Chapter 14

SECURITY AUDITING OF INTERNET OF THINGS DEVICES IN A SMART HOME

Suryadipta Majumdar, Daniel Bastos and Anoop Singhal

Abstract Attacks on the Internet of Things are increasing. Unfortunately, transparency and accountability that are paramount to securing Internet of Things devices are either missing or implemented in a questionable manner. Security auditing is a promising solution that has been applied with success in other domains. However, security auditing of Internet of Things devices is challenging because the high-level security recommendations provided by standards and best practices are not readily applicable to auditing low-level device data such as sensor readings, logs and configurations. Additionally, the heterogeneous nature of Internet of Things devices and their resource constraints increase the complexity of the auditing process. Therefore, enabling the security auditing of Internet of Things devices requires the definition of actionable security policies, collection and processing of audit data, and specification of appropriate audit procedures.

This chapter focuses on the security auditing of Internet of Things devices. It presents a methodology for extracting actionable security rules from existing security standards and best practices and conducting security audits of Internet of Things devices. The methodology is applied to devices in a smart home environment, and its efficiency and scalability are evaluated.

Keywords: Internet of Things, security auditing, formal verification

1. Introduction

The popularity of Internet of Things devices is growing rapidly. In fact, Statista [38] projects that more than 75.44 billion devices will be operational by 2025. However, current Internet of Things devices are easy targets of compromise due to implementation flaws and misconfigurations [1, 31, 43]. Verifying the flaws and misconfigurations, and

ensuring the accountability and transparency of the devices [1, 11] are essential for consumers and vendors.

Security auditing – verifying system states against a set of security rules – has become standard practice in enterprise security environments (see, e.g., Delloite [13], KPMG [23] and IBM [21]). Its advantages include supporting a range of security rules that cover system and network configurations, enabling examinations of the effects of events on system states, and delivering rigorous results via formal verification methods [25] as opposed to using other security solutions such as intrusion detection.

Security auditing has the potential to become a leading security measure against emerging threats to Internet of Things devices. However, security audits of Internet of Things devices are hindered by three principal challenges. First, existing security standards and best practices (e.g., [11, 14, 16, 17, 34]) provide high-level recommendations instead of guidance for conducting security audits of low-level system data in Internet of Things devices. Second, most Internet of Things devices cannot conduct the auditing process autonomously because they have limited resources to log audit data [43] and execute the formal verification tools used for auditing. Third, identifying the essential audit data to be collected for each security rule can be tedious.

Considerable research has focused on Internet of Things device security, including application monitoring, intrusion detection, device fingerprinting and access control. A few solutions provide *ad hoc* lists of rules for security tasks such as mobile application verification, network traffic inspection and access control [5, 6, 9, 43]. However, what is missing is a generic approach for automatically defining actionable rules that can be used to verify Internet of Things device security. Also missing is an auditing methodology designed for Internet of Things devices that can verify a range of security rules and present detailed audit reports with evidence of breaches. Unfortunately, existing auditing solutions for other environments such as the cloud (e.g., [10, 25, 26, 29]) are not applicable to Internet of Things devices because of their heterogeneous audit data sources, resource constraints and limited logging functionality.

This chapter proposes a security auditing methodology for Internet of Things devices. The methodology supports the extraction of actionable security rules from existing security standards and best practices, and enables security audits of Internet of Things devices. As a proof of concept, the security auditing methodology is applied to devices in a smart home environment, and its efficiency and scalability are evaluated (e.g., auditing 1,000 smart home networks within ten minutes).

2. Preliminaries

This section reviews key Internet of Things security standards and best practices, highlights the challenges to security auditing of Internet of Things devices, and defines the threat model.

2.1 Security Standards and Best Practices

This section reviews several security standards and best practices, namely NISTIR 8228 [11], NISTIR 8259 [17], OWASP IoT Security Guidance [34], ENISA Good Practices for Security of IoT [16] and the U.K. Government's Code of Practice for Consumer IoT Security [14]:

- **NISTIR 8228 [11]:** NIST's internal report NISTIR 8228 provides security and privacy recommendations for Internet of Things environments. It identifies four capabilities of Internet of Things devices: (i) transducer, (ii) data, (iii) interface and (iv) support. Additionally, it discusses a number of generic security recommendations for Internet of Things devices. In Section 4 below, security rules are specified based on the four device capabilities identified by NISTIR 8228.

- **NISTIR 8259 [17]:** Unlike NISTIR 8228, NIST's internal report NISTIR 8259 provides specific security recommendations for Internet of Things device vendors. It identifies six activities that vendors should consider during the pre-market and post-market phases. Additionally, it identifies five risk mitigation goals for consumers: (i) asset management, (ii) vulnerability management, (iii) access management, (iv) data protection and (v) incident detection.

- **OWASP IoT Security Guidance [34]:** OWASP's IoT Security Guidance includes recommendations for device vendors, application developers and consumers. The recommendations, which are divided into ten categories, are used in Section 4 to identify actionable security rules for Internet of Things devices.

- **ENISA Good Practices for Security of IoT [16]:** ENISA's Good Practices for Security of IoT focuses on the software development lifecycle. Its principal audiences are Internet of Things software developers, integrators, and platform and system engineers. It provides recommendations for preventing the introduction of vulnerabilities via the insecure implementation of software development lifecycle processes. The main contributions are an

analysis of security concerns in all phases of the software development lifecycle, detailed asset and threat taxonomies, and good practices that enhance the security and mapping of ENISA good practices to related standards, guidelines and schemes.

- **U.K. Government Code of Practice [14]:** The code of practice published by the Department for Digital, Culture, Media and Sport of the U.K. Government focuses on consumer Internet of Things device security. The code of practice includes recommendations for device vendors, application developers and service providers. The recommendations, which are divided into 13 categories, are used in Section 4 to identify actionable security rules for Internet of Things devices.

