAegis uses an event driven architecture where protocols such as DNS-SD over mDNS produce events. These events are multiplexed by their service discovery data to the appropriate handlers. For example an IoT printer may advertise the `_http._tcp`, and `_printer._tcp` services. The `_printer._tcp` service includes a `TXT` record with information about the printer that aids in the identification of the printer manufacturer and model number. At this point the appropriate device profile can be notified through its registered event listener. Based on these events, follow up requests are made to query the firmware version of the new printer and check if the new printer has a custom password set. Further execution on that event tree can be halted, and the tasks registered for that device would then be made available to the application or system service. These tasks could be executed once on demand, periodically, or as updates are released for the device.

A device-centric solution requires the following key functions (a) host discovery, (b) device identification, (c) configuration management, and (d) firmware updates.

Host Discovery. Several well-known techniques can be leveraged for host discovery, as discussed in Section 2. IoTAegis uses the Avahi-client library [34] in our framework to construct a DNS-SD profile that generate events for discovered services and hosts on the network. Next, we aggregate results by hosts and obtain a complete list of services advertised on each host using DNS-SD per host.

Device Identification. IoTAegis matches service strings with different types of devices and automatically identifies IoT device types. Specifically, `_printer._tcp` corresponds to a Unix printer, `_pdl-datastream._tcp` to a Page Description Language (PDL) printer, `_ipp._tcp` to an Internet Printing Protocol (IPP) printer, `_ipps._tcp` and `_ipp-tls._tcp` to IPP printers using HTTPS. Within the DNS-SD `TXT` records are key-value-pairs, one of which describes the product by including the brand and model number.

Devices with insufficient identifying information or no DNS-SD advertisement may require a request to be made through a protocol profile. For example a check could be performed on their HTTP index page to match against signatures. Additional protocol profiles can be provided to expand the capabilities of the framework as necessary, and new device profiles would provide the logic required to identify more devices. Ideally, a standard API was drafted and the device manufacturer would provide its own device profiles, but third parties could develop them as well.

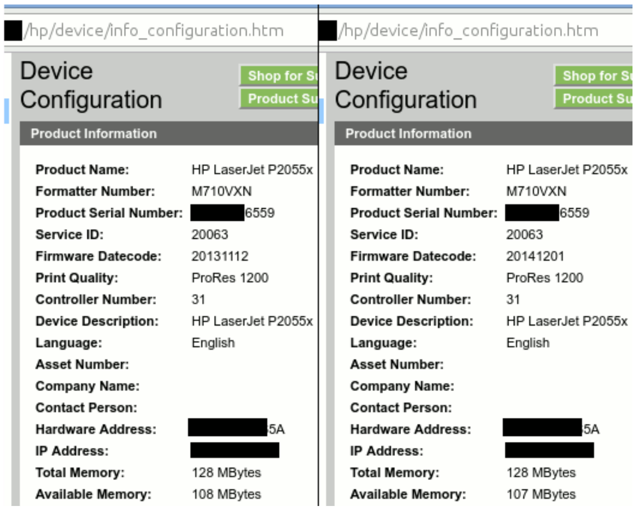
Configuration Management. Some printers for example do not have an administrative password set by default or have a default password that is easy to look up. When devices use simple HTTP authentication, IoTAegis sets a header with the default authentication information. If a page requiring authentication can be accessed with the default user name and password, it is clear the device needs to have a custom password set. The user can be prompted or an automatic password can be generated, set, and stored in a password manager where both the security framework and the user can access it as needed.

In addition to problems with the default password configuration, tasks defined in the device profile can address extraneous services enabled by default or device specific security considerations. Thus, a device profile for IoTAegis can serve as a replacement for the vendor specific device configuration software that is shipped with IoT devices. This provides the end user with a common place and interface to manage a wide variety of devices connected to their home or business network in contrast to needing to use each device's individual management interface.

