Information and Software Technology 164 (2023) 107326

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Model-based security testing in IoT systems: A Rapid Review

Francesca Lonetti [*], Antonia Bertolino, Felicita Di Giandomenico

Istituto di Scienza e Tecnologie dell'Informazione, CNR, via G. Moruzzi, 1, 56124 Pisa, Italy

ARTICLE INFO

Keywords:
Internet of Things
Model-based testing
Security testing

ABSTRACT

Context: Security testing is a challenging and effort-demanding task in IoT scenarios. The heterogeneous devices expose different vulnerabilities that can influence the methods and cost of security testing. Model-based security testing techniques support the systematic generation of test cases for the assessment of security requirements by leveraging the specifications of the IoT system model and of the attack templates.
Objective: This paper aims to review the adoption of model-based security testing in the context of IoT, and then provides the first systematic and up-to-date comprehensive classification and analysis of research studies in this topic.
Method: We conducted a systematic literature review analyzing 803 publications and finally selecting 17 primary studies, which satisfied our inclusion criteria and were classified according to a set of relevant analysis dimensions.
Results: We report the state-of-the-art about the used formalisms, the test techniques, the objectives, the target applications and domains; we also identify the targeted security attacks, and discuss the challenges, gaps and future research directions.
Conclusion: Our review represents the first attempt to systematically analyze and classify existing studies on model-based security testing for IoT. According to the results, model-based security testing has been applied in core IoT domains. Models complexity and the need of modeling evolving scenarios that include heterogeneous open software and hardware components remain the most important shortcomings. Our study shows that model-based security testing of IoT applications is a promising research direction. The principal future research directions deal with: extending the existing modeling formalisms in order to capture all peculiarities and constraints of complex and large scale IoT networks; the definition of context-aware and dynamic evolution modeling approaches of IoT entities; and the combination of model-based testing techniques with other security test strategies such as penetration testing or learning techniques for model inference.

1. Introduction

The Internet of Things (IoT) is experiencing exponential growth, with developments and applications across many sectors such as home automation, retail, manufacturing, energy, transport, health, smart cities and public infrastructures. This scenario becomes increasingly attractive for attackers at any level: devices, communication channels, software and applications become potential attack surface areas, as they all could expose threats and vulnerabilities [1].

The heterogeneity of devices participating in the IoT implies different technologies and levels of complexity, due not only to their specific tasks, but also to differences in manufacturing, software, firmware, versions, interfaces or transmission speeds. Every device could be vulnerable in many parts, including its hardware, firmware, physical and web interface, as well as the adopted network protocols and services [2]. Vulnerabilities of IoT systems can also descend from unsecure default settings, unsecure update mechanisms or outdated components. Moreover, traditional authentication and authorization methods, mainly based on pre-shared cryptographic keys, are not applicable to IoT devices, since key management for resource-constrained components is hardly feasible [3].

For all the above reasons, extensive security testing of IoT systems, as well as the evaluation of their conformance to the security requirements, become crucial to prevent errors and security vulnerabilities, and to guarantee their trustworthiness. Currently, the most popular security testing techniques adopted in IoT deal with: (i) penetration testing, in which vulnerabilities are identified by simulating real-world attacks [4], and (ii) fuzz testing, aiming at stressing the system under test with non-valid data inputs or messages [5].

[*] Corresponding author.
E-mail addresses: francesca.lonetti@isti.cnr.it (F. Lonetti), antonia.bertolino@isti.cnr.it (A. Bertolino), felicita.digiandomenico@isti.cnr.it (F. Di Giandomenico).

https://doi.org/10.1016/j.infsof.2023.107326
Received 19 December 2022; Received in revised form 3 August 2023; Accepted 30 August 2023
Available online 3 September 2023

A relevant approach to assess complex and distributed systems is Model-based Testing (MBT) [6], which derives test cases automatically from a (set of) model(s) of the System under Test (SUT) and/or of its environment. MBT has been attracting great interest in research for the last two decades at least, and some recent studies observed several concrete benefits deriving from its adoption in practice. For example, Garousi et al. [7] report that in a controlled experiment with a software testing company, MBT helped improve test case design in comparison to the previously used model-free test scripts, and also increased the fault-detection effectiveness, along with other *"intangible but important benefits"*. Other practical benefits reported by Peleska et al. [8] include the automated traceability of requirements, smooth regeneration of test procedures in regression testing, intuitive and more efficient analysis of test results.

In view of the above benefits in adopting MBT, even for the development and testing of IoT systems we assist to a growing interest in model-based approaches, e.g., [9–11]. MBT approaches allow to tackle the heterogeneity of the IoT devices and network protocols. In fact, they rely on unified concepts at model level and leverage state-of-the-art Model-driven Engineering (MDE) technologies to target low-level technical aspects of devices, programming languages or protocols that are source of heterogeneity [12]. Therefore, MBT lends itself as a suitable and effective testing approach for IoT systems, especially since it can support the assessment of different architectural decisions [13], and facilitates integration testing by allowing for component mocking [9]. In a recent work, Ahmad et al. [14] provide a comprehensive overview of the benefits and the challenges of adopting MBT for IoT systems, and illustrate MBT specificities for several testing concerns through demonstrations with real case studies in the context of FIWARE EU IoT enabled platforms.

Within the broad scope of MBT, Model-based Security Testing (MBST) more specifically addresses the security requirements of the SUT, such as authentication, authorization, confidentiality and integrity of exchanged data [15]. MBST clearly yields the same benefits above discussed with regard to MBT. However, when it is applied for security testing, the models have to be enriched with the security goals that the SUT will have to abide by, as exemplified by Peroli et al. [16], but with the key observed advantage of the reusability of such security-enriched models. In another study [17], Mahmood et al. report that their MBST approach could facilitate the systematic identification of threats and contributed to save time and manual effort thanks to the automated generation of test-cases.

Traditionally, security has not always been considered a priority by IoT system engineers during the design phase, but in recent years awareness is growing that security concerns of IoT systems must be addressed since the early design stages [10,18,19], and correspondingly, some studies addressing MBST in IoT have appeared. However, to the best of our knowledge, no effort has been spent so far on the systematic review and classification of studies on MBST for IoT systems. From a comparison of our paper with related work (see Table 1), this is the first comprehensive classification of research results about MBST in IoT.

Felderer et al. [15] conducted an extensive review of more than one hundred papers published until 2016 on MBST. Even though in this study a large number of articles about model-based security testing is reviewed, there is no reference to IoT applications. Several works address the broad topic of security testing, without focusing on IoT, e.g., Refs. [20,27]. Other works overview general testing tools and techniques for IoT, but without specifically addressing MBT [21] or security issues [25]. Other proposals informally review MBT methods as well as security techniques for IoT, e.g., references [14,26,28], but they do not follow a systematic study of the literature. Finally, the authors of [24] present a systematic mapping study on the use of model-based approaches to assess non-functional aspects of IoT systems, including also security. However, this work only provides a list of papers addressing security but does not present an analysis and classification of model-based solutions for security testing in IoT.

The aim of this paper is to fill this gap by providing a systematic and up-to-date classification and analysis of existing works on MBST in IoT contexts. We decided to follow the Rapid Review research method. The main reason was that the latter usually allows authors to deliver evidence in a shorter time-frame and with lower effort than the best known Systematic Literature Review method [32]. This reduction of time and effort, which is achieved through some simplification of the process, can be important given the timeliness of the covered topic. However, it is important to notice that we applied all efforts to maintain a rigorous and repeatable protocol [33] and to deliver a comprehensive and informative study.

We analyzed 803 publications retrieved from an automated search on Scopus, which is a comprehensive meta-database covering conference proceedings and journal publications from major publishers [34], and a snowballing cycle over Google Scholar. From this set of papers, only 17 primary studies passed the systematic selection and have been classified according to a set of relevant analysis dimensions. Although we did not find a high number of primary studies in absolute terms, they all appeared in a short time range (last 5 years), thus indicating that MBST is currently gaining a growing attention in the context of IoT systems. For this reason, our effort to collect and categorize into one systematic review the found studies can be useful both to inspire ([35], p. 2) "*areas for further investigation*", and to "*appropriately position new research activities*" on this topic. With such motivations, we discuss among others the pursued test objectives, the formalisms and techniques used, the attacks targeted, as well as the open challenges and research gaps.

According to the main outcomes of our study, MBST appears indeed a promising research direction. It has been already applied in core application domains of IoT systems, and allowed to identify several types of attacks. The growing adoption of MBST is also facilitated by the increasing usage of standard IoT technologies that provide useful support for model design. With this work, which is the first one reviewing and classifying the existing literature on MBST for IoT in a systematic way, we aim at driving further research on this topic and fostering its application on larger case studies.

