Internet of Things and Cyber-Physical Systems 1 (2021) 1–7

Contents lists available at ScienceDirect

Internet of Things and Cyber-Physical Systems

journal homepage: www.keaipublishing.com/en/journals/internet-of-things-and-cyber-physical-systems

A framework for intelligent IoT firmware compliance testing

Mohan Krishna Kagita [a,*], Giridhar Reddy Bojja [a], Mohammed Kaosar [b]

[a] School of Computing and Mathematics, Charles Sturt University, Melbourne, Australia
[b] Discipline of Information Technology, Media and Communication, Murdoch University, Australia

ARTICLE INFO

Keywords:
IoT
IoT security
Compliance testing
Firmware vulnerabilities

ABSTRACT

The recent mass production and usage of the Internet of Things (IoT) have posed serious concerns due to the unavoidable security complications. The firmware of IoT systems is a critical component of IoT security. Although multiple organizations have released security guidelines, few IoT vendors are following these guidelines properly, either due to a lack of accountability or the availability of appropriate resources. Some tools for this purpose can use static, dynamic, or fuzzing techniques to test the security of IoT firmware, which may result in false positives or failure to discover vulnerabilities. Furthermore, the vast majority of resources are devoted to a single subject, such as networking protocols, web interfaces, or Internet of Things computer applications. This paper aims to present a novel method for conducting compliance testing and vulnerability evaluation on IoT system firmware, communication interfaces, and networking services using static and dynamic analysis. The proposed system detects a broad range of security bugs across a wide range of platforms and hardware architectures. To test and validate our prototype, we ran tests on 4300 firmware images and discovered 13,000+ compliance issues. This work, we believe, will be the first step toward developing a reliable automated compliance testing framework for the IoT manufacturing industry and other stakeholders.

1. Introduction

The variety of IoT devices and their mass production ensures their use in daily life. IoT has a huge effect on human life because it supports a wide variety of applications. IoT has increased machine-to-machine (M2M) and human-to-machine (H2M) connectivity by orders of magnitude [1]. However, with the benefits of IoT come security risks and limitations. As a relatively new sector, the IoT manufacturing industries and consumers generally lack the necessary skills to properly develop and operate these devices. Another problem that companies face is the limited time-to-market given to developers, integrators, and designers, which results in insufficient testing. As a result, not only has IoT spawned a slew of attack vectors, but it has also effectively become a playground for IoT hackers. IoT attackers use these devices to pilot attacks on other networked networks due to a lack of security enforcement [2].

The firmware is an essential component in the security environment of IoT devices. An IoT device's firmware contains the majority of the software stack, including the kernel, filesystem, system/third-party libraries, programs, and even the bootloader [3–5]. Aside from the negligence that IoT firmware protection suffers from, other obstacles such as collection and reverse engineering of firmware make vulnerability analysis of an IoT firmware a time-consuming job. Furthermore, the IoT paradigm's lack of standardization makes firmware vulnerability an unavoidable guest [2,6]. Despite the fact that some governments have recognized the problem by requiring manufacturers to follow a code of practice [7], and the research community is contributing by supporting vulnerability analysis and, as a result, closing loopholes [8–12], the main governing body is the IoT manufacturers, vendors, developers, service providers, Original Equipment Manufacturers (OEM), and Original Design Manufacturers (ODM) (ODM). Unfortunately, these sectors have consistently shown a lack of respect for security [13]. Despite the fact that numerous firmware vulnerability analysis solutions have been created, almost all of them concentrate on improving analysis methods and rarely on integrating them to create an easy deploy/use a system for auditing the firmware, network services, and communication/web interfaces. The primary motivation for this research is to bridge this gap so that various stakeholders in the IoT ecosystem can test the compliance of their devices and audit various firmware components. As a result, IoT devices can hit end-users with a sufficient level of protection.