Firmware Updates. While newer devices may include automatic firmware updates, our survey demonstrates there are plenty of devices running old firmware versions. To update the firmware of an HP printer IoTAegis downloads the appropriate remote firmware update (RFU) file and transmits it using TCP to the raw print port of the printer (9100). The latest update can be found by querying an FTP site for a matching RFU file with the greatest version number of date code. Some logic may be required to update very old firmware to intermediate versions prior to updating to the latest version. This logic can be included in the device profile, but in general the devices should be at most one version behind. In this case, the latest supported firmware version will install without requiring intermediate updates. Once the firmware is downloaded and transmitted, the printer executes the update commands contained in the RFU file that perform the update using the self-contained data without user intervention. Thus, the devices on the network can be kept up-to-date with the latest firmware versions available to prevent attackers from exploiting known vulnerabilities that have already been patched.

V. Evaluation

We tested the features of IoTAegis on an HP 2055x printer. We first verified that the firmware files downloaded by IoTAegis matched the ones downloaded through a browser. Then, we successfully updated the firmware twice by applying the next newest update and an update to the newest version. The initial firmware datecode was 20120615, the intermediate datecode was 20131112, and the final datecode was 20141201. The before and after screenshots of the device configuration administration page hosted by the printer for the last update are displayed in Figure 8. We hid most of the identifying information, but the last few digits of the serial number and hardware address are shown to validate that the screenshots were taken from the same device.



[redacted]/hp/device/info_configuration.htm

Device Configuration

Product Information

Product Name:	HP LaserJet P2055x
Formatter Number:	M710VXN
Product Serial Number:	[redacted]6559
Service ID:	20063
Firmware Datecode:	20131112
Print Quality:	ProRes 1200
Controller Number:	31
Device Description:	HP LaserJet P2055x
Language:	English
Asset Number:	
Company Name:	
Contact Person:	
Hardware Address:	[redacted]5A
IP Address:	[redacted]
Total Memory:	128 MBytes
Available Memory:	108 MBytes

[redacted]/hp/device/info_configuration.htm

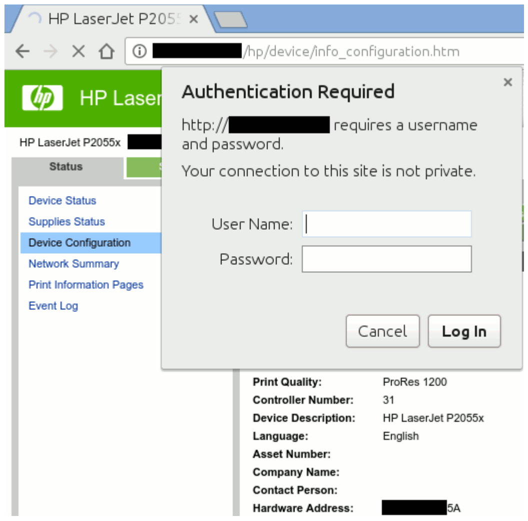
Device Configuration

Product Information

Product Name:	HP LaserJet P2055x
Formatter Number:	M710VXN
Product Serial Number:	[redacted]6559
Service ID:	20063
Firmware Datecode:	20141201
Print Quality:	ProRes 1200
Controller Number:	31
Device Description:	HP LaserJet P2055x
Language:	English
Asset Number:	
Company Name:	
Contact Person:	
Hardware Address:	[redacted]5A
IP Address:	[redacted]
Total Memory:	128 MBytes
Available Memory:	107 MBytes

Fig. 8: Printer configuration page before (left) and after (right) IoTAegis downloaded and updated the firmware.

IoTAegis also successfully configured the printer to use custom password. Figure 9 shows the authentication prompt when attempting to access a sensitive part of the printer's administration page. We envision IoTAegis being integrated with a password manager of the user's choice. This gives the user easy access to the passwords in a secure place, and IoTAegis can retrieve the passwords when future configuration or firmware updates are needed.


HP LaserJet P205

[redacted]/hp/device/info_configuration.htm

HP Laser

HP LaserJet P2055x [redacted]

Status

- Device Status
- Supplies Status
- Device Configuration
- Network Summary
- Print Information Pages
- Event Log

Authentication Required

http://[redacted] requires a username and password.

Your connection to this site is not private.