2.2 Security Auditing Challenges

Security auditing of Internet of Things devices faces four principal challenges:

- Existing security standards and best practices like NIST 8228 [11] and OWASP IoT Security Guidance [34] provide high-level recommendations to programmers and practitioners instead of guidance for conducting automated security operations such as monitoring and auditing Internet of Things devices. As a result, the recommendations cannot be used directly to create actionable rules for verifying Internet of Things device security. For instance, the OWASP high-level recommendation "ensure proper authentication/authorization mechanisms" has to be instantiated to an actionable rule such as "require a unique username and complex password of more than eight characters to access a smart door" in order to enable security verification.

- The recommendations listed in NISTIR 8228 [11], NISTIR 8259 [17], OWASP IoT Security Guidance [34], ENISA Good Practices for Security of IoT [16] and U.K. Government Code of Practice [14] significantly differ in their scopes, objectives and descriptions. Furthermore, some recommendations conflict with each other. A single comprehensive source for actionable rules does not exist. As a result, it is necessary to systematically analyze all the high-level recommendations, interpret them, resolve conflicts and then derive actionable security rules.

- System information such as the hardware specifications and software APIs of Internet of Things devices from different vendors are

published in different formats and use vendor-specific jargon (as reported in [15]). Therefore, it is essential to first normalize the vendor-specific materials and interpret the high-level recommendations specified in the standards and best practices in the context of the various implementations.

- Even after actionable rules are specified, Internet of Things devices are unable to conduct auditing processes autonomously because they have limited storage for logging audit data and computational power for executing formal verification tools. For example, a smart door would not be able to execute Sugar [40], a Boolean satisfiability solver.

This work attempts to overcome these challenges by deriving actionable rules for verifying Internet of Things device security and designing a security auditing methodology for Internet of Things devices.

2.3 Threat Model

The threat model assumes that Internet of Things devices have implementation flaws, misconfigurations and vulnerabilities that are exploited by malicious entities to violate security rules. A remote server or a local hub or gateway is required to perform security audits. The communications between Internet of Things devices and the verification server are secured by end-to-end encryption mechanisms. Privacy threats associated with data sharing by Internet of Things devices are beyond the scope of this research. However, they could be addressed in future work using a privacy-friendly auditing technique.

To keep the discussion concrete, the remainder of this work uses the context of a smart home to elaborate the concepts underlying the proposed security auditing methodology.

3. Security Auditing Methodology

Figure 1 provides an overview of the security auditing methodology for Internet of Things devices. The methodology involves three steps:

- **Step 1:** Build a knowledge base from Internet of Things device security standards and best practices, and details of Internet of Things device designs and implementations.
- **Step 2:** Translate the system knowledge and keywords to security rules.
- **Step 3:** Audit Internet of Things devices using the security rules.

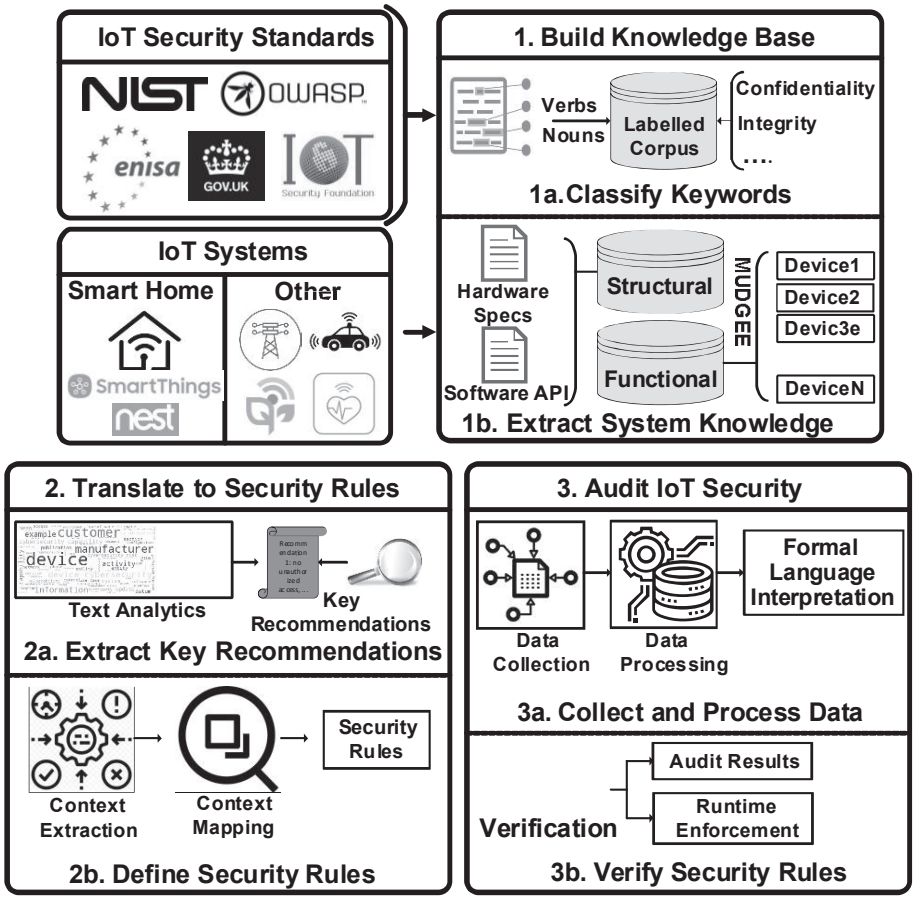

Figure 1. Security auditing methodology.