Static, dynamic, fuzzing and hybrid approaches are the most

*Corresponding author.
E-mail addresses: m.kagita@cqumail.com (M.K. Kagita), Mohammed.Kaosar@murdoch.edu.au (M. Kaosar).

https://doi.org/10.1016/j.iotcps.2021.07.001
Received 16 May 2021; Received in revised form 11 July 2021; Accepted 21 July 2021
Available online 4 August 2021

commonly used for analysis, with fuzzing techniques being used most recently. The majority of current solutions, on the other hand, are based on static or dynamic analysis. Static process investigation is carried out by evaluating code structure without conducting any execution. Naturally, static code decompilation and debugging are needed to properly analyze the code. Dynamic analysis, on the other hand, is carried out by running code or checking a live system to analyze its actions. It is critical to understand that both methods are needed for rigorous firmware security testing. This is due to the fact that both techniques have their own set of limitations. Static analysis suffers from issues such as false positives [14–16] and necessitates extensive pre-processing such as firmware selection, unpacking, decompilation, debugging, and other reverse engineering techniques [9]. When the firmware picture is obfuscated and/or encrypted, static analysisis often impossible [17,18]. Dynamic research, on the other hand, is constrained by the lack of large-scale IoT devices and the lack of support for automated virtualization tools such as [15].

Fundamentally, compliance testing and vulnerability evaluation in IoT can be orchestrated using various modules such as the ones mentioned below:

- Reverse engineering module
- Analysis module
- Storage module
- Data processing/Intelligence module
- Communication module
- Reporting module (s)

Table 1 lists the characteristics of an optimal solution. However, several emerging solutions place too much emphasis on the research module, ignoring the importance of other modules in creating a comprehensive system. Despite this, there are numerous solutions based on static [19–24] or dynamic [9,16,17,25–27] analysis techniques. Since there are too many IoT firmware bugs, a single research approach can not detect them all. Static analysis often detects vulnerabilities such as hard-coded passwords, memory protection bugs, obsolete modules, and so on. Dynamic research, on the other hand, can be used to evaluate open ports, unstable network networks, insecure firmware upgrade processes, and so on. There aren't many solutions that combine static and dynamic analysis. Even if some solutions exist, their range of vulnerability detection is very limited [28]. Other options, such as [29], are very difficult to use. The majority of current research focuses on one of many attack vectors, such as firmware, networking protocols, and communication interface vulnerabilities. As a result, we suggest a system that addresses a broader range of vulnerabilities found in IoT firmware, networking protocols, and communication interfaces. Furthermore, our architecture is intended to be modular, configurable, and extendable, allowing various stakeholders in the IoT ecosystem to audit their devices. Finally, the platform has an easy-to-use web interface that will allow both a Quality Assurance (QA) team member and an average desktop user to use it.

Look at the device's vulnerabilities. Through this system, the IoT manufacturing industry and geeky end-users will verify the compliance of their IoT devices with the OWASP guidelines [30].

Table 1
Qualities of an ideal framework for firmware analysis.

Feature	Domain
Open source solution	Standardization
Autonomous solution with little/no human intervention requirement	Intelligence
Evolve with the time	Intelligence
Support for multiple platforms and architectures	Reverse engineering
Audit heterogeneous devices with different interfaces	Reverse Engineering
Scalable solution	Analysis, Reverse Engineering
Discover existing as well as new vulnerabilities	Analysis
Wider vulnerability coverage	Analysis
Fast and robust	Analysis
Combination of multiple analysis techniques	Analysis

1.1. Our involvement

- We have created a novel, adaptable, and extensible platform for IoT stakeholders that identifies vulnerabilities through static and dynamic analysis.
- The proposed system validates compliance with the OWASP protection guidelines. In addition, the identified vulnerabilities are validated and rated for criticality.
- We tested our system on a variety of firmware images from various vendors and device types and discovered a total of 13,375 compliance issues out of over 4300 firmware images.
- For testing purposes, we have made an alpha version of our static analysis module available online 1.

1.2. Paper organization

The proposed structure and its environmental configuration are described in Section 2 of this paper. Section 3 assesses the framework's success using a variety of criteria. Section 4 addresses the framework's shortcomings. Section 5 concludes and discusses potential future jobs.

2. Proposed framework

2.1. Approach and overview

Given that static or dynamic analysis alone cannot detect numerous vulnerabilities, we propose a solution for IoT manufacturers that employs both analysis techniques for vulnerability evaluation and enforcement testing of IoT firmware. Fig. 1 depicts the framework's architecture. The proposed architecture is tailored for vulnerability detection in such a way that various vulnerabilities are identified using an analytical approach that is best suited for its detection. For example, static analysis is useful for determining the kernel version number, but the dynamic analysis is useful for detecting network service vulnerabilities. This is the path we took in order to save time and money.