The remainder of the article is structured as follows: Section 2 describes the background concepts of this study, while Section 3 presents the related work; Section 4 exposes the Rapid Review research methodology, and Section 5 shows the results; then Section 6 discusses the most interesting challenges and research directions emerged from this study; Section 7 presents threats to validity, and finally Section 8 concludes the paper with a summary of the key findings.

2. Background

The main topics addressed in this Rapid Review span over two major research directions that are: model-based security testing and IoT security testing.

2.1. Model-based security testing

MBT leverages explicit models that specify the relevant information of the SUT and/or its environment [6]. The main goal of MBT is the derivation of test cases from the model in an automatic way according to a set of test selection criteria. Specifically, in the MBT process three main steps can be identified:

- building a model (test model) of the SUT and/or its environment from given requirements, existing specifications or the SUT;
- defining a set of test selection criteria to reduce the number of derived test cases;
- generating (typically in automatic way) a set of test cases from the model applying test selection criteria. It is important to remark that the tests so generated are implementation-independent and must be then translated into more concrete tests to allow for their (automated) execution on the SUT.

Table 1
Comparison with related works.

Paper	Year	Aim of review	Review approach	Focus on MBT	Focus on security testing	MBST analysis and classification	Focus on IoT testing
[15]	2016	Taxonomy and classification of MBST approaches	Systematic search	✓	✓	✓	–
[20]	2017	Security testing approaches	Informal	–	✓	–	–
[14]	2018	MBT approaches for conformance, security and robustness in IoT	Informal	✓	✓	–	✓
[21]	2018	Testing tools and techniques for IoT	Informal	–	✓	–	✓
[22]	2018	Comparison of testing tools for IoT	Systematic search	–	–	–	✓
[23]	2019	Mapping study on testing of IoT	Systematic search	–	✓	–	✓
[24]	2020	Mapping study on model-based quality assessment in IoT	Systematic search + manual search	✓	✓	–	✓
[25]	2020	Mapping study on integration and interoperability testing in IoT	Systematic search + snowballing	–	–	–	✓
[26]	2021	Automotive cybersecurity testbed and test methods	Informal	✓	✓	–	✓
[27]	2021	Security testing techniques	Systematic search	–	✓	–	–
[28]	2022	Testing methods and testbeds in IoT	Informal	✓	✓	–	✓
[29]	2023	Research communities on vulnerability assessments and ethical hacking	Systematic search	✓	✓	–	✓
[30]	2023	Bibliometric analysis on model-based system engineering of IoT	Systematic search	✓	–	–	–
[31]	2023	MBT and MBST approaches in automotive domain	Systematic search	✓	✓	✓	–
This review	2023	Review and classification of MBST approaches in IoT	Systematic search + snowballing	✓	✓	✓	✓

In the context of security, model-based security testing consists of model-based testing of security requirements [15]. It represents an attractive research area enabling automation and enhancement of security test procedures in industrial environments. Several model-based frameworks for testing security properties have been developed using different formalisms such as Unified Modeling Language (UML) based diagrams, timed automata, or Colored Petri Nets (CPNs), among others.

UML represents a family of design notations largely adopted to model software architectures [36]. In MBST, UML class diagrams [37] are used to model the different entities of the IoT scenario, such as for instance the smart objects, the server nodes or the exchanged messages among nodes. To express constraints on UML models, the Object Constraint Language (OCL) [38] is used. OCL expressions allow to navigate the model elements and define operations on these elements leveraging the first-order predicate logic. In the context of MBST, OCL is used to specify operations of the IoT system related to devices and protocols as well as the test purposes. Test cases are generated usually by adopting structural coverage criteria or search-based techniques applied to OCL constraints. An example on how to generate test data leveraging class diagram models and OCL constraints is presented, e.g., in [39]. Specifically, the authors represent the model of the SUT by a class diagram that defines attributes and functions. From this class diagram, an object diagram is instantiated that is then used as input data for the operation parameters during test generation. In addition, the dynamic behavior of the tested functions is described by pre and post conditions expressed in OCL code. The latter specifies the functions behavior listing all possible cases. Using the object diagram and the OCL constraints, test cases can be derived automatically (for instance in [39] the CertifyIt tool is adopted).

Moreover, UML security profiles have been defined that extend UML specification with security related information. For instance, labels including security information (in UMLsec [40]) or role-based access control policies (in SecureUML [41]) or stereotypes indicating the attack surface (in the security profile presented in [42]) are added to the UML specification. In the context of IoT systems, the authors of [43] leverage the security profile presented in [42] to generate penetration tests for automotive systems.

In the last two decades, timed automata and their extensions (price timed automata, extended timed automata) have been used to model and verify security properties (for instance, analyses of role-based access control (RBAC) models or correctness checking of security protocols) [44]. Timed automata are represented as directed and connected graphs extended with clocks (real-valued variables) and invariants (i.e. constraints on clocks) and are adopted to formally verify timed events and their order in concurrent timed systems [44]. The timed automata formalism is supported by a large number of model checking tools and techniques, of which the most popular is UPPAAL [45]. Recently, timed automata have been used as a target formalism to model and verify attack trees [46] in security contexts. Attack trees, inspired by fault trees, are a tree-based formalism representing the attacker's behavior. In the context of IoT testing, Krichen et al. [47] show how attack trees can be translated in a network of price timed automata that is then used for extracting test cases, leveraging the UPPAAL tool.

An alternative graphical notation for modeling security properties deals with CPNs that are an extension of Petri Nets with a high level programming language to express and validate timed constrains in large systems [48]. CPNs have been used for instance to model trusted authentication architectures for IoT applications and verify by model-checking that these architectures satisfy a set of security properties [49].

Other formalisms less frequently adopted for modeling security requirements are: (i) Data Flow Diagrams (DFDs), representing the system as well as the external entities, processes, data flows, and data stores interacting with it [50], complemented with threat templates that define several attributes describing the characteristics of the threat to the system [51]; (ii) more specific graph types, such as topology graphs [52] or semi-formal graphs [53]; finally (iii) Business Process Model and Notation (BPMN) [54] allowing to specify security scenarios and to generate test cases for covering the events of end-to-end security processes [14].

MBST has been applied for assessing the functionalities of authorization mechanisms, specifically access and usage control systems, where the model is usually derived from the policy that is used for configuring the authorization mechanism [55]. MBST has been also applied in specific critical domains, such as automotive for validating the over-the-air software update systems [17]. Furthermore, MBST has been integrated with other security testing approaches such as fuzzing or penetration testing, where functional models are complemented with the specification of threats and potential vulnerabilities. Specific techniques for deriving the test model from previous artifacts of security engineering have been also proposed [56].

2.2. IoT security testing

Recent findings [57] show that cyber-attacks are growing, especially towards infrastructure-less networks, such as IoT. Shah and Sengupta [58] compiled a comprehensive classification of the broad variety of potential attacks threatening modern IoT devices, including wearable devices, smart-home devices and machine-to-machine devices. The huge number of interconnected devices can be exploited by the cybercriminals to perform attacks that may involve the security of large pervasive information systems or even human life. A widely referred example of these attacks is the Mirai attack in 2016 [59], in which the attacker was able to identify a large number of IoT devices and use them to cause a Denial of Service (DoS) attack on Domain Name System servers, hindering the access to most popular web sites. As just one example among many potential safety–critical attacks, in February 2021 the media reported the timely discovery of the attempt to poison the water supplied by a water-treatment plant in Oldsmar (Florida) [60]. The intruder was able to take control of the plant by operating from remote on the IoT device that controlled the level of sodium hydroxide (i.e., caustic soda), trying to increase it to degrees that human tissues cannot tolerate, before being fortunately spotted by a monitoring operator.

Many taxonomies exist aiming to classify IoT attacks according to different dimensions such as the adopted wireless communication technologies [61], the different layers of the IoT technology [58,62] or the vulnerability object (i.e. devices, network, software or data) [63,64]. This last classification represents a common approach to present IoT attacks and considers four broad categories: physical attacks, network attacks, software attacks and data attacks [63,64].