3.1 Step 1: Build a Knowledge Base

In order to audit the security of Internet of Things devices in a smart home, it is important to understand existing security standards and best practices as well as details of Internet of Things device designs and implementations. A knowledge base is created to codify this knowledge. Creating the knowledge base involves two steps:

- *Extract Keywords from the Guidelines Listed in Security Standards and Best Practices:* Keywords are extracted by parsing the contents of relevant sections in security standards and best practices documents. A corpus is then created with the relevant terms, mainly nouns and verbs, the two parts of speech that convey the

essence of the recommendations. Finally, the keywords in the corpus are classified based on standard security goals such as confidentiality, integrity and availability [3].

- *Collect System Knowledge Related to Internet of Things Device Designs and Implementations:* Structural knowledge about devices such as sensors and actuators is obtained from vendor-provided materials, including hardware specifications and software APIs as described in [15]. Functional knowledge such as network protocol usage is obtained by analyzing the network behavior of Internet of Things devices using MUDGEE [19].

3.2 Step 2: Translate to Security Rules

Having created the knowledge base, knowledge in the repository is translated to actionable security rules for Internet of Things devices. The translation process involves two steps:

- *Extract Recommendations from the Classified Keywords in the Security Standards and Best Practices:* The extraction of recommendations is automated using text analysis algorithms such as term frequency-inverse document frequency and sentiment analysis [39]. Next, the results are manual inspected to create a shortlist of the recommendations.

- *Define Actionable Security Rules by Instantiating the Recommendations with System Knowledge:* Actionable rules are defined by extracting the contexts of the recommendations using deep contextualized learning [35]. The context associated with each recommendation is mapped to related system knowledge and each recommendation is then mapped to a concrete security rule.

3.3 Step 3: Audit IoT Device Security

Having obtained a set of actionable security rules, the final objective is to conduct security audits of Internet of Things devices. Security auditing involves two steps:

- *Collect and Process Audit Data for the Security Rules Covering Internet of Things Devices:* The audit data source corresponding to each security rule for each Internet of Things device is identified and the logged data is collected. The collected data is preprocessed to a formal language format such as first-order logic.

- *Verify the Security Rules Using Formal Verification:* The first-order logic expressions are converted to the input format required

by a formal verification tool such as Sugar [40], a Boolean satisfiability solver. The verification results are interpreted. Finally the auditing results are presented to support various capabilities such as evidence analysis and security decision enforcement (e.g., allow or deny at runtime).

4. Auditing Smart Home Security

This section demonstrates the application of the proposed methodology in a use case involving the security auditing of Internet of Things devices in a smart home environment.

4.1 Security Rule Definition

This first step in the security auditing methodology is to define concrete security rules. To establish a bridge between the high-level guidelines in security standards and best practices and low-level system information pertaining to smart home devices, a list of concrete security rules from the standards and best practices and the literature were specified to formulate the security auditing problem.

Table 1 shows sample security auditing rules identified from relevant standards and best practices, smart home literature and real-world smart home implementations (e.g., Google Nest). The specific standards and best practices used were NIST 8228 [11], OWASP IoT Security Guidance [34] and U.K. Government Code of Practice [14].

The running example in this chapter will consider the following versions of rules *R1* and *R6* in Table 1:

- `R1:` Smart lock must not be in the unlocked state when other devices are in the vacation mode.
- `R6:` Photo or video capture are not allowed in a bathroom.

4.2 Data Collection

The next step is to collect the audit data to verify the security rules. Figure 2 shows sample data collected about Google Nest products. To obtain the data to verify the cross-device rules, the fields noted in the blueprints must be extracted from the event logs corresponding to each rule. Note that the required data was already collected by the devices, so no changes to the devices were necessary. In order to obtain data for auditing the rules pertaining to device capabilities (rules *R7* and *R8*), the types of all the installed devices were identified by analyzing the network traffic using the IoT Inspector [20] and IoTSense [7] tools.

Table 1. Sample security rules for conducting security audits of devices in a smart home.

		Standards and Best Practices		
Security Rule	**Description**	**NIST8228 [11]**	**OWASP [34]**	**UK Govt. [14]**
R1: No unauthorized access 1	Smart lock must be in the locked state when the smart plug is in the vacation mode [6, 43]	DE.AE-3, DE.CM-7	I6, I8, I9	6, 10
R2: No unauthorized access 2	Smart lock must be in the locked state when the smart light must be in the off state [6, 43]	DE.AE-3, DE.CM-7	I6, I8, I9	6, 10
R3: No lighting in an empty home	Smart light must be in the off state when the smart plug is in the vacation mode [6, 36]	DE.AE-3	I6, I7, I9	6, 7, 10
R4: No sound in the sleep mode	Speaker must be in the silent mode when the clock is in the sleep mode [6]	DE.AE-3	I6, I9	6, 7, 10
R5: Consistent climate control	Temperature cannot simultaneously correspond to cooling and heating [6]	DE.AE-3	I9	6, 7, 10
R6: No sensitive information leak	Device location must not be sensitive if the device has a camera [6]	DE.CM-7	I5	*6, 7, 10*
R7: Proper flood detection	Water flooding alarm is in the on state when the water level is higher than the threshold [6]	PR.AC-4	I6, I7, I9	7, 10
R8: Proper smoke detection	Smoke alarm state is in the on state when the smoke amount is higher than the threshold [42]	PR.AC-4	I6, I7, I9	7, 10

Table 2. Sample data collected about Google Nest products.