The framework's operation is straightforward. The firmware binary images are initially collected and forwarded to the pre-processing module for unpacking. The firmware components such as the bootloader, kernel, filesystem(s), and application are extracted during this point. The metadata collection module not only collects appropriate firmware information but also communicates with the database to determine if the image under investigation has previously been scanned using a fast hash comparison. If any firmware images are found to be duplicated in the database, the scanning will be halted and an already stored report will be shown as output. The research engine is augmented by a validation module that verifies vulnerabilities. The database stores all bugs, metadata, and hashes of firmware images for future reference. Following the completion of the study, the final report is compiled, which includes the vulnerabilities, their criticality, and a list of files associated with such vulnerabilities. A significant feature of the system is that it uses hard-coded data from previously scanned firmware images (such as IP addresses, URLs, and so on) to retrieve relevant firmware images. This greatly expands the firmware repository for future analysis.

2.2. This section goes through the static and dynamic analysis modules in depth. While static and dynamic analysis can be performed concurrently, our framework recommends performing static analysis first. This is

due to the fact that some insights from the static analysis can be useful in performing effective dynamic analysis. For example, during static analysis, it might be possible to retrieve the web application's login credentials, which can then be used during dynamic analysis to uncover additional vulnerabilities. One of the framework's primary goals is to

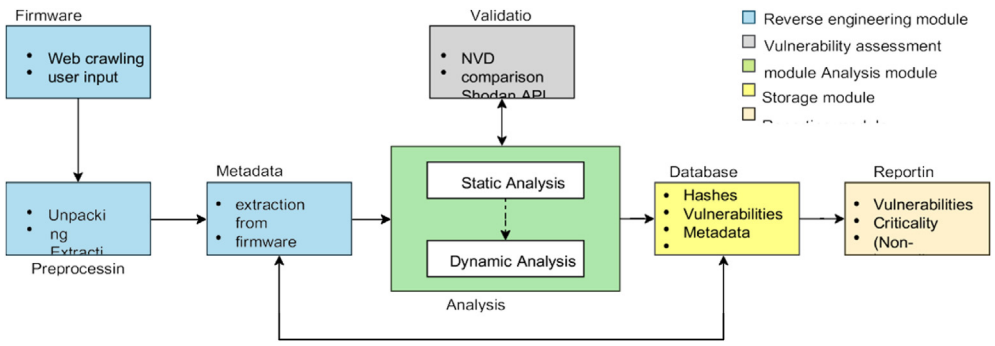

Fig. 1. Overview of the analysis framework.

verify the firmware image's security compliance with OWASP guidelines. The analysis engine tests the protection using the following techniques to perform this test:

A. Files with high-security risks

A typical firmware contains a number of components that can be checked in various ways for security.

The following is a list of files and the security checks that were conducted on them:

- ((X.509)) Certificates of Achievement: SSL or SSH security certificates with private keys are usually encapsulated in firmware with no safeguards. These certificates can be easily retrieved using static analysis. The certificate specifics are printed in the final report by our framework.
- Outdated components: Many software components, such as the kernel, bootloader, system libraries, binaries packages such as busy box, and so on, are considered to be vulnerable in older versions. To efficiently identify vulnerable areas, our system selectively scans for security-sensitive obsolete components.
- Hard-coded credentials: Manufacturers, integrators, and/or developers often leave hard-coded information such as IP addresses, email addresses, domain/hostnames, comments, and application-specific details for debugging purposes, which attackers may exploit.
- Password files: password and shadow are the main password files. To break passwords, our system employs the reverse hashing strategy. We've found that cracking such passwords typically results in IoT device hijacking because the same keys are often used for the device's Telnet interface.
- Configuration files: Different configuration files such as. cfg,.ini, and. conf are checked for network or device interface credentials.

B .Verification

For vulnerability evaluation, two primary validation methods are used:

IoT vendors sometimes use the same credential across a range of identical IoT devices [19]. The system identifies a number of possible target devices that are using the same credential. This goal is achieved by using the Shodan API to perform a reverse certificate search [31].

The NVD database contains known vulnerabilities for each outdated software component [32]. The framework scans the NVD database for vulnerabilities relevant to each extracted software component version in order to compile a comprehensive report.

C. ELF review

The ELF analysis is a useful feature of our system. Several tests are run to look for security hardening features in any firmware image, such as:

- Support for Position Independent Executable (PIE)
- Support for Relocation Read-Only (RELRO)
- Use of Stack Canary
- Features that have been removed

Furthermore, ELF files are scanned for potentially harmful API calls. In the final report, the user is given a comparison of total function calls versus risky API calls.