In physical attacks, the attacker is able to physically interact with the user or node of the IoT system by, for instance: (i) replacing the node or part of its hardware (hardware tampering attack); (ii) injecting a malicious node between the network nodes or injecting malicious code into a node (injection attack) in order to have access and control all the data flowing in the network; (iii) keeping the node awake for long time by feeding wrong input and causing power consumption and then node shutdown (sleep denial attack); (iv) sending fake signals to interrupt the ongoing radio transmissions of the IoT node or jamming the signals in the wireless network denying the communication between the IoT nodes (jamming attack); (v) physically manipulating the IoT user in order to obtain confidential information (social engineering attack).

Network attacks target the IoT system network and consist, for instance, in: (i) spoofing RFID (Radio Frequency Identification) signal to get the RFID tag identifier information imprinted on the RFID tag, then using this identifier to transmit attacker's data and obtain full access to the systems (spoofing attack); (ii) reading, modifying or even deleting data on the RFID nodes leveraging the lack of authentication mechanisms in RFID systems (RFID unauthorized access); (iii) eavesdropping and controlling the communication between two IoT nodes in order to access restricted data (eavesdropping attack); (iv) flooding messages or connection requests into the IoT network which result into slow down or crash of the network resource (DoS attack).

Software attacks leverage software security vulnerabilities of the IoT system. Examples of such kind of attacks are virus, worms, or mobile malware such as Trojan horse; through the usage of these malicious software an attacker may, e.g., leak or tamper vital information, or cause performance degradation or denial of service. A major vulnerability derives from the IoT lower processing power compared to general IT equipment such as servers or PCs, which prevents the usage of widely deployed operating systems, such as Windows or Linux. The result is that general IT cybersecurity tools do not work on IoT devices, exposing them to numerous cyber attacks; in particular, malware attacks are showing an increasing trend, as presented in a recent report by Sonic Wall.[1] Vulnerabilities in web applications and related software for IoT devices further increase the attack surface. Web applications can, for example, be exploited to steal user credentials or push malicious (firmware) updates.

Finally, data attacks refer to malicious actions aiming to compromise the security or privacy of data stored or exchanged in the IoT network. Examples of such attacks deal with: (i) data inconsistency, in which the attacker aims to compromise the data integrity; (ii) unauthorized access, in which unauthorized users can gain access to sensitive data violating access control mechanisms; (iii) data leakage, in which malicious users can access and disclose sensitive or confidential data.

ENISA[2] provides recommendations and guidelines for the identification and mitigation of threats that might impact the IoT supply chain, proposing cybersecurity testing as a main activity to detect misconfigurations or errors of IoT devices. The main goal of IoT security testing is to detect any potential vulnerability of the IoT system under test that could be exploited by an attacker or malicious user. By identifying potential vulnerabilities, and putting in place more robust security mechanisms, the ultimate goal is to improve the overall security and the users' trust in the system itself.

IoT security testing represents a broad term encompassing a plethora of testing methodologies and tools targeting the different security requirements of the IoT system and the different components of the IoT scenario. Testing the security of IoT systems aims to guarantee authentication and authorization of devices and people during data access, the integrity of the transmitted data thorough the usage of encryption techniques and their availability. Mobile security testing is applied in the IoT context if devices communicate through mobile networks, while cloud security testing is performed if the exchanged data are in the cloud. Also firmware and hardware vulnerability detection represents a common form of IoT security testing [65]. Fuzz testing and penetration testing are the most popular test methods to detect vulnerabilities in IoT. The former aims to run the SUT with a large amount of malformed input data, in order to monitor the status of the program and detect abnormal situations. Fuzzing techniques are usually applied to detect memory-corruption flaws or other vulnerabilities in IoT device firmware [66] or are combined with static analysis

[1] https://www.sonicwall.com/resources/white-papers/2022-sonicwall-cyber-threat-report/

[2] https://www.enisa.europa.eu/publications/guidelines-for-securing-the-internet-of-things

techniques to detect and verify authentication flaws in IoT embedded systems [67]. Penetration testing represents an attempt to breach the security of the system in order to report the flaws that can cause vulnerabilities. The work in [68] provides an overview of different types and issues of penetration testing in IoT.

In recent years, MBST is gaining more attention for assessing the security of IoT protection mechanisms in several application domains. The goal of this paper is to review and classify MBST approaches as detailed in Section 5.

3. Related work

Testing of IoT systems is the subject of several research studies in the last years, proposing a variety of IoT testing methods and infrastructures. Although several surveys exist, no previous work provides an up-to-date systematic review and classification of model-based security testing for IoT systems.

The authors of [21] overview the different types of testing that can be applied in the IoT context such as usability, scalability and security, and provide a survey of existing testing tools and technologies in the field. This work overviews informally existing methods and tools for testing, without applying a systematic approach, nor providing a classification of existing solutions. The work in [23] makes a systematic classification of types of testing applied in the context of IoT, including security testing; however, it does not provide explicit references to model-based testing approaches. The already mentioned paper by Ahmad et al. [14] provides a comprehensive overview of how MBT can address several IoT testing challenges, including conformance testing, robustness testing, and security testing. However, the study focuses on the specificities of applying MBT approaches in the IoT domain, and does not aim at providing a systematic review of the literature on MBST, as we do here.

A recent survey [28] summarizes the latest testing techniques addressing different IoT domains and testing objectives. This survey includes model-based testing as one of the prominent testing techniques for IoT and classifies a set of existing model-based testing approaches for IoT according to the adopted test model and the addressed research area, including healthcare, smart home, smart cities among others. This broad survey also analyzes some frameworks and testbeds for security of IoT devices; however, it does not refer to specific solutions for model-based security testing.

A systematic literature review that analyzes the research on vulnerability assessments and ethical hacking based on keywords of articles published between 1975 and 2022 is presented in [29]. It identifies security testing and internet of things as two of the important research communities in this field. Moreover, model-based testing is considered one of the two dominant subcommunities of security testing. However, this study analyzes each research community in terms of the most cited articles, the most cited authors, the top publication forums, and the most cited affiliation countries without performing a classification of the papers according to their content. Similarly, the work in [30] presents a bibliometric literature analysis of model-based system engineering of IoT, considering the use of MDE as one of the key emerging themes and future research areas in the design and development of IoT systems while the work in [22] provides a brief comparison of available testing tools in the IoT domain.

The authors of [24] present an extensive mapping study on the use of model-based approaches to assess quality aspects of IoT systems. This study reviews those papers, published from 2009 to 2019, that presented the explicit adoption of models to validate quality aspects of IoT applications, including performance, reliability and security. Therefore, the work is a broader review, aiming to show how testing in general and more specifically model-based testing approaches are used to assess quality aspects of IoT systems. Instead, our work specifically focuses on security aspects of IoT systems and provides a classification of existing solutions on model-based security testing. The authors of [15] provide a taxonomy and specific classification criteria for MBST approaches, used to systematically classify existing model-based security testing approaches in different domains until 2016. However, this work does not contain any explicit reference to IoT systems. Similarly, a very recent survey covers MBT and MBST approaches in the automotive domain without any reference to IoT systems [31].

Recent surveys cover specific topics within the broad field of IoT testing. For instance, the authors of [26] make a survey of seven security testbeds and four methods for cybersecurity testing tailored to the automotive domain, including MBT among them. The work in [25] provides a systematic mapping study of testing methods, with the only scope to address integration and interoperability of IoT devices.

Finally, other studies address the more general topic of security testing. For instance, the authors of [27] provide a general taxonomy of security testing techniques whereas Anwer et al. [20] propose a mapping of security testing techniques with the attack types they cover, without addressing model-based testing. However, these general surveys do not focus on security needs of IoT systems or specific vulnerabilities of the IoT system under test.

To facilitate comparison, related works are summarized in Table 1. The table shows: in the first column the reference to the work; in the second column the publication year; in the third column the aim of the review (some reviews cover broad topics such as security testing or IoT testing, other surveys focus on specific testing techniques such as MBT or MBST); in the fourth column the adopted research method (in particular, whether the selection of presented studies is done ad hoc or adopting a systematic procedure); the fifth column indicates if the work addresses MBT; the sixth column indicates if the work addresses security testing; the seventh column indicates if the work provides an analysis and classification of MBST; finally the last column indicates if the work focuses on IoT testing. Notwithstanding the growing interest on model-based testing and security aspects of IoT applications in the last years, the analysis of the literature (summarized in Table 1) shows the lack of a survey and systematic classification of MBST solutions for IoT applications. The goal of this paper (as shown in the last row of Table 1) is to fill this gap by providing an up-to-date comprehensive classification of research studies in MBST for IoT. This paper also aims to discuss challenges and gaps about MBST for IoT, paving the way to further research in the field.