Device	Collected Data
Smoke Detector	id:1, device_id:vgUlapP6, locale:en-US, software_version:4.0, last_connection: 2018-12-31T23:59:59.000Z, battery_health:replace, co_alarm_state:ok, smoke_alarm_state:ok
Camera	id:1, device_id:2saNS6kQ, software_version:3.9, name:Front Door, is_streaming:false, web_url:`https://home.nest.com/cameras/2saNS6kQ`, is_online:false
Thermostat	id:4, device_id:vgUlapP6, locale:en-US, software_version:3.6, last_connection: 2019-1-05T15:59:59.000Z, ambient_temperature_f:70, ambient_temperature_c:38, humidity:75

Example 1: In the case of rule *R1*, event logs of the smart lock, smart plug and thermostat were collected. The following data was collected: smart_lock1.lock_state:locked, smart_lock2.lock_state:unlocked, smart_plug1.vacation_state:on, thermostat1.vacation_state:on.

In the case of rule *R6*, the sensing capabilities of all the bathroom devices were collected: showerhead:{bluetooth, microphone}, smart mirror:{camera, ambient light sensor, motion sensor}, water pebble:{bluetooth}.

4.3 Formal Language Translation

The next step is to convert the audit data and security rules to formal language expressions for the verification step. To this end, the security rules were expressed in the input format of the formal verification tool, i.e., as a constraint satisfaction problem for Sugar [40]. Programs were developed to translate the collected audit data for input to the selected verification tool.

Example 2: Rule `R1` is expressed as the predicate:

$$\forall \texttt{l} \in \texttt{Smartlock}, \forall \texttt{p} \in \texttt{Smartplug}, \forall \texttt{h} \in \texttt{Smarthome}, \forall \texttt{s} \in \texttt{Device-Status}$$
$$(\texttt{LockState(h, l, l.s)} \wedge \texttt{VacationState(h, p, p.s)}) \wedge (\texttt{IsOff(l.s)}) \longrightarrow \texttt{(IsOff(p.s))}$$

The corresponding constraint satisfaction problem (CSP) predicate is:

```
(and LockState(h,l,l.s) VacationState(h,p,p.s) (IsOff(l.s))
                                                (not (IsOff(p.s)))
```

The `LockState(h,l,s)` relation indicates the lock status `s` of smart lock `l` in smart home `h`. The `VacationState(h,p,s)` relation indicates the vacation status `s` of smart plug `p` in smart home `h`. The `IsOff(l.s)` relation indicates that the status of device `d` is `OFF`. For example, the `lockState(h1,l1,s1)` relation is `TRUE` when there exists a smart home `h1` with smart lock `l1` in state `s1`; otherwise the relation is `FALSE`. Similarly, the other relations in the CSP predicate are evaluated. Note that the CSP predicate becomes `TRUE` when rule *R1* is breached. Next, the relations are instantiated for each tuple associated with the audit data as: `(LockState (supports (h1,l1,OFF) (h1,l2,ON) ...))` and `(VacationState (supports (h1,p1,ON) ...))`.

4.4 Verification

Verification leverages formal techniques such as the Boolean satisfiability problem (SAT), declarative logic programming (Datalog) and satisfiability modulo theory (SMT), which have been used in several security applications [10, 24, 27, 28, 33]. These techniques are recognized for their expressivity of security rules, provable security and rigorous results. The verification tools are hosted at a server to overcome the resource constraints of Internet of Things devices. Note that no changes to the design or functionality of the verification tools are required.

Example 3: The CSP predicate of rule *R1* instantiated with the states of smart locks and smart plugs is executed in Sugar [40], a satisfiability problem solver.

4.5 Evidence Extraction

The final step is to interpret the outcome of the formal verification and prepare the audit reports. This effort is very specific to the verification tools that are used because most formal tools have their own output formats. However, in all cases, they provide rigorous results to identify evidence of any and all security rule violations.

Example 4: After verifying the CSP predicate of rule *R1* using the collected data, Sugar returns `SAT`, which indicates that the predicate is `TRUE` for the given data and that rule *R1* is violated. Furthermore, as evidence, Sugar identifies the tuple that caused the breach. Specifically, `LockState(h1,l1,OFF)` and `VacationState(h1,p1,ON)`, which mean

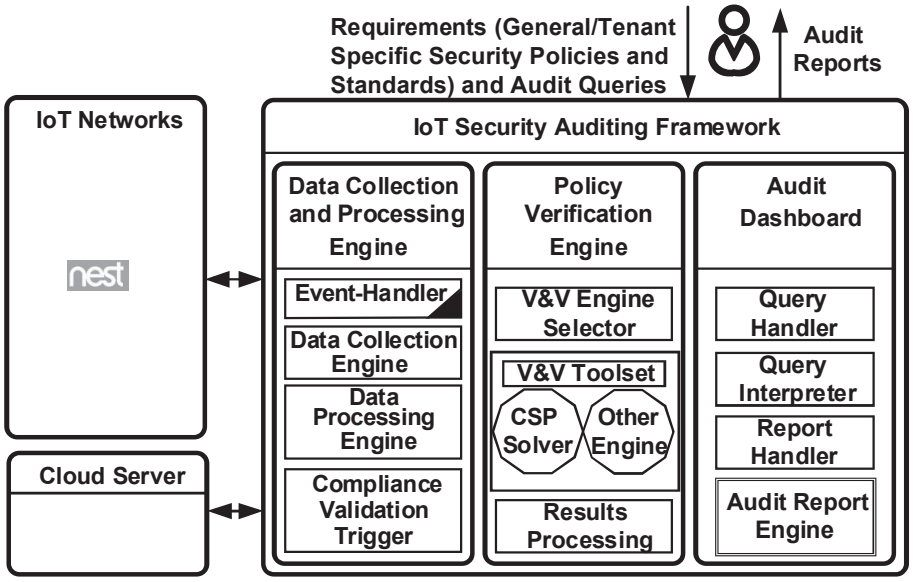

Figure 2. Security auditing framework.

that smart lock `l1` is unlocked when smart plug `p1` is in the vacation mode in the same smart home `h1`.