D. Check open ports and utilities

Searching for open network ports is not only easy, but it can also provide the bad guys with a large attack surface. The system highlights a list of open ports as well as the vulnerable services operating on those ports. Out-of-date network services are detected and recorded to the customer.

E. Unsafe web interface

Many IoT devices have web interfaces that enable users to run user applications or configure the system. Despite the fact that the network is a vast domain in and of itself, the proposed architecture conducts a number of security checks after searching for a login page. To log into the web interface, credentials acquired during previous stages of review are used.

F. Vulnerabilities in network protocols

IoT systems communicate using a variety of networking protocols and access technologies. The most widely used networking protocols are Telnet and SSH. To retrieve passwords for Telnet or SSH protocols, our system searches for all configuration and password data. Furthermore, many IoT devices use default or weak usernames and passwords [2,19]. Wireless networking technologies such as Bluetooth Low Energy (BLE) and Wifi are used by a large number of IoT users, with some older devices using Zigbee. Our system sniffs and analyzes protection strength through a series of checks.

Fig. 2 depicts the coverage of security vulnerabilities as well as the type of analysis approach used to perform the checks. Furthermore, it maps each test to the OWASP guidelines to ensure that the device's firmware is safe.

2.3. *Environmental preparation*

Fig. 3 depicts the framework's environmental configuration. Two methods are used to collect firmware files.

(1) Feedback from a user through a web interface. (2) The system crawled the internet on its own. In the first example, the metadata of a firmware image can be entered by the user, while in the second case, the application scrapes web pages referring to a firmware image to obtain the metadata. Metadata usually includes the vendor name, IoT system form, firmware version, and so on. The computer server is the centre where the

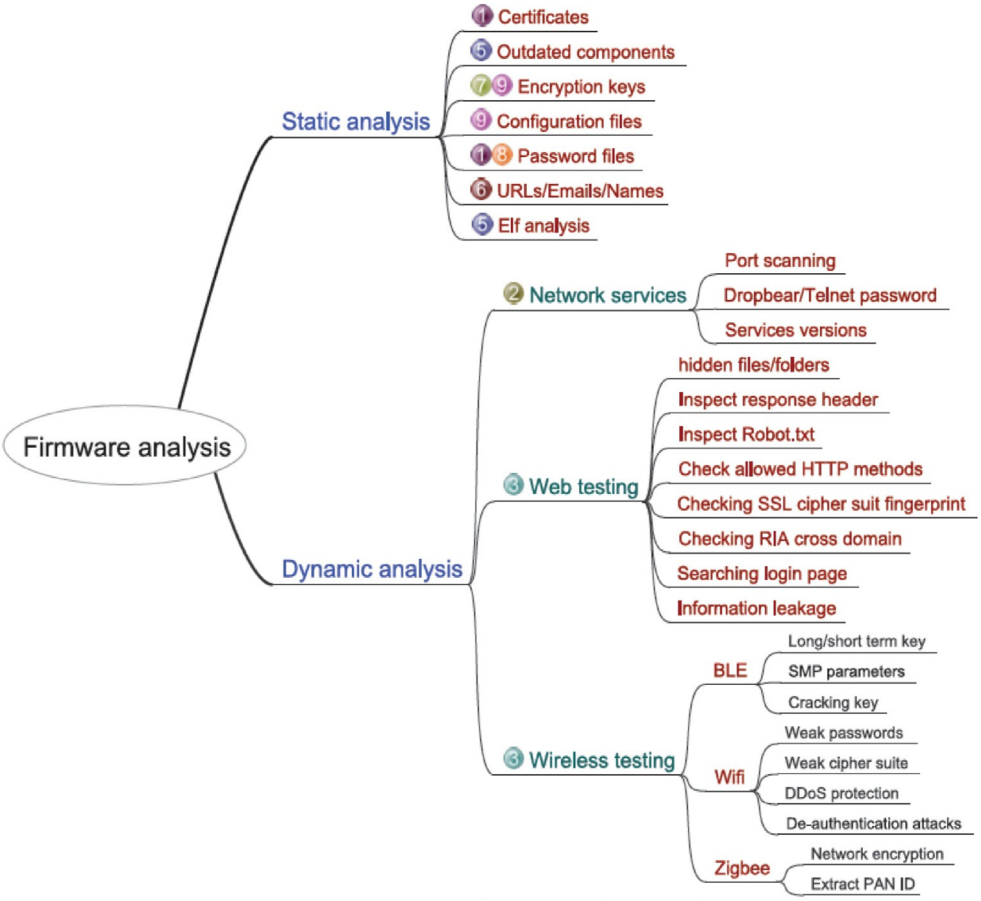

Fig. 2. Framework's coverage of OWASP guidelinesa
Static and dynamic analysis
(b) OWASP top 10 IoT guidelines.