4. Research method — RAPID REVIEW

In this section, we present the Rapid Review research process we have adopted to guide our work. A Rapid Review is a method of knowledge synthesis that aims to give evidence on a problem with a lower cost than a Systematic Literature Review [32]. The Rapid Review has been designed to review new or emerging research topics or provide updates of previous reviews. It follows a systematic protocol, however some steps of a full systematic process are simplified or omitted to give more timely results [33]. For instance, Rapid Review limits the search results by considering only a search source or a reduced publication date, does not conduct quality assessment, or presents results with no formal synthesis [69]. Rapid Review can speed up the knowledge transfer process to practitioners [70] and represents a complementary approach that does not aim to substitute Full Systematic Review [33]. Nevertheless, it has been showed that Rapid Review complemented by a rigorous snowballing process, can reach as good results as Full Systematic Reviews [71]. We detail our review process in Section 4.1.

4.1. Review process

In conducting our review, we followed the well-known guidelines by Kitchenham and co-authors [35] for performing systematic literature reviews in Software Engineering and the Rapid Review protocol of [69]. Our research goal is to characterize the model-based software testing techniques and technologies that address security in the context of

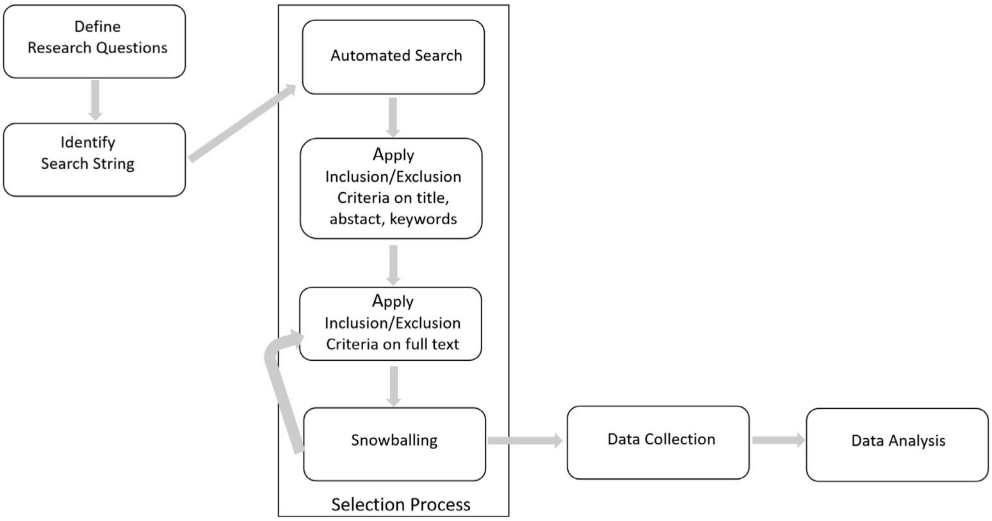

Fig. 1. Review process.

IoT. More precisely, we are interested to understand whether (and how) model-based testing is applied for security purposes into the IoT domain, what are the adopted formalisms, techniques, the addressed attacks and application domains as well as the testing purposes. To accomplish our research goal, we first formulated a set of research questions that our study aims to answer, which are presented in Section 4.1.1. These research questions identify the most important aspects of MBST in IoT, including challenges, gaps and future research directions. According to these research questions we defined the search terms and then the search string presented in Table 2. We executed this search string on the electronic database, and applied the selection process described in Section 4.1.2. Then we collected the data as explained in Section 4.1.3 and identified the set of relevant primary studies that we analyzed to answer the research questions. We present the results of our analysis in Section 5 and Section 6. Fig. 1 better details our review process.

4.1.1. *Research questions*

To reach our research goal, we formulated the following Research Questions (RQs):

1. RQ1: What are the formalisms mostly used for model specification in MBST for IoT?
2. RQ2: What are the main testing objectives of the proposed MBST approaches in IoT?
3. RQ3: What are the techniques mainly used for test cases generation/execution for MBST in IoT?
4. RQ4: What are the most targeted applications/domains of MBST in IoT?
5. RQ5: What are the most targeted attacks of MBST in IoT?
6. RQ6: What are the challenges, gaps and future research directions related to MBST in IoT?

4.1.2. *Selection process*

According to [69], to reduce the search time for primary studies in performing our Rapid Review, we selected papers by querying the Scopus[3] repository, which includes results from most relevant software engineering digital libraries. The search on Scopus has been executed in two rounds: in the former we selected English papers until November 2021; in the latter additional English papers were selected until April 2022. In particular, in each round of our search (see Fig. 2), we applied in Scopus the search string of Table 2 to title, abstract and keywords. In each round, our papers selection process included three main steps. During the first step, each of the authors read title, abstract and keywords of a subset of papers and applied inclusion and exclusion criteria described in Table 3. The subset of papers assigned to each author was randomly selected and the load of papers assigned to each author was balanced. Moreover, to assure consistency during the process, this step was conducted in several iterations with plenary meetings among all the authors at each iteration to review the assignment and resolve possible doubts. In this step, we wanted to exclude papers not targeting the Rapid Review topics, i.e., papers not addressing model-based testing solutions for guaranteeing security in IoT domains. Also works not including primary studies (such as survey papers or monographs, theses or books) and not peer-reviewed papers were excluded.

In the second step, starting from the set of papers obtained in the previous step, we read the full text of a subset of papers and applied the inclusion and exclusion criteria of Table 3. Every paper was read by two authors. Precisely, the first author read the whole set of papers (which provided consistency along the process), whereas the second and third author read half of the papers randomly chosen. Moreover, several meetings among all three authors have been carried out to discuss possible doubts during the paper selection. In this second step, we wanted to exclude, after reading the full paper, studies not clearly addressing model-based security testing solutions in IoT domain. Note that, according to inclusion/exclusion criteria of Table 3, we did not consider in our Rapid Review: (i) publications presenting classical fuzzy or penetration testing approaches without adopting explicit models; (ii) static analysis solutions or model-checking solutions which only consider the model verification. According to [15], we consider as model-based testing approaches only proposals where at least abstract test cases are derived. Following the guidelines of Rapid Review [69], quality assessment procedures of primary studies have not been applied. We included journal, conference, book chapter and workshop papers retrieved from Scopus and Google Scholar databases. This also ensures a more complete and inclusive view of the model-based security testing for IoT, particularly important in the case of emergent research directions. Nevertheless, as a guarantee for quality, in our selection process we only accepted peer-reviewed papers.

Finally, in the third step, to complement the results of the automated search and verify that all the relevant primary studies have been included, a backward and forward snowballing procedure [72] was performed by each of the authors on a balanced and randomly chosen subset of papers found in the second step. Specifically, for each of these papers, we analyzed for backward snowballing its list of

[3] https://www.scopus.com

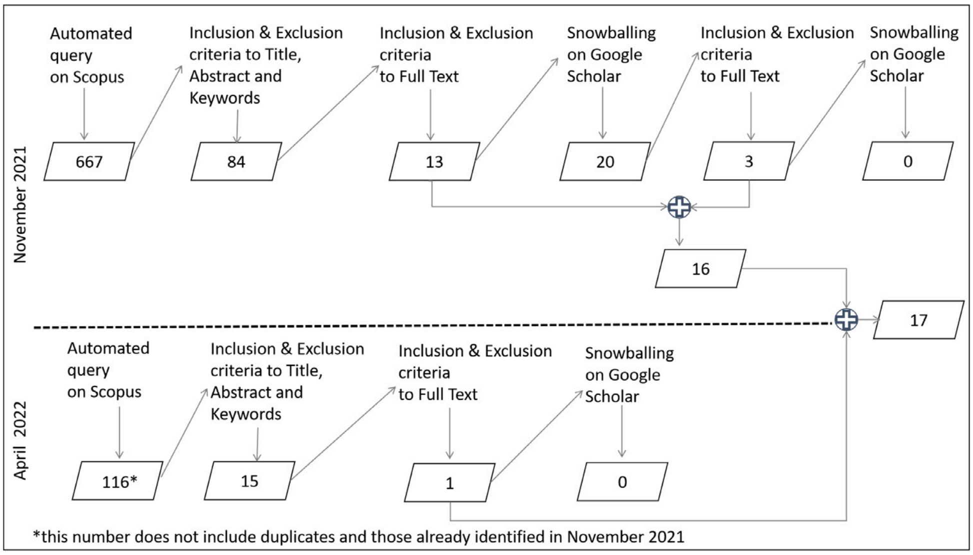
November 2021
Automated query on Scopus
Inclusion & Exclusion criteria to Title, Abstract and Keywords
Inclusion & Exclusion criteria to Full Text
Snowballing on Google Scholar
Inclusion & Exclusion criteria to Full Text
Snowballing on Google Scholar
667
84
13
20
3
0
16
17
April 2022
Automated query on Scopus
Inclusion & Exclusion criteria to Title, Abstract and Keywords
Inclusion & Exclusion criteria to Full Text
Snowballing on Google Scholar
116*
15
1
0
*this number does not include duplicates and those already identified in November 2021

Fig. 2. Selection process and numerical outcomes.