5. Security Auditing Framework

Figure 2 shows a high-level representation of the security auditing framework for Internet of Things devices. It primarily interacts with Internet of Things devices to collect audit data and with a cloud server to store audit data and delegate auditing computations (i.e., verification). It also interacts with users such as security experts and smart home owners to obtain the auditing requirements (security policies) and provide them with audit results in the form of reports. The framework is designed to work with a variety of Internet of Things networks. However, this work is restricted to smart home environments.

The framework has three main components: (i) data collection and processing engine, (ii) policy verification engine and (iii) audit dashboard:

- **Data Collection and Processing Engine:** The data collection and processing engine incorporates data collection and data processing sub-engines. The data collection engine collects the re-

quired audit data in the batch mode using smart home platforms such as Google Nest. Required audit data may also be collected from an Internet of Things hub and/or Internet of Things cloud server depending on the specific smart home implementation.

The data processing engine filters the collected data to retain the data needed to verify the security rules. It converts this data to a uniform format and subsequently translates it to formal language expressions. The final processing steps generate the code for verification and store it in the audit repository database for use by the policy verification engine. The code that is generated depends on the verification engine that is employed.

- **Policy Verification Engine:** The policy verification engine verifies security policies and identifies security violations. Upon receiving an audit request and/or updated inputs, the engine invokes the back-end verification and validation (V&V) algorithms. Formal methods are employed to express system models and audit policies, facilitating automated reasoning, which is more practical and effective than manual analysis.

  When a security audit policy is breached, supporting evidence is obtained from the output of the verification back-end. After the compliance validation is completed, the audit results and evidence are stored in the audit repository database and made accessible to the audit report engine. Depending on the security policies being verified, multiple formal verification engines may be incorporated.

- **Auditing Dashboard:** The auditing dashboard enables users to select various standards and best practices as well as specific security policies drawn from the standards and best practices. After an auditing request is submitted and processed, summarized results are presented via the auditing dashboard. Details of the violations are provided along with their supporting evidence. Audit reports are archived for stipulated periods of time.

6. Experiments and Results

This section presents the proof-of-concept experiments and their results.

6.1 Experimental Setup

The experiments employed physical and virtual testbeds. The physical testbed comprised 23 smart home products from several vendors,

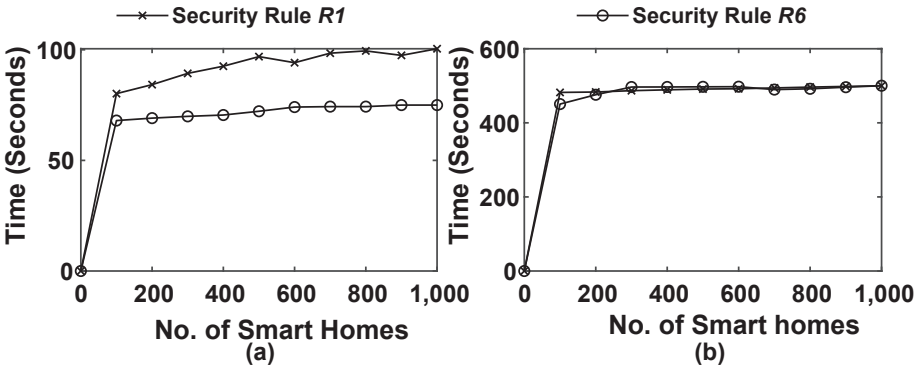

Figure 3. Time efficiency of the security auditing methodology.

15 Raspberry Pi single-board computers and 11 sensors for configuring the smart home devices. The virtual testbed simulated five smart home products – smart lock, smart plug, thermostat, camera and smoke detector.

The devices in the two testbeds produced outputs in standard formats (e.g., based on Google Nest API specifications [18]) and transmitted them to a cloud server for storage in a MySQL database. The two testbeds were connected to a virtual hub configured on a cloud server (Microsoft Azure IoT Hub [30]) or a physical hub. The hubs were connected to an auditing server with a 3.70 GHz Intel Core i7 Hexa core CPU and 32 GB memory.

The two testbeds were employed to generate datasets for the experiments. First, the physical testbed generated real data from the smart home products. The virtual testbed generated scaled-up data for up to 1,000 smart home networks based on the real data to help evaluate the scalability of the security auditing methodology. Each experiment was repeated 100 times and the average measurements were employed in the evaluations.

6.2 Experimental Results

The first set of experiments sought to measure the time efficiency of the security auditing methodology. Figure 3 shows the total times required to individually verify rule *R1* mandating that no unauthorized door be opened and rule *R6* mandating that no photo or video is allowed in a toilet for up to 1,000 smart homes. Figure 3(a) shows the total verifications times for five devices per smart home whereas Figure 3(b) shows the corresponding total verification times for 15 devices per smart home. The results reveal that the total times are not linear

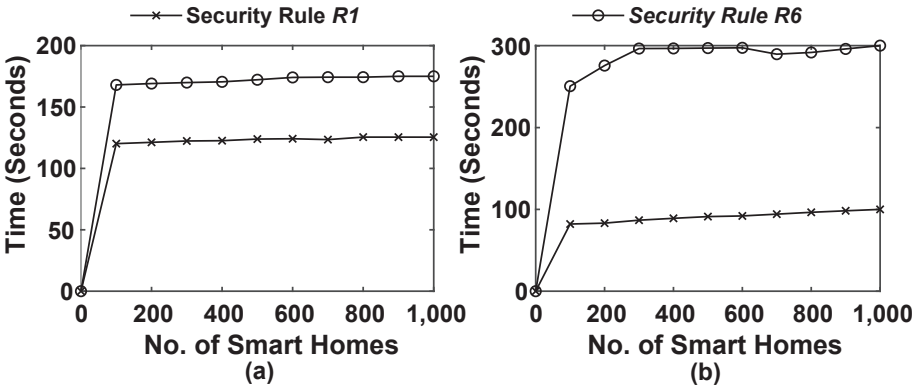

Figure 4. Data collection and processing time reqiurements.

functions of the number of smart homes to be verified. Additional results (not reported here due to space constraints) reveal that auditing additional security rules would not lead to significant increases in the total verification times.