vulnerability tests are run. The server is powered by an Intel Xeon 4208 processor with 16 threads and 32 GB of RAM. The firmware images take up more than 200 GB of storage space in one of the two databases. The computer server is connected to both databases. While one database collects firmware added by the user or crawled from the site, the other is used to store analysis and vulnerability data. The computer server is linked to the Internet in order to perform run-time comparisons of software product versions with the NVD database and to scan for devices with duplicate certificates through the Shodan API. The computer server is linked to a wireless module that supports BLE, Wi-Fi, and Zigbee communication interfaces to speak to heterogeneous IoT devices based on the communication technologies they support for dynamic analysis. Fig. 3 depicts a basic setup. Aside from that, web research is carried out directly by searching for the web interface of IoT devices.

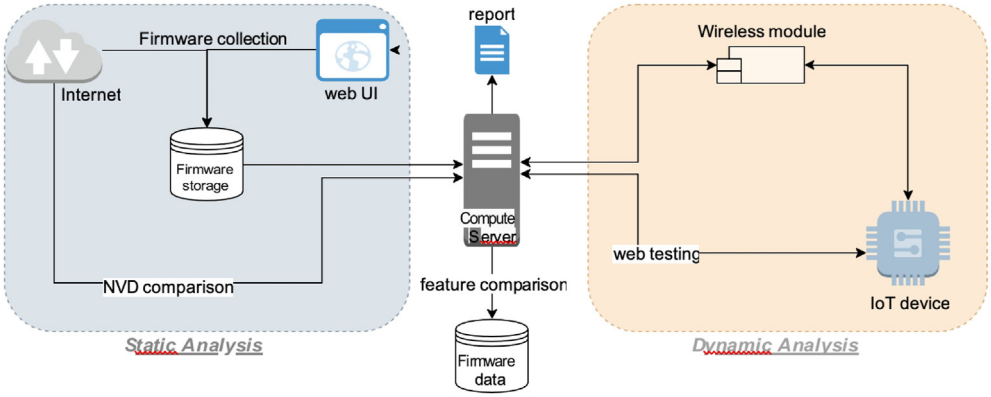

Fig. 3. Environment setup.

Instead of starting from scratch, the following open-source tools were used to build the framework:

- Binwalk2: This program is used to unpack firmware binary files.
- Crackle 4: Used for Bluetooth Low Energy (BLE) encryption cracking.
- John the Ripper (JTR) 3: Used for password cracking.

Python is the most common programming language because of its comprehensive library support.

3. Evaluation

We rate our framework based on its speed, supported platforms and architectures, and vulnerability detection capability. Furthermore, rather than the amount of firmware images tested, the focus was placed on the selection of a diverse set of firmware images. As a result, the framework's research engine was tested on a variety of datasets. Table 2 provides a classification of the total number of firmware files. Miscellaneous IoT devices include smart switches, wifi extenders, webcams, radio antennas, smart connectors, and smart motion cameras, among others. Over 4300 firmware images were gathered from 76 separate vendor repositories.

It was not possible to perform complex analysis on multiple devices due to the limited number of IoT devices available. However, using our platform, one can perform (selective) dynamic analysis on their IoT computers. Nonetheless, we subjected the collected firmware images to rigorous static processing.

3.1. Output at runtime

Our framework takes on average 10 s to statically analyze a firmware image, depending on the size of the firmware. This includes the time spent bruteforcing default passwords (s). The time it takes to crack a password, on the other hand, is adjustable. If a firmware image was encrypted, the static analysis was skipped in some instances. Though dynamic analysis is faster, its speed is dependent on the previous static analysis that was performed as some input.

Certain dynamic analysis compliance checks can be avoided by using static analysis. Overall, when compared to other solutions that use techniques like fuzzing, our framework can perform analysis relatively quickly. Because of our framework's combination of static and dynamic analysis, time-consuming tasks such as password brute-forcing and dictionary attacks over interfaces such as Telnet and web applications are avoided entirely.