User Name:

Password:

Cancel | Log In

Print Quality:	ProRes 1200
Controller Number:	31
Device Description:	HP LaserJet P2055x
Language:	English
Asset Number:	
Company Name:	
Contact Person:	
Hardware Address:	[redacted]5A

Fig. 9: Authentication prompt when accessing a sensitive page after IoTAegis set a password.

These demonstrations show that IoTAegis is able to solve the two primary channels of attack motivated by our survey data. Attackers will not be able to exploit known vulnerabilities in out-of-date firmware or gain control of devices through insecure configurations such as default passwords.

Solution Costs We implemented a proof-of-concept framework and demonstrated that we are able to automatically check passwords on IoT devices and update firmware on HP printers. We tested the proof-of-concept on an Intel Core i5-3330 CPU at 3.00GHz running Ubuntu 16.10 and recorded the CPU time as well was the network usage for the first 60 seconds of operation. The test subnet had 67 compute devices and 25 printers. The network usage by protocol is shown in Figure 10, where the mDNS traffic was generated by the avahi library for service discovery and the HTTP traffic was used to obtain the firmware and password state information from the detected printers. Also, the peak network usage was about 350kBps (2.8mbps), and most of the traffic occurred in the first 20 seconds of operation.

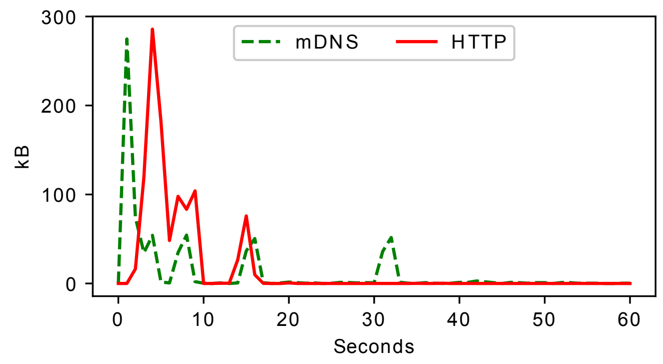


Fig. 10: Network usage during first 60 seconds of operation

During the 60-second test interval the service avahi-daemon used 1.34 seconds of CPU time in user mode and 0.32 seconds in kernel mode. The proof-of-concept app used 0.51 seconds of CPU in user mode and 0.08 seconds in kernel mode. This amounts to a utilization of 3.75% of one core during the 60-second test period. The utilization would be 11.25% if we assume all the activity took place in the first 20 seconds. The peak memory usage of the proof-of-concept app was 99.6MB, but with rate limits on the DNS-SD queries and optimization this number could be brought down to a number that would be practical for consumer grade routers.

Our password update demonstration only requires a single HTTP request to complete. This should apply in most cases unless an initial request is required to obtain a nonce to use the submission form. Verification that the update succeeded requires an additional request. One limitation of the firmware updates is the printer being updated must be in the "ready" state for the update to complete—there can be no paper jams or active print jobs.

Scalability and Scope. While our proof-of-concept is limited in scope because of the limited protocol and device profiles it contains, the general principles behind its design can be applied to a wide variety of IoT devices. The primary requirement for IoTAegis to work with an IoT device is it must have the appropriate device profile network protocol profile and the network profile dependencies. Protocol profiles can be created for Bluetooth and near field communications to support non-IP-based devices.

IoTAegis can be easily deployed in different forms such as a phone app, a browser extension, or as a feature of a network device or home router. It is already suitable for both wired and wireless Ethernet interfaces. The development of a standardized API would make it easier for device manufacturers and third parties to develop profiles supporting more IoT devices. This also enables alternative, but compatible, framework implementations to be developed that can use the same profiles.

VI. Related Work

IoT Vulnerability Disclosure. The security issues of IoT devices have drawn people's attention especially after the notorious Mirai attack. A number of papers exploit and disclose the security vulnerabilities of different IoT devices such as smart home appliances and apps [7], industrial control devices [9], home automation devices [35], HP printers [5], [26], and baby monitors [10]. In [5] an analysis of printer attacks is provided, and an open-source tool called PRET is developed to evaluate the vulnerabilities of printers.