Figure 4 shows the data collection and data processing times for rules *R1* and *R6* for up to 1,000 smart homes. Figure 4(a) shows the total times for five devices per smart home whereas Figure 4(b) shows the total times for 15 devices per smart home. Since the data collection and processing tasks are each performed only once for each audit request, the overheads are acceptable for auditing such large environments.

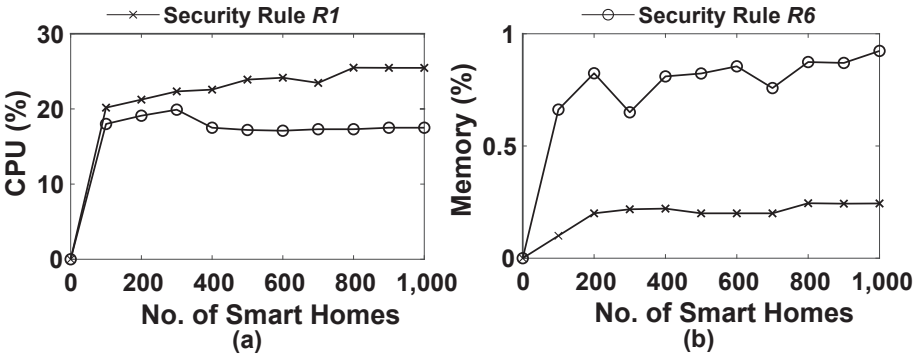

Figure 5. CPU and memory usage.

The second set of experiments sought to measure the CPU and memory usage of the security auditing methodology. Figure 5 shows the CPU and memory usage results for auditing rules *R1* and *R6* for up to 1,000

smart homes. Figure 5(a) shows the CPU usage results for auditing up to 1,000 smart homes with a fixed number of devices per smart home. The CPU usage remains within 26% for the largest dataset (1,000 smart homes). Furthermore, significant leveling in the CPU usage is seen for 300 or more smart homes. Note that other security rules show the same trends in CPU usage, which are expected because CPU usage is highly dependent on the amount of data collected.

Figure 5(b) shows the memory usage for auditing up to 1,000 smart homes with a fixed number of devices per smart home. Increases in memory usage are only observed beyond 800 smart homes. Investigation of the peak in memory usage for 200 homes revealed it to be the result of internal memory consumption by the Sugar verification tool. In any case, the highest memory usage over all the experiments is just 0.92%. Other security rules show the same trends in memory usage due to the high dependence of memory usage on the amount of data collected.

7. Discussion

Due to the expressiveness of the underlying formal verification method (i.e., SAT solver), the security auditing methodology can support a wide-range of security rules. Specifically, any rule that can be expressed as a constraint satisfaction problem would be supported. The main overhead in adding new rules comes from identifying the data required for auditing and locating their sources. This support can be provided by integrating additional tools in the security auditing framework.

The security auditing methodology and by extension the security auditing framework are designed to work with other Internet of Things networks, such as those encountered in the smart health, precision agriculture and autonomous vehicle environments. The main effort in adapting the methodology and framework to the new environments would involve finding data sources, collecting data and dealing with application-specific data formats. Most of the other steps are environment-agnostic and could be applied with minor modifications.

8. Related Work

The majority of research efforts in the area of Internet of Things security have focused on areas such as application monitoring, intrusion detection and access control [2, 6, 8, 12, 32, 37, 43, 44, 46]. This section attempts to show that, while the research described in this chapter differs from other Internet of Things security approaches in terms of scope and objectives, they complement each other.

Application monitoring techniques such as [22, 41] execute the source code of Internet of Things device applications to analyze the applications. For example, ContextIoT [22] and SmartAuth [41] are permission-based systems for monitoring individual applications. ProvThings [43] creates provenance graphs using security-critical APIs to support Internet of Things device forensics. Soteria [4] and IoTGuard [6] verify security and safety policies by performing static and dynamic code analyses, respectively. Unlike these source code analysis tools, the research described in this chapter presents a concrete solution for conducting security audits of the logs and configurations of Internet of Things devices.

Several security solutions have been developed for smart homes. For example, Zhang et al. [46] monitor isolation-related properties of Internet of Things devices using a virtual channel. Yang et al. [44] protect Internet of Things devices by hiding them in onion gateways. HoMonit [45] proposes a smart home monitoring system that employs deterministic finite automaton models for Internet of Things devices. However, none of these methods provide a security auditing solution for Internet of Things devices.

Modern security auditing methodologies can be categorized as retroactive, incremental or proactive in nature. A retroactive approach such as [25] audits a system after the fact, which means that it does not prevent irreversible damage such as denial of service or sensitive data loss. An incremental auditing approach such as [28, 10] audits the impacts of a change event; whereas such an approach overcomes the limitations of a retroactive approach, it causes significant delays in the response time. A proactive approach [10, 26, 29] computes a portion of the auditing effort in advance to keep runtime computations lightweight and, thus, provides practical response times, such as a few milliseconds to audit a mid-sized cloud [26]. However, unlike the security auditing methodology presented in this chapter, these auditing methods are not applicable to Internet of Things devices. This is primarily due to the computational and storage constraints of Internet of Things devices, their heterogeneity and limited logging functionality.

9. Conclusions

The proposed security auditing methodology for Internet of Things devices facilitates the extraction of actionable security rules from existing security standards and enables the automated auditing of the security rules using formal verification techniques and tools. Experiments conducted on physical and virtual testbeds with Internet of Things devices in smart home environments demonstrate the efficiency and scalability

of the security auditing methodology, including the ability to complete security audits of 1,000 smart home networks within ten minutes.