Table 2
Classification of IoT firmware images.

Sr.	Firm ware type	Distinct vendors	Im ages
1	Miscellaneous	22	1993
2	Routers/Access Points	14	1247
3	Network/IP Camera	19	438
4	DVR	2	283
5	Smart Switch	1	91
6	Switches	3	60
7	Wi-Fi Camera	1	52
8	IP phone	1	34
9	NVR	4	31
10	Smart Bulb	1	31
11	Intercom/Doorbell	4	25
12	Smart TV	2	22

3.2. Assistance

Because of the heterogeneous existence of IoT, a testing system must support various architectures and platforms. This is why we put a premium on support for various architectures and platforms. At the same time, because of its greater market share, we concentrated more on Linux platforms. As a result, our architecture will be able to thoroughly examine the vast majority of produced devices. Table 3 compares the platforms and architectures provided by our framework to those of other well-known solutions. There is also support for static and dynamic analysis. Despite the fact that [28] offers better support and employs both static and dynamic analysis, it only detects memory corruption errors.

3.3. Identification of vulnerabilities

Table 4 compares the vulnerabilities covered by our approach to those covered by other well-known solutions. The proposed system was able to find more than 13,300 compliance issues with the OWASP IoT guidelines after checking the 4300+ firmware files. The following were some of the most noticeable patterns:

- The majority of the firmware images contained hard-coded data such as addresses and URLs. • The majority of the firmware contained standard username and password combinations.
- Almost all firmware images examined by our ELF review procedure lacked security hardening functionality.
- Many firmware images included certificates, the details from which could be easily extracted; and • Several bugs that occurred in one version of firmware persisted in subsequent versions.

We tested our prototype on devices such as Bluetooth hands-free, D-Link routers, and ZigBee-enabled smart home devices and discovered that the majority of them were non-compliant with the OWASP guidelines. The Bluetooth dynamic analysis tested important parameters of the Security Management Protocols (SMP) that could be used for attacks such as fake device formation and key cracking of Long/Short term keys. The Wi-Fi module validated the enforcement of critical security parameters

Table 3
Proposed framework's comparison of architecture and platform support.

			Supported Architectures				Supported Platforms		
Framework	Static	Dynamic	ARMS	MIPS	X86	Linux	Windows	RTOS	BareMetal
Our Framework	x	x	x	x	x	x		x	x
[19]	x		x	x		x			
[9]	x	x	x	x	x	x		x	
[17]		x	x	x	x	x	x	x	x
[22]	x		x	x		x			
[23]	x		x	x	x	x			
[33]		x	x	x		x			
[28]	x	x	x	x	x	x	x	x	x

Table 4
Firmware vulnerabilities detection comparison.

Vulnerabilities	Framework/Reference							
	Our Framework	[22]	[23]	[19]	[9]	[17]	[33]	[28]
Weak Passwords	x	x	x	x			x	x
Hard-Coded Credentials	x	x	x	x	x			x
Misconfiguration			x	x	x			
Outdated Components	x		x	x	x		x	
SSL Certificates	x		x	x			x	
No/Weak encryption	x		x					
Insecure web Interface	x			x	x		x	
Information Leakage	x	x			x			x
Memory Corruption		x	x			x	x	x
Backdoors	x	x	x	x				x
Insecure bootloader	x							
Vulnerable network services	x		x			x	x	x

such as encryption schemes, DDoS authentication, and weak passwords, among others. Furthermore, the Zigbee module can assess the encryption of Personal Area Network (PAN) IDs and remove them, which can be used to insert malicious traffic into the network.

4. Limitation

Testing stable firmware updates in IoT devices is one of the main attack vectors. However, including that test would significantly expand the project's reach. We believe that checking certain OWASP guidelines warrants its own structure. Another guideline that our system does not answer is IoT device hardware security. Despite the fact that we discussed its importance in developing an ideal solution in Table 1, it will involve a hardware module that has all of the communication interfaces to analyze the system. Finally, the proposed architecture is incapable of detecting zero-day vulnerabilities.