Table 2
Search string.

({model} $<OR>$ {modeling } $<OR>$ {model-based})
$<AND>$ {security} $<AND>$ ({testing} $<OR>$ {test}) $<AND>$ ({IoT} $<OR>$ {Internet of Things})

references, and for forward snowballing its citations in Google Scholar.[4] We examined the text of these papers applying the inclusion/exclusion criteria of Table 3, according to the guidelines for snowballing in systematic literature reviews presented in [72]. As depicted in Fig. 2, the snowballing procedure ended after two iterations in the first round and one iteration in the second round respectively, until no new paper was found. As detailed in Section 4.1.3, the low number of primary studies included after the snowballing process (only 3 papers) confirmed the completeness of the results of the automated search (with publication date from the beginning to April 2022).

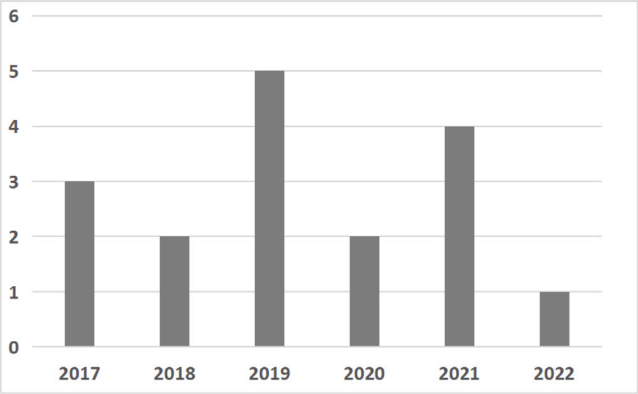

Fig. 3. Distribution of primary studies by year.

4.1.3. *Data collection*

The result of our initial search executing the query of Table 2 in November 2021 on Scopus database, was 667 primary studies (see Fig. 2). After applying the inclusion and exclusion criteria presented in Table 3 to title, keywords and abstract, we excluded from the initial set of papers: proceedings titles and tables of contents (114 sources), books (1 paper), surveys (3 papers), and papers not addressing testing strategies or algorithms or test frameworks (465), thus obtaining a set of 84 papers. After reading the full text of these 84 papers we excluded: 1 paper that was in Spanish (only title and abstract were in English), and other 70 papers that did not address MBST solutions for IoT systems. Then, we obtained a set of 13 primary studies; after performing the first iteration of backward and forward snowballing over Google Scholar on this set of papers, other 20 papers were identified, of which 3 passed the selection process[5] and were added, eventually obtaining a set of 16 primary studies. No relevant paper was found in the second iteration of snowballing on this set of 3 papers. In April 2022, we re-executed the same query of Table 2 on Scopus in order to update the list of sources obtaining a set of 116 new sources. We applied the same selection process described in Section 4.1.2 to this set of additional sources. In this second round, we selected 15 papers after applying the inclusion and exclusion criteria presented in Table 3 to title, keywords and abstract. From this set, after reading the full text only one primary study was selected. This relevant primary study has been added to previously selected primary studies, obtaining a final set of 17 primary studies. In this case, no further papers were identified after performing backward and forward snowballing. The selected primary studies are listed in Table 5 under the column labeled *Ref#*.

5. Data analysis

Model-based security testing in IoT represents a recent research topic. All the selected primary studies have been published in the last five years (as showed in Fig. 3). Various types of venues have been considered over the years (see Table 4). We observed that in total there are 8 conference articles, 6 journal articles, 2 book chapters and 1 workshop paper considered in this study.

In this section, we analyze the results of our Rapid Review and answer the research questions presented in Section 4.

As depicted in Table 5, we classify the selected primary studies (column labeled *Ref#*) according to some relevant dimensions that are: (i) the formalism used for specifying the input model for test cases generation (column labeled *formalism*); (ii) the test objective (column

[4] http://scholar.google.com

[5] Note that, among these 3 papers, the study by Ahmad et al. [14] includes both a broader overview of MBT and a section proposing a specific MBST approach, then it has been both cited as a related work and included among the primary studies.

Table 3
Inclusion and exclusion criteria.

Inclusion criteria
Studies on techniques/algorithms/strategies leveraging models for testing security in IoT
Studies on model-based security testing aspects, architectures, frameworks that are relevant in IoT
Studies on explicit models for guaranteeing security in IoT
Exclusion criteria
Studies from fields different from software testing in IoT
Studies from fields different from security in IoT
Studies not explicitly presenting model-based testing approaches for security in IoT
Studies on classical security testing approaches (like fuzzy or penetration) without explicit models in IoT
Studies on static analysis or consistency checking or verification of the test model in IoT
Not primary studies (surveys, editorials, panels, theses, white papers, books, etc.) or not peer-reviewed studies

Table 4
Year and venue of selected primary studies.

Ref#	Year	Venue	Journal/conference name
[14]	2018	Journal	Advances in Computers
[73]	2020	Conference	International Conference on Evaluationof Novel Approaches to Software Engineering
[74]	2021	Journal	Computers, Materials & Continua
[47]	2019	Conference	International Conference on Evaluation of Novel Approaches to Software Engineering
[43]	2018	Conference	Euromicro Conference on Digital System Design
[75]	2019	Journal	IEEE Access
[49]	2021	Journal	IEEE Internet of Things Journal
[76]	2017	Conference	Global Internet of Things Summit
[77]	2019	Conference	Central European Cybersecurity Conference
[78]	2017	Book chapter	Cognitive HyperconnectedDigital Transformation:Internet of Things Intelligence Evolution
[79]	2021	Conference	International Conference on Software Testing, Verification and Validation
[80]	2019	Conference	International Conference onInformation Security Applications
[81]	2017	Conference	International Conference on Smart Cities,Infrastructure, Technologies and Applications
[82]	2020	Book chapter	Smart Infrastructure and Applications
[52]	2021	Workshop	IEEE European Symposium on Securityand Privacy Workshops
[37]	2019	Journal	Computer Standards & Interfaces
[53]	2022	Journal	Computers & Electrical Engineering

labeled *test objective*), representing the goals of model-based testing, including security testing, compliance verification of the system with the functional model and other non-functional properties, such as load or performance; (iii) the adoption of attack/threat models for test cases derivation (column labeled *attack/threat model*); (iv) the test technique proposed for test cases generation (column labeled *test technique*); (v) artifacts or tools involved in the test generation (column labeled *test generation*); (vi) artifacts or tools involved in the test execution (column labeled *test execution*); (vii) the target IoT application that is the object of testing (column labeled *target application*) and finally (viii) the specific domain of the IoT application (column labeled *application domain*). Moreover, in Table 6 we show the most targeted security attacks by the selected primary studies. Each of the authors initially classified a balanced set of papers randomly selected, then the final classification has been discussed among all the authors.

In the following, we answer to the first five research questions (RQ1, RQ2, RQ3, RQ4, RQ5), whereas we refer to Section 6 for a discussion of challenges, gaps and future research directions related to MBST in IoT, for answering RQ6.

RQ1- What are the formalisms mostly used for model specification in MBST for IoT? In model-based security testing, several formalisms are used to define the model of the SUT. The most used modeling language to define the IoT system is **UML**. In 7 out of the 17 primary studies UML class diagrams are used to describe the abstract IoT objects of the system and their dependencies [14,37,43,75,76,78,80]. In the 85% of the cases, the UML notation is complemented by the **Object Constraint Language** (OCL) [38] in order to express the operations and expected dynamic behavior of the system under test [14,37,75,76,78,80]. For each operation (for instance of an IoT protocol), a set of parameters, preconditions and postconditions is specified [75,76,80]. In only one primary study [43], **UML security profile** is adopted that include stereotypes to specify vulnerable points in the IoT system or parts of it that are protected against violations. These stereotypes define for instance the used encryption and handshaking protocols or the properties that are protected with authentication and authorization mechanisms [43]. **Timed automata** and their extensions are the second commonly used formalism (used in 5 out of 17 primary studies) to define the IoT system and the security aspects of interest [47,73,74,81,82]. Timed automata represent a finite automata model that may be defined as finite graphs extended with clocks and simple constraints over states, clocks or deadlines.