Future research will focus on injecting privacy into the security auditing process, which is required because security policy verification is currently conducted on a remote server. Research will also focus on automating the important task of converting raw information from security standards and best practices into actionable security rules, which is currently performed manually. Other research will consider simplifying the auditing workload using an incremental as opposed to a batch approach. It will also investigate applications of the security auditing methodology and by extension the security auditing framework to other Internet of Things networks, such as those encountered in smart health, precision agriculture and autonomous vehicle environments.

This chapter is not subject to copyright in the United States. Commercial products are identified in order to adequately specify certain procedures. In no case does such an identification imply a recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the identified products are necessarily the best available for the purpose.

References

[1] O. Alrawi, C. Lever, M. Antonakakis and F. Monrose, SoK: Security evaluation of home-based IoT deployments, *Proceedings of the IEEE Symposium on Security and Privacy*, pp. 1362–1380, 2019.

[2] M. Balliu, M. Merro and M. Pasqua, Securing cross-app interactions in IoT platforms, *Proceedings of the Thirty-Second IEEE Computer Security Foundations Symposium*, pp. 319–334, 2019.

[3] C. Bellman and P. van Oorschot, Best practices for IoT security: What does that even mean? arXiv: 2004.12179 (`arxiv.org/abs/2004.12179`, 2020.

[4] Z. Berkay Celik, P. McDaniel and G. Tan, Soteria: Automated IoT safety and security analysis, *Proceedings of the USENIX Annual Technical Conference*, pp. 147–158, 2018.

[5] Z. Berkay Celik, P. McDaniel, G. Tan, L. Babun and A. Selcuk Uluagac, Verifying Internet of Things safety and security in physical spaces, *IEEE Security and Privacy*, vol. 17(5), pp. 30–37, 2019.

[6] Z. Berkay Celik, G. Tan and P. McDaniel, IoTGuard: Dynamic enforcement of security and safety policy in commodity IoT, *Proceedings of the Network and Distributed Systems Security Symposium*, 2019.

[7] B. Bezawada, M. Bachani, J. Peterson, H. Shirazi, I. Ray and I. Ray, Behavioral fingerprinting of IoT devices, *Proceedings of the Workshop on Attacks and Solutions in Hardware Security*, pp. 41–50, 2018.

[8] S. Bhatt, F. Patwa and R. Sandhu, An access control framework for cloud-enabled wearable Internet of Things, *Proceedings of the Third IEEE International Conference on Collaboration and Internet Computing*, pp. 328–338, 2017.

[9] S. Birnbach, S. Eberz and I. Martinovic, Peeves: Physical event verification in smart homes, *Proceedings of the Twenty-Sixth ACM SIGSAC Conference on Computer and Communications Security*, pp. 1455–1467, 2019.

[10] S. Bleikertz, C. Vogel, T. Gross and S. Modersheim, Proactive security analysis of changes in virtualized infrastructure, *Proceedings of the Thirty-First Annual Computer Security Applications Conference*, pp. 51–60, 2015.

[11] K. Boeckl, M. Fagan, W. Fisher, N. Lefkovitz, K. Megas, E. Nadeau, D. Gabel O'Rourke, B. Piccarreta and K. Scarfone, Considerations for Managing Internet of Things (IoT) Cybersecurity and Privacy Risks, NISTIR 8228, National Institute of Standards and Technology, Gaithersburg, Maryland, 2019.

[12] J. Choi, H. Jeoung, J. Kim, Y. Ko, W. Jung, H. Kim and J. Kim, Detecting and identifying faulty IoT devices in smart homes with context extraction, *Proceedings of the Forty-Eighth Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 610–621, 2018.

[13] Deloitte, Cybersecurity and the Role of Internal Audit, New York (`www2.deloitte.com/us/en/pages/risk/articles/cybersecurity-internal-audit-role.html`), 2019.

[14] Department for Digital, Culture, Media and Sport, Code of Practice for Consumer IoT Security, Government of the United Kingdom, London, United Kingdom, 2018.

[15] A. Dolan, I. Ray and S. Majumdar, Proactively extracting IoT device capabilities: An application to smart homes, *Proceedings of the Thirty-Fourth Annual IFIP WG 11.3 Conference on Data and Applications Security and Privacy*, pp. 42–63, 2020.

[16] European Union Agency for Cybersecurity, Good Practices for Security of IoT: Secure Software Development Lifecycle, Athens, Greece (`www.enisa.europa.eu/publications/good-practices-for-security-of-iot-1`), 2019.

[17] M. Fagan, K. Megas, K. Scarfone and M. Smith, Foundational Cybersecurity Activities for IoT Device Manufacturers, NISTIR 8259. National Institute of Standards and Technology, Gaithersburg, Maryland, 2020.

[18] Google, Nest API Reference, Mountain View, California (`developers.nest.com/reference/api-overview`), 2019.

[19] A. Hamza, D. Ranathunga, H. Gharakheili, M. Roughan and V. Sivaraman, Clear as MUD: Generating, validating and applying IoT behavioral profiles, *Proceedings of the Workshop on IoT Security and Privacy*, pp. 8–14, 2018.

[20] D. Huang, N. Apthorpe, F. Li, G. Acar and N. Feamster, IoT Inspector: Crowdsourcing labeled network traffic from smart home devices at scale, *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 4(2), article no. 46, 2020.

[21] IBM, IBM Cloud Compliance Program, Armonk, New York (`www.ibm.com/cloud/compliance`), 2021.