5. Conclusion and future works

The firmware of IoT devices is a significant attack vector. Until selling their products, the IoT manufacturing industry must conduct compliance testing. Our proposed framework is intended for IoT stakeholders to verify compliance with OWASP guidelines and provide an easy-to-use interface for firmware image vulnerability assessment. The proposed architecture offered a broad range of vulnerability detection in a limited period of time. Despite its limitations, the system can be viewed as a first step toward a flexible automated compliance testing and vulnerability assessment solution for the security quality assurance of IoT devices. Dealing with the framework's weaknesses would be a priority in the future. The system is currently being reviewed with a prototype. We will, however, create an API for it to be used as a tool by developers and researchers. Another important task is to thoroughly examine the results obtained from firmware images in order to fine-tune the system, especially the analysis engine. Furthermore, we will dig deeper into the data to determine the types of vulnerabilities found in various IoT system types, platforms, and architectures. We believe that this study will help the research community, as well as IoT manufacturers and service providers, understand the root causes of IoT security issues.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Rahul Singh, Bharat Bhushan, Ashi Tyagi, Deep learning framework for cybersecurity: framework, applications, and future research trends, 2021, https://doi.org/10.1007/978-981-33-4367-2_80.
[2] Abhik Chaudhuri, IoT Cyber Security—A Discourse on the Human Dimension, 2018, https://doi.org/10.1201/9781315200644-11.
[3] Arunan Sivanathan, Habibi Gharakheili, Hassan, Vijay Sivaraman, Managing IoT cyber-security using programmable telemetry and machine learning, in: IEEE Transactions on Network and Service Management, 2020, https://doi.org/10.1109/TNSM.2020.2971213, 1-1.
[4] Soraya Sinche, Pablo Hidalgo, José Fernandes, Duarte Raposo, Jorge Sá Silva, André Rodrigues, Ngombo Armando, Fernando Boavida, Analysis of student academic performance using human-in-the-loop cyber-physical systems, Tele.com (NY) 1 (2020) 18–31, https://doi.org/10.3390/telecom1010003.
[5] Kagita, Mohan Krishna, Madda Varalakshmi, A detailed study of security and privacy of internet of Things (IoT), Int. J. Distributed Sens. Netw. 9 (2020).
[6] Mohan Krishna Kagita, Security and Privacy Issues for Business Intelligence in lOT, 2019, pp. 206–212, https://doi.org/10.1109/ICGS3.2019.8688023.
[7] Morteza Pour, Dylan Watson, Bou-Harb, Elias, Sanitizing the IoT Cyber Security Posture: an Operational CTI Feed Backed up by Internet Measurements, 2021.
[8] Navod Thilakaratahne, Mohan Krishna Kagita, Thippa Gadekallu, The role of the internet of Things in health care: a systematic and comprehensive study, International Journal of Engineering and Management Research 10 (2020) 145–159, https://doi.org/10.31033/ijemr.10.4.22.
[9] A. Sivasangari, Ajitha Ponnupillai, Brumancia Prince, L. Sujihelen, Rajesh Ganesan, Data security and privacy functions in fog computing for healthcare 4 (2021), https://doi.org/10.1007/978-3-030-46197-3_14, 0.
[10] Mohammad Wazid, Das, Ashok Kumar, Sachin Shetty, Prosanta Gope, Joel Rodrigues, Security in 5G-enabled internet of Things communication: issues, challenges and future research roadmap, in: IEEE Access, 2020, https://doi.org/10.1109/ACCESS.2020.3047895, 1-1.
[11] Oliver Nock, Jonathan Starkey, Angelopoulos, Constantinos Marios, Addressing the security gap in IoT: towards an IoT cyber range, Sensors 20 (2020), https://doi.org/10.3390/s20185439.