The Internet Census 2012 [33] was conducted to scan all IPv4 addresses using unsecured embedded devices that were probed through classic telnet login `root:root`. They identified 244,000 or more unprotected Internet printers and about 2.4 million UPnP devices. Patton et al. [11] employ SHODAN to reveal the scale of vulnerable IoT devices. It classifies devices based on types and identities the ones with default passwords. Our work employs multiple strategies to discover vulnerable IoT devices in a university network, rather than using SHODAN search engine. Hence, we expect to include devices that are not visible to the public Internet but are susceptible to insider attacks launched within the campus network. An IoT honeypot and sandbox are developed in [13] to analyze Telnet-based attacks against various IoT devices. This work showed that at least four DDoS malware families target Telnet-enabled IoT devices.

IoT Defence Solutions. A number of solutions have been proposed to secure IoT devices through a security framework [16], SDN (software-defined networking) [17], and authentication [18].

ZENworks [36] is an effective security and configuration management solution for compute devices. Symantec Norton Core Router [22] aims to protect IoT devices in a home network from network-level perspective. It detects suspicious activities within a home network, and quarantines infected connected devices, but does not handle updates to the device firmware or changes to the device configuration. IoTAegis solves the problem from device level and provides a general framework that can be applied to different IoT devices regardless of their type, manufacturers, and network interfaces (wired and wireless). Our solution also provides password checks and automatic firmware updates to prevent potential exploits.

Jia et. al [14] provide a taxonomized IoT attack app dataset based on reported IoT attacks and constructing new IoT attacks. They also propose a context-based access control system for IoT devices. This system supports fine-grained context identification and runtime prompts with rich context information to help users authenticate sensitive actions and perform access control.

Fernandes et al. [15] propose the FlowFence framework that enforces the declared data flow patterns within IoT apps for sensitive data and prevent all other flows. The solution can be incorporated with existing IoT apps with small overhead.

VII. Conclusion

The infamous Mirai attack has shown that many IoT devices do not have adequate security protections and can be leveraged to cause significant damage. In this paper, an IoT security disclosure is provided that shows that about 58.9% devices do not keep up-to-date firmware and 51.3% or more do not have a user defined password. Our measurements also indicate that the number of non-compute devices may dominate the number of compute devices in a typical campus network. To solve the problem, we developed IoTAegis to offer device-level security protection for various IoT devices. Our framework is effective, scalable, easy-to-deploy with low computational cost. It enables the development of a single management platform for configuring and securing the IoT. In future work, we plan to extend our framework to include network based protection (firewall rules), support other IoT devices, and extract a flexible API from IoTAegis that would aid in the development of third party protocol and device profiles that may serve as the basis of a standardized API.

References

[1] S. Bluetooth, "Bluetooth specification version 1.1," *Available HTTP: http://www. bluetooth. com*, 2001.

[2] E. C. M. Association *et al.*, "Ecma340–near field communication interface and protocol (nfcip-1)," 2004.

[3] E. Ronen, C. O'Flynn, A. Shamir, and A. Weingarten, "Iot goes nuclear: Creating a zigbee chain reaction," http://iotworm.eyalro.net/, 2017.

[4] Samsung, "Samsung privacy policy–smarttv supplement," 2015.

[5] J. Müller, V. Mladenov, and J. Somorovsky, "Sok: Exploiting network printers," 2017.

[6] Wikipedia, "2016 dyn cyberattack — wikipedia, the free encyclopedia," 2017, [Online; accessed 19-February-2017].

[7] E. Fernandes, J. Jung, and A. Prakash, "Security Analysis of Emerging Smart Home Applications," in *IEEE S&P*, 2016.

[8] S. Yoon, H. Park, and H. S. Yoo, *Security Issues on Smarthome in IoT Environment*, 2015.

[9] J. Wurm, K. Hoang, O. Arias, A. R. Sadeghi, and Y. Jin, "Security analysis on consumer and industrial IoT devices," in *IEEE ASP-DAC*, 2016.