Distributed IoT systems can be also modeled by using **hierarchical colored Petri Nets** [49] that allow to model concurrency, synchronization, communication, and resource sharing, or by using **Business Process Model and Notation (BPMN)** models that allow to specify end-to-end security scenarios [14]. Other formalisms adopted in only one primary study are: **topology graph** [52]; **data flow graph** [77]; and **semi-formal graph** based formalism such as Multi Cloud Application Composition Model (MACM) [53] to define all relevant system components and data exchange among them. Finally, the authors of [79] use behavioral models expressed by **finite state machines** (Mealy machines) derived by automata learning for test cases generation.

To complement the IoT system specification, **attack trees** are used in 4 out the 17 primary studies [47,52,73,74]. They allow to represent graphically the strategy of a given attacker or malicious party to violate the security mechanisms of the IoT system. These attack trees can be transformed into a network of **price timed automata** [47,74] that are an extension of timed automata in which costs are assigned to states and edges. These price timed automata define the basic attack steps the attacker needs to perform in order to achieve the goal and provide the input for the tests generation. In 2 out of the 17 primary studies, **threat templates** [53,77] are adopted for specifying security flaws or a list of malicious behaviors of the attacker. In [53], the threat model is automatically derived from the specification of the IoT system, modeled using the MACM formalism. The threat templates or attack vectors representing an abstraction of the threats or attacks of the system are

Table 5
Primary studies classification.

Ref#	Formalism	Test objective	Attack/threat model	Test technique	Test generation	Test execution	Target application	Application domain
[14]	-UML class diagrams -OCL -BPMN	-security -conformance -robustness	–	-coverage of the model -fuzzing	-TTCN-3 test cases using CertifyIt	-C/C++ tests executed by TITAN in the FIT IoT Lab	-encryption/decryption algorithms -oneM2M based secure communication protocols	–
[73]	-timed automata	-functional -load -security	-attack tree	-ad hoc strategy	-abstract test cases	–	-blockchain based secure communication	-automotive
[74]	-price timed automata	-security	-attack tree	-application of UPPAAL strategy	-test scenarios	–	-web-based monitoring of operating room	-healthcare
[47]	-price timed automata	-security	-attack tree	-application of UPPAAL strategy	-TTCN-3 test cases	-tests run on cloud	-secure traffic control	-smart cities
[43]	-UML class diagrams	-security -performance	-UML profile	-penetration testing	-abstract models	-Python scripts	-CAN bus messages exchange	-automotive
[75]	-UML class diagrams -OCL	-security -risk analysis -MUD model extension	–	OCL structural coverage	-test cases using CertifyIt	-Junit tests	-EDHOC protocol implementation	–
[49]	-hierarchical colored Petri nets	-security -conformance	–	-MBT/CPN tool	-XML tests	–	-authentication -authorization -encryption	–
[76]	-UML class diagrams -OCL	-security -conformance	–	-OCL structural coverage -test purpose selection	-TTCN-3 test cases using CertifyIt	-tests executed by TITAN in IoT Testbed	-authorization -oneM2M-based post certification monitoring	–
[77]	-data flow diagrams	-security -risk analysis	-threat templates	-ad hoc strategy	-XML tests	yes	-MQTT protocol implementation	–
[78]	-UML class diagrams -OCL	-security -conformance -robustness	–	OCL structural coverage -test purpose selection -fuzzing	-TTCN-3 test cases	-C++ tests executed by TITAN	-security certification process	–
[79]	-Mealy machines	-security -conformance	–	-fuzzing	yes	yes	-MQTT protocol implementation	–
[80]	-UML class diagrams -OCL	-security	–	-OCL structural coverage	-test cases using CertifyIt	-Junit tests	-EDHOC protocol implementation	–
[81]	-extended timed automata	-security	–	-application of UPPAAL strategy	-TTCN-3 test cases	-tests run on TT4RT platform	-attack protection mechanism	-smart cities
[82]	-extended timed automata	-security	–	-application of UPPAAL strategy	-TTCN-3 test cases	-tests run on cloud	-attack protection mechanism	-smart cities
[52]	-topology graph	-security	-attack tree	-attack tree coverage	-attack vectors	yes	-vulnerabilities detection	-automotive
[37]	-UML class diagrams -OCL	-security -risk analysis	–	-OCL structural coverage	-TTCN-3 test cases using CertifyIt	-tests executed by TITAN in the FIT IoT Lab	-C-oAP-DTLS certification	–
[53]	-MACM graph model	-security	-automatically derived threat model	-penetration testing	-testing plan	yes	-OEM monitoring system	-smart home

Table 6
Primary studies *vs* attacks.

Ref#	Denial of service (DoS)	Dictionary	Injection	Brute force	Eavesdropping	Tampering	Data leakage	Key logger	Social engineering
[14]					✓				
[73]		✓		✓				✓	✓
[74]		✓		✓				✓	✓
[47]		✓		✓				✓	✓
[43]	✓			✓					
[75]	✓				✓				
[49]					✓	✓			
[76]	✓								
[77]	✓								
[78]	✓	✓							
[79]							✓		
[80]			✓		✓				
[81]									
[82]	✓								
[52]									
[37]	✓		✓						
[53]	✓		✓		✓	✓	✓		

mapped on the entities of the graph and used to automatically derive security test cases representing attack scenarios.

RQ2: What are the main testing objectives of the proposed MBST approaches in IoT? The main objective of MBST is to evaluate the **security aspects** of frameworks, infrastructures and solutions developed in the IoT context [14,37,47,52,53,74,80–82]. In the 100% of the reviewed studies, the overall main goal is to identify weak points in the IoT system from the early design, by referring to the specified models of the security mechanisms. More precisely, the addressed security properties can deal with authentication or authorization (in 2 primary studies) or encryption mechanisms (in 2 primary studies) aiming to protect sensitive data in fog/cloud-based IoT applications. In 2 primary studies, the testing phase addresses also generic IoT protection applications against the most important attacks such as distributed DoS attacks or eavesdropping [43,49]. The authors of [73] propose a testing strategy for specific block-chain based secure vehicles communication and IoT decentralized security framework.

Other MBST approaches address specific objectives related to the security of the adopted IoT solutions. For instance, the authors of [75] adopt a model-based testing approach to **enhance the Manufacturer Usage Description (MUD) profile**, a standard aiming to specify IoT device behavior, through access control lists. So, the results of model-based testing are leveraged to derive augmented MUD profiles, which consider additional security aspects. In 2 primary studies [14,78], MBT is combined with fuzzing for assessing the **robustness** of the IoT system. Another objective of MBST is the **security risks analysis** and **enforcement of security certification** frameworks and processes for specific categories of IoT products based on common standards (in 2 primary studies) [37,77]. The ability of certifying the security level of a smart IoT device is a key aspect for its adoption and deployment in the large scale IoT environment. To this purpose, MBST is used to generate tests for the smart object according to a set of selected vulnerabilities. The test report obtained after the testing process is used to provide a refined vulnerability risk mark. During the certification process, this risk mark is adopted to derive a cybersecurity label included in the generated certificate. The authors of [76] adopt model-based testing for **post-certification monitoring** of IoT systems, based on security policies management and enforcement. Specifically, when IoT devices that do not conform to a target specification level are deployed, test cases derived using MBST approaches are executed for assessing their security functional behavior. In case of failure of a subset of the test cases, test results are used as inputs to specify and enforce policies for correcting vulnerable system behavior, as well as for runtime monitoring of the policies deployed in the IoT environment.

Besides security aspects, the 30% of model-based testing strategies allow to check the **conformance** of the implemented functionalities against a formal specification [14,49,78,79] or IoT standard [76]. In particular, the authors of [52,79] adopt model-based testing to test the IoT system for conformance against a learned model, i.e. the goal is to execute concrete inputs to check if the observed behavior from the SUT conforms to the model. In [79], the conformance is checked by fuzz testing that is able to detect security vulnerabilities or unexpected behavior of the IoT system through exhaustive testing with invalid and unexpected inputs. Finally, 2 primary studies [43,73] address also **load and performance** requirements of the adopted IoT solutions whereas in only one primary study the correct **functional implementation** of security mechanisms is evaluated.