[12] Mohan Krishna Kagita, Navod Thilakaratahne, Thippa Gadekallu, Praveen Reddy, A Review on Security and Privacy of Internet of Medical Things, 2020.
[13] Oliver Nock, Jonathan Starkey, Angelopoulos, Constantinos Marios, Addressing the security gap in IoT: towards an IoT cyber range, Sensors 20 (2020), https://doi.org/10.3390/s20185439.
[14] Weishan Zhang, Qinghua Lu, Qiuyu Yu, Zhaotong Li, Yue Liu, Lo, Sin Kit & Chen, Shiping & Xu, Xiwei & Zhu, Liming, Blockchain-based federated learning for device failure detection in industrial IoT, in: IEEE Internet of Things Journal, 2020, https://doi.org/10.1109/JIOT.2020.3032544.
[15] Navod Thilakaratahne, Mohan Krishna Kagita, Thippa Gadekallu, Praveen Reddy, The Adoption of ICT Powered Healthcare Technologies towards Managing Global Pandemics, 2020.
[16] A K M Bahalul Haque, Bharat Bhushan, Security Attacks and Countermeasures in Wireless Sensor Network, 2021, https://doi.org/10.1201/9781003107521-2.
[17] Mohan Krishna Kagita, Li Xiujuan, Machine learning techniques for multi-media communications in business marketing, J. Mult.-Valued Log. Soft Comput. 36 (2021) 151–167.
[18] Mohan Krishna Kagita, Navod Thilakaratahne, Dharmendra Rajput, Surekha Lanka, A Detail Study of Security and Privacy Issues of Internet of Things, 2020.
[19] Nayanika Shukla, Bharat Bhushan, Attacks, Vulnerabilities, and Blockchain-Based Countermeasures in Internet of Things (IoT) Systems, 2021, https://doi.org/10.1201/9781003133391-14.
[20] Mohan Krishna Kagita, Navod Thilakaratahne, Thippa Gadekallu, Praveen Reddy, Saurabh Singh, A Review on Cyber Crimes on the Internet of Things, 2020.
[21] Mohan Krishna Kagita, Vijay Velaga, AN updated new security architecture for IOT network based ON software-defined networking (SDN), International Journal of Innovative Research in Computer and Communication Engineering 5 (2018), https://doi.org/10.26562/IRJCS.2018.FBCS10087.
[22] Nikhil Sharma, Ila Kaushik, Vikash Agarwal, Bharat Bhushan, Aditya Khamparia, Attacks and Security Measures in Wireless Sensor Network, 2021, https://doi.org/10.1002/9781119711629 ch12.
[23] A K M Bahalul Haque, Bharat Bhushan, Security Attacks and Countermeasures in Wireless Sensor Network, 2021, https://doi.org/10.1201/9781003107521-2.
[24] Sukriti Goyal, Nikhil Sharma, Ila Kaushik, Bharat Bhushan, Blockchain as a Solution for Security Attacks in Named Data Networking of Things, 2021, https://doi.org/10.1016/B978-0-12-821255-4.00010-9.
[25] Dhanabal Thirumoorthy, Tucha Kedir, Rabira Geleta, Abdul Kadir, Chirag Patel, A secured frame work for searching and sharing of datain cloud based services using IoT, SSRN Electronic Journal 6 (2019) 503–507.
[26] Gueltoum Bendiab, Betty Saridou, L. Barlow, N. Savage, Stavros Shiaeles, IoT Security Frameworks and Countermeasures, 2021, https://doi.org/10.1201/9781003006152-7.
[27] Mamoun Alazab, Sitalakshmi Venkatraman, Paul Watters, Moutaz Alazab, Zero-day malware detection based on supervised learning algorithms of API call signatures, CRPIT 121 (2011).
[28] Roderic Broadhurst, Peter Grabosky, Mamoun Alazab, Brigitte Bouhours, Steve Chon, Organizations and Cybercrime: an Analysis of the nature of groups engaged in cybercrime, International Journal of Cyber Criminology 8 (2014) 1–20, https://doi.org/10.2139/ssrn.2345525.
[29] Maleh Yassine, Youssef Baddi, Mamoun Alazab, Lo'ai Tawalbeh, Imed Romdhani, Artificial Intelligence and Blockchain for Future Cybersecurity Applications, 2021, https://doi.org/10.1007/978-3-030-74575-2.
[30] Nighat Usman, Saeeda Usman, Fazlullah Khan, Mian Ahmad Jan, Ahthasham Sajid, Mamoun Alazab, Paul Watters, Intelligent dynamic malware detection using machine learning in IP reputation for forensics data analytics, Future Generat. Comput. Syst. 118 (2021) 124–141, https://doi.org/10.1016/j.future.2021.01.004.
[31] Erdal Ozkaya, Md Rafiqul Islam, Malicious Dark Net—Tor Network, 2019, https://doi.org/10.1201/9780367260453-4.
[32] Hussein Alnabulsi, Md Rafiqul Islam, Web Sanitization from Malicious Code Injection Attacks: Applications and Techniques in Cyber Security and Intelligence, 2019, https://doi.org/10.1007/978-3-319-98776-7_27.
[33] Subahi, Anoud & Theodorakopoulos, George, Ensuring Compliance of IoT Devices with Their Privacy Policy Agreement, 2018, pp. 100–107, https://doi.org/10.1109/FiCloud.2018.00022.