[22] Y. Jia, Q. Chen, S. Wang, A. Rahmati, E. Fernandes, Z. Mao and A. Prakash, ContexIoT: Towards providing contextual integrity to appified IoT platforms, *Proceedings of the Network and Distributed Systems Security Symposium*, 2017.

[23] KPMG, Governance, Risk and Compliance Services, New York (`home.kpmg/xx/en/home/services/advisory/risk-consulting/internal-audit-risk.html`), 2021.

[24] T. Madi, Y. Jarraya, A. Alimohammadifar, S. Majumdar, Y. Wang, M. Pourzandi, L. Wang and M. Debbabi, ISOTOP: Auditing virtual network isolation across cloud layers in OpenStack, *ACM Transactions on Privacy and Security*, vol. 22(1), article no. 1, 2018.

[25] T. Madi, S. Majumdar, Y. Wang, Y. Jarraya, M. Pourzandi and L. Wang, Auditing security compliance of the virtualized infrastructure in the cloud: Application to OpenStack, *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, pp. 195–206, 2016.

[26] S. Majumdar, Y. Jarraya, M. Oqaily, A. Alimohammadifar, M. Pourzandi, L. Wang and M. Debbabi, LeaPS: Learning-based proactive security auditing for clouds, *Proceedings of the Twenty-Second European Symposium on Research in Computer Security, Part II*, pp. 265–285, 2017.

[27] S. Majumdar, T. Madi, Y. Wang, Y. Jarraya, M. Pourzandi, L. Wang and M. Debbabi, Security compliance auditing of identity and access management in the cloud: Application to OpenStack,

Proceedings of the Seventh IEEE International Conference on Cloud Computing Technology and Science, pp. 58–65, 2015.

[28] S. Majumdar, T. Madi, Y. Wang, Y. Jarraya, M. Pourzandi, L. Wang and M. Debbabi, User-level runtime security auditing for the cloud, *IEEE Transactions on Information Forensics and Security*, vol. 13(5), pp. 1185–1199, 2018.

[29] S. Majumdar, A. Tabiban, M. Mohammady, A. Oqaily, Y. Jarraya, M. Pourzandi, L. Wang and M. Debbabi, Proactivizer: Transforming existing verification tools into efficient solutions for runtime security enforcement, *Proceedings of the Twenty-Fourth European Symposium on Research in Computer Security, Part II*, pp. 239–262, 2019.

[30] Microsoft, Azure IoT Hub, Redmond, Washington (`azure.microsoft.com/en-ca/services/iot-hub`), 2019.

[31] S. Notra, M. Siddiqi, H. Gharakheili, V. Sivaraman and R. Boreli, An experimental study of security and privacy risks with emerging household appliances, *Proceedings of the IEEE Conference on Communications and Network Security*, pp. 79–84, 2014.

[32] T. O'Connor, R. Mohamed, M. Miettinen, W. Enck, B. Reaves and A. Sadeghi, Homesnitch: Behavior transparency and control for smart home IoT devices, *Proceedings of the Twelfth Conference on Security and Privacy in Wireless and Mobile Networks*, pp. 128–138, 2019.

[33] OpenStack Project, OpenStack Congress, Austin, Texas (`wiki.openstack.org/wiki/Congress`), 2015.

[34] OWASP Foundation, IoT Security Guidance, Bel Air, Maryland (`www.owasp.org/index.php/IoT_Security_Guidance`), 2019.

[35] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee and L. Zettlemoyer, Deep contextualized word representations, *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics, Volume 1 (Long Papers)*, pp. 2227–2237, 2018.

[36] E. Ronen and A. Shamir, Extended functionality attacks on IoT devices: The case of smart lights, *Proceedings of the IEEE European Symposium on Security and Privacy*, pp. 3–12, 2016.

[37] M. Serror, M. Henze, S. Hack, M. Schuba, and K. Wehrle, Towards in-network security for smart homes, *Proceedings of the Thirteenth International Conference on Availability, Reliability and Security*, article no. 18, 2018.

[38] Statista, Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025, New York (www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide), November 27, 2016.

[39] M. Taboada, J. Brooke, M. Tofiloski, K. Voll and M. Stede, Lexicon-based methods for sentiment analysis, *Computational Linguistics*, vol. 37(2), pp. 267–307, 2011.

[40] N. Tamura and M. Banbara, Sugar: A CSP to SAT translator based on order encoding, *Proceedings of the Second International CSP Solver Competition*, pp. 65–69, 2008.

[41] Y. Tian, N. Zhang, Y. Lin, X. Wang, B. Ur, X. Guo and P. Tague, SmartAuth: User-centered authorization for the Internet of Things, *Proceedings of the Twenty-Sixth USENIX Security Symposium*, pp. 361–378, 2017.

[42] P. Vervier and Y. Shen, Before toasters rise up: A view into the emerging IoT threat landscape, *Proceedings of the Twenty-First International Symposium on Research in Attacks, Intrusions and Defenses*, pp. 556–576, 2018.

[43] Q. Wang, W. Ul Hassan, A. Bates and C. Gunter, Fear and logging in the Internet of Things, *Proceedings of the Network and Distributed Systems Security Symposium*, 2018.

[44] L. Yang, C. Seasholtz, B. Luo and F. Li, Hide your hackable smart home from remote attacks: The multipath onion IoT gateways, *Proceedings of the Twenty-Third European Symposium on Research in Computer Security, Part I*, pp. 575–594, 2018.

[45] W. Zhang, Y. Meng, Y. Liu, X. Zhang, Y. Zhang and H. Zhu, HoMonit: Monitoring smart home apps from encrypted traffic, *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp. 1074–1088, 2018.

[46] Y. Zhang and J. Chen, Modeling virtual channel to enforce runtime properties for IoT services, *Proceedings of the Second International Conference on the Internet of Things, Data and Cloud Computing*, article no. 102, 2017.