RQ3: What are the techniques mainly used for test cases generation/execution for MBST in IoT? More than 40% of the primary studies adopt **coverage criteria** to guide the model exploration during the derivation of test cases. These coverage criteria are related to the adopted modeling formalism. In 5 out of the 17 primary studies, **structural coverage of the OCL code**, specifying the SUT model and containing the operations of devices and protocols of the IoT system, is adopted. Functional test cases are derived using the exploration of the symbolic states of the model and covering the expected behaviors (test targets) expressed by OCL post-conditions [14,37,75,78,80]. Specifically, test cases are sequences of operations of the model from an initial state to a state in which the test target is verified. In 2 studies [47,74] attack tree models are used to derive abstract test cases. **Coverage of the attack tree model** is adopted in [52] for (semi-)automated generation of test cases. These test cases are specified as attack vectors for specific targets covering a path of the model. A cost is associated to each attack vector according to the addressed vulnerability. Test cases are executed, starting from those with lower costs. However, as showed in this review, test selection criteria based on behavioral coverage of the model, which represent the standard approach to generate functional tests in model-based testing, are not enough to deal with security testing objectives. Then, additional test cases (security functional test cases) are generated aiming to deal with **specific test purposes** capturing the security testing objectives [76,78]. These test purposes are formally or informally defined and represent operations or states or behaviors. They are transformed into complete test cases able to trigger unexpected behaviors of the system, or discover vulnerability trying to bypass existing security mechanisms [37].

In the context of IoT security, almost 30% of validation techniques combine formal verification and model-based testing. Specifically, test generation techniques are based on model checking. The main idea is that, after exploring all the states space of the model and verifying the specific security properties, abstract test cases are generated from this state space. These test cases support the correct implementation of the IoT system and are used to verify the conformance of the implementation with the IoT formal specification. In 4 of the primary studies [47,74,81,82] the test generation problem is represented as a reachability problem that can be solved with an extension of the well-known **model checking tool UPPAAL** [45]. The authors of [49] instead adopt the Model-based testing/Colored Petri Net **(MBT/CPN) tool** [83] for deriving abstract test cases from the verified and validated specification of the IoT system. Using this model checking tool, the model's state space is explored and input and output events are considered to guide the generation of test cases. These test cases represent different scenarios related to security issues.

In 2 of the analyzed studies, behavioral modeling is combined with penetration testing [43,53] in order to discover security issues. Specifically, in [43], **penetration testing** is applied to virtual prototypes (executable models) instead of to physical prototypes. These virtual prototypes simulate hardware and software components of IoT devices, and the interconnections among devices. Tests are generated by attack surface models specified by UML stereotypes that complement the behavioral model represented by UML class diagram, and then executed on the virtual prototype. The adoption of virtual prototypes allows to identify vulnerabilities and to verify the applicability of security mechanisms at early stages of the development process. In [53], a testing plan in terms of a list of attacks is generated by leveraging a public catalogue of common attack patterns. This list of attacks is mapped on the nodes of the MACM model and on the threats identified for the IoT system under test, and it can thus be referred by a penetration tester to systematically conduct the system security testing. In 3 studies, **fuzz testing** is combined with model-based testing to assess the robustness of the IoT system or to reveal possible faulty behaviors of the IoT system [14,78,79]. Specifically, in [79], the behavioral model of the system is automatically inferred by automata learning of a reference implementation and adopted to derive the test cases. These test cases are then concretized into invalid inputs or message sequences to the SUT by using fuzzing techniques in order to reveal weaknesses of the system.

Finally, 2 works [73,77] adopt **ad hoc test strategies** for test cases generation. For instance, the authors of [77] adopt a threat-model driven test cases generation, for creating test cases as sequences of commands derived from a threat model. Security tests are obtained by adding specific commands, representing the attack patterns derived from publicly available vulnerabilities repositories and included in the threat model, to the commands sequences. In the proposal of [73] another different test strategy is used to generate test cases. In particular, test cases are considered as trees whose nodes are collections of states of the model of the SUT. Tests are generated as extensions of the test tree by defining successors to leaf nodes, following the testing strategies presented in [84].

In all the analyzed testing strategies, a set of test cases is generated. The most used tool for tests generation is **CertifyIt** [85] (adopted in 5 out of the 17 primary studies). The generated test cases are **abstract test cases**. The most used format (adopted in 40% of the analyzed studies) for defining these test cases is a standardized test scripting language called **Testing and Test Control Notation version 3 (TTCN-3)** [86]. In 2 primary studies, abstract test cases are defined into XML format [49,77]. These abstract test cases are then translated into executable tests specifying low-level implementation details. These executable test cases are represented in different languages such as **JUnit tests** (in 2 studies) [75,80] or **C/C++ tests** (in 2 studies) [14,78] or **Python scripts** (in 1 study) [43]. Concerning the test execution, the most adopted MBT tool is **TITAN** [87] (in 4 primary studies). To support the automated test execution, test adapters are used, containing all the functions defined in the model [37,76], which must be implemented in order to execute the test suite on the real IoT devices. The generated tests can be executed also on a **cloud-oriented test architecture** [47,82], or specific **large scale testbeds** [76] including the FIT IoT Lab [14,37], or the TT4RT platform [81].

RQ4: What are the most targeted applications/domains of MBST in IoT? The analysis of the collected data provides preliminaries evidences that the targeted application domains span over core applications domains of IoT systems [88], i.e. transportation, smart cities, smart homes, and health care. The most targeted application domains of the proposed testing solutions are: automotive (in 3 primary studies) and smart cities (in 3 primary studies). The papers addressing the **automotive** domain [43,52,73] provide frameworks or solutions to assure secure transmission in Vehicle-to-Vehicle and Vehicle-to-Infrastructure communications, also based on **blockchain technology** [73]. Other applications in automotive domain deal with **secure messages exchange** on the CAN bus and **vulnerabilities detection** [43,52]. Concerning the **smart cities** domain [47,81,82], the testing object is represented by: (i) applications able to guarantee security of traffic control systems; (ii) security mechanisms for standard application and network layer protocols; (iii) protection approaches against the main security threats in the IoT.

The remaining papers address **healthcare** (1 primary study) [74] or **smart home** (1 primary study) [53] sectors. In both cases, the analyzed solutions mainly perform security assessment of IoT based **monitoring** systems. These monitoring systems aim to control either web-based management of available resources in the operating rooms [74], or home automation infrastructures such as the open source Open Energy Monitor (OEM) system [53].

However, more than 50% of the selected primary studies do not deal with any specific application domain. They present security evaluation strategies of applications or communication protocols or trusted execution environments without targeting any specific domain [14,37,49,75–80]. For instance, the authors of [75,80], apply model-based testing strategies to analyze security properties of different implementations of the **Elliptic Curve Diffie–Hellman over COSE (EDHOC) protocol** for the authenticated key exchange in IoT devices. The works in [14,49] evaluate authentication, authorization and encryption/decryption mechanisms of trusted execution environments as well as protocols for secure communication based on oneM2M standard, while the authors of [77,79] apply testing strategies to find inconsistencies in the **Message Queuing Telemetry Transport (MQTT) implementations**. Finally, other approaches present general testing solutions for **certification** of IoT protocols such as **Constrained Application Protocol (CoAP)** and **Datagram Transport Layer Security (DTLS)** [37], or IoT devices certification according to defined standards such as oneM2M [76].

RQ5: What are the most targeted attacks of MBST in IoT? Table 6 presents the attacks targeted by at least one of the analyzed primary studies. As can be seen, the most targeted type of attacks (in almost the 50% of the primary studies) is **Denial of Service (DoS)**, occurring when the attacker sends too many messages to the main server/host making it unavailable to real users [37,43,53,75–78,82]. Specifically, the work in [43] models the behavior of an attacker and the attack surface of a DoS attack. The adopted model also allows for specifying the performance and timing behavior of the system, enabling the analysis of DoS attacks. In [43], such model-based approach has been applied to an automotive system for the injection of malicious CAN messages in the network. Other approaches aim to mitigate the risk of future DoS attacks with a post certification monitoring solution [76].

As in other distributed systems, also in IoT environments, cracking the password of protected files is another common goal of an attacker. In 3 of the analyzed solutions, password cracking is modeled by using attack trees [47,73,74] in which the root of the attack tree models the main goal of the attacker, whereas the leaves correspond to the basic attacks. Typical basic attacks that an intermediate node can exploit to store some data packets and decipher them are: (i) **dictionary** attack (in 4 out of 17 primary studies), if the key used to cipher data is a dictionary word [43,47,73,74]; (ii) **brute force** attack (in 4 out of 17 primary studies), where a large number of possible key permutations are checked [73,74], (iii) **key logger** attack (in 3 out of 17 primary studies) [53,73], able to monitor and record each information typed on the computer; or finally (iv) **social engineering** (in 3 out of 17 primary studies), dealing with psychological manipulation of people in disclosing confidential information.