[10] M. Stanislav and T. Beardsley, "HACKING IoT: A Case Study on Baby Monitor Exposures and Vulnerabilities," 2015.

[11] M. Patton, E. Gross, R. Chinn, S. Forbis, L. Walker, and H. Chen, "Uninvited connections: A study of vulnerable devices on the internet of things (iot)," in *IEEE JISIC*, 2014.

[12] J. Matherly, "Shodan search engine," https://www.shodan.io, 2009.

[13] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "IoTPOT: Analysing the Rise of IoT Compromises," in *USENIX WOOT*, 2015.

[14] Y. J. Jia, Q. A. Chen, S. Wang, A. Rahmati, E. Fernandes, Z. M. Mao, and A. Prakash, "ContexIoT: Towards Providing Contextual Integrity to Appified IoT Platforms," in *NDSS*, 2017.

[15] E. Fernandes, J. Paupore, A. Rahmati, D. Simionato, M. Conti, and A. Prakash, "FlowFence: Practical Data Protection for Emerging IoT Application Frameworks," in *USENIX SEC*, 2016.

[16] A. F. A. Rahman, M. Daud, and M. Z. Mohamad, "Securing sensor to cloud ecosystem using internet of things (IoT) security framework," in *ACM ICC*, 2016.

[17] V. Sivaraman, H. H. Gharakheili, A. Vishwanath, R. Boreli, and O. Mehani, "Network-level security and privacy control for smart-home IoT devices." in *IEEE WiMob*, 2015.

[18] M. A. Jan, P. Nanda, X. He, Z. Tan, and R. P. Liu, "A robust authentication scheme for observing resources in the internet of things environment," in *IEEE TrustCom*, 2014.

[19] Z. Zheng and A. Reddy, "Towards improving data validity of cyber-physical systems through path redundancy," in *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security*. ACM, 2017, pp. 91–102.

[20] Z. Zheng and A. N. Reddy, "Safeguarding building automation networks: The-driven anomaly detector based on traffic analysis," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2017, pp. 1–11.

[21] Z. Zheng, S. Jin, R. Bettati, and A. N. Reddy, "Securing cyber-physical systems with adaptive commensurate response," in *2017 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2017, pp. 1–6.

[22] Symantec Corp., "Norton core," https://us.norton.com/core, 2017.

[23] Z. Alliance *et al.*, "Zigbee specification," 2006.

[24] Z.-W. Alliance, "Z-wave," 2015.

[25] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded Internet*. John Wiley & Sons, 2011, vol. 43.

[26] H.-P. D. Company, "Certain hp printers and hp digital senders, remote firmware update enabled by default," 2011.

[27] G. Lyon, "What is your operating system letting others do? nmap now!" https://nmap.org/, 2017.

[28] E. W. Zakir Durumeric and J. A. Halderman, "Zmap: Fast internet-wide scanning and its security applications," in *USENIX SEC*, 2013.

[29] DigitalBond, “Redpoint: Digital bond ics emulation tools,” https://github.com/digitalbond/Redpoint, 2014.

[30] S. Cheshire and M. Krochmal, “DNS-Based Service Discovery,” RFC 6763 (Proposed Standard), Internet Engineering Task Force, Feb. 2013. [Online]. Available: http://www.ietf.org/rfc/rfc6763.txt

[31] ——, “Multicast DNS,” RFC 6762 (Proposed Standard), Internet Engineering Task Force, Feb. 2013. [Online]. Available: http://www.ietf.org/rfc/rfc6762.txt

[32] I. Polycom, “Polycom web configuration utility user guide.”

[33] C. Botnet, “Internet census 2012,” 2012. [Online]. Available: http://census2012.sourceforge.net/paper.html

[34] L. Poettering, T. Lloyd, and S. Simons, “Avahi,” http://avahi.org/, 2017.

[35] B. Fouladi and S. Ghanoun, “Honey, I’m Home!!–Hacking Z-Wave Home Automation Systems, *BlackHat 2013*.”

[36] M. Focus, “Zenworks,” 2017. [Online]. Available: https://www.microfocus.com