Other approaches [37,75,78,80] define test patterns and test scenarios able to: (i) evaluate the fulfillment of specific security properties as confidentiality or availability of the IoT system; (ii) estimate the risk that the system incurs in several attacks, in particular DoS attacks, **injection** of malicious messages, or dictionary attacks.

Other types of attack addressed by model-based testing approaches are: (i) **eavesdropping** attacks where an attacker intercepts the data in transit or gains privileged access to them (in 5 out of 17 papers) [14,49,75]; (ii) **tampering** attacks resulting in unauthorized data or system modification (in 2 out of 17 papers) [49,53]; (iii) **data leakage** in which the adversary can access and disclose local data in an unauthorized way (in 2 out of 17 papers) [53,79].

In [75] MBST is also used to extend the MUD profile in order to detect and mitigate potential security attacks on the devices, including DoS or eavesdropping attacks.

More general approaches [53,77] for security assessment of an IoT system allow to define a list of attack patterns mapped on the nodes of the IoT system model. These approaches leverage the CAPEC (Common Attack Pattern Enumeration and Classification)[6] catalogue of common attack patterns [53], maintained by MITRE, or other publicly available catalogs like the Common Vulnerabilities and Exposures (CVE)[7] [52,77]. These attack patterns contain information to implement an attack, like the preconditions, the needed resources as well as the attack execution flow, and allow testers to evaluate the feasibility of specific attacks [53] including DoS, eavesdropping, data leakage, message injection, among the others.

Only 4 out of the 17 primary studies target at least 4 different types of attacks whereas the remaining primary studies target 2 or lesser types of attacks. Moreover, 3 primary studies [43,75,80] show the effectiveness of the proposed solutions for assessing some specific attacks in the context of a case study.

The types of attacks targeted in the analyzed primary studies cover around 40% of the typical attacks of IoT systems described in Section 2.2. With respect to the attacks taxonomy analyzed in Section 2.2 and presented in [63], software attacks (such as virus, worms or malware) and some types of network attacks (such as replay attack, wormhole attack, routing information attack, sinkhole attack, RFID spoofing, sybil attack, traffic analysis attack) are not considered in the analyzed primary studies.

6. Challenges, gaps and future research directions related to MBST in IoT

In this section, we answer *RQ6: What are the challenges, gaps and future research directions related to MBST in IoT?* Specifically, this section provides challenges and future research directions as emerged by our analysis of the primary studies in MBST. Additional research issues and gaps we identified are also presented.

Our Rapid Review confirms that MBST represents a suitable and effective approach for testing IoT systems. Recent IoT systems are using more and more standardized approaches for communication and security.

More than 40% of the analyzed papers in this review target standard technologies. Specifically, standard communication protocols such as MQTT [77,79] or CoAP [37] are addressed. Model-based testing solutions are applied to the EDHOC key establishment protocol for constrained IoT devices, which is being standardized by the Internet Engineering Task Force (IETF) [75,80]. Moreover, IoT systems and their security functionalities are more and more modeled according to the specifications proposed by oneM2M standard addressing security solutions and interoperability for Machine-to-Machine and IoT technologies [37,76]. This growing **adoption of standard technologies** [75,76,79] in the IoT context allows to leverage the potential of MBST of designing a generic model of the SUT based on the standards, from which to derive then security test cases that can be executed over different system implementations.

One major advantage of MBST is the ability to model general security concepts of the IoT system and then, leveraging MDE technologies, systematically generate automated tests able to cover the security test objectives of specific devices or network protocols [78]. However, another important challenge of MBST raised in almost 30% of the analyzed papers is **full automation**, due to the increasingly large and heterogeneous nature of devices and network technologies of IoT systems [37,75,77,78,80].

A key challenge of MBST is represented by the **complexity of the model definition** [52,79]. IoT devices and protocols have a lot of peculiarities and security constraints. In 4 out the 17 primary studies, is evidenced the necessity to **extend existing modeling formalisms** [81,82] to capture the heterogeneous elements of IoT systems and provide security test models or threat models covering all interesting security aspects of the system under test and its environment [49,77]. From the results collected in this review, more than 40% of primary studies integrate different models to specify different aspects of the IoT system behavior. For instance, data flow diagrams are used to model all relevant system components of an SUT and their interactions, while threat models are used to abstract the threats to the system that are mapped on the data flow diagrams for test cases generation [77]. UML class diagrams [43,75,76,78] are leveraged to define the IoT system behavior while UML security profiles [43] model vulnerable points in the IoT system or parts of the system that are protected against violations. Moreover, specific security issues are modeled using attack trees [47,52,73,74] or threat templates [77].

The model definition can be a very expensive and effort consuming task for complex and large scale IoT systems. To overcome this problem, in 2 of the analyzed primary studies, automata learning techniques [52,79] can be adopted to infer a behavioral model of the tested system. This behavioral model can be for instance inferred by a reference implementation of the system and used for test cases derivation [79]. Although learning-based approaches are becoming widely used in a variety of contexts, there is also increasing debate on resilience related aspects when employed in critical sectors, e.g. with reference to the degree of explainability they can achieve [89], or about security issues they can be exposed to [90]. Therefore, we believe there is opportunity to explore other directions. Among them, an interesting solution to investigate in the future to address the complexity of the model definition could be the **modularization of the test model**. The main goal will be to understand how this technique, well-known for the specification and modeling of large scale systems, could be leveraged in MBST for designing and testing the security requirements of IoT systems. However, an additional challenge to tackle in this context will be that notoriously security is a **non compositional property**. A gap we identified in the analysis of the primary studies is that existing IoT modeling solutions do not explicitly target the **dynamicity and evolution of real IoT systems** during the SUT model definition. Therefore, in our vision, a future research issue in MBST will be to investigate **context-awareness and dynamic evolution modeling** approaches of IoT entities, as well as effective strategies leveraging these enhanced models to drive automated generation of security tests.

In the analyzed primary studies, a new level of complexity in security testing of IoT is represented by the **scalability and heterogeneity of IoT devices** as well as by the IoT data features. As claimed

[6] MITRE. Common Attack Pattern Enumerations and Classifications. Available from: https://capec.mitre.org.

[7] MITRE. Common Vulnerabilities and Exposures. Available from: http://cve.mitre.org.

in the analyzed studies, the large amount of heterogeneous devices, the heterogeneous communication protocols and network conditions (overburdened Wi-Fi channels, unreliable network hardware, and slow or inconsistent Internet connections) [14,78], the different sensitivity levels of data [73], the heterogeneous technologies and data format [76] introduce a lot of IoT scenarios that need to be tested to assess the security of IoT devices and applications as well as the secure communication without loss of data [14]. The extensions of the security models in order to include time information represent a possible solution to allow the verification of scalability and performance properties [49]. Moreover, MBST is more flexible to be adapted to heterogeneous and dynamic environments than conventional testing characterized by manual processes or script-based test automation [77]. The automated generation of test cases and the reusability of the security models that can be enriched with the scalability constraints that the SUT will have to abide by, are two key potential advantages of MBTS to address the impact of scalability and heterogeneity of devices in the testing of the IoT-based systems.

The analysis of the selected primary studies evidenced a **multi-technology convergence** [37,43,53,79] in the security assessment of IoT systems. In more than 30% of the analyzed studies, MBT is combined with other security test strategies such as penetration testing [43,53,74], fuzz testing [79] or learning techniques for model inference [79], in order to discover security issues and to manage the testing requirements during the IoT device's lifecycle. In our vision, the adoption of multiple strategies such as fuzzing, penetration, model learning, represents a research direction that needs to be strengthened in the future for security assessment of IoT systems, to overcome the limitations of individual testing techniques and address the complexity and peculiarities of the IoT scenario at hand.

We think that the analyzed papers in this Rapid Review also miss to address another important challenge of the security validation of the IoT systems that is related to the **growing adoption of open source technologies**. Open-source and open-specification designs are frequently used in IoT technology and pose additional challenges in terms of security and vulnerability detection [91]. Current risk assessment and validation processes, including model-based security testing technologies, need to take into account a much wider landscape of threats of IoT systems related also to the adoption of open hardware. The need in MBST is to consider complex modeling scenarios including an heterogeneity of open software and open hardware components. For designing IoT systems using open hardware, the open source community adopts for instance the RISC-V Instruction Set Architecture (ISA) formalization. Exploiting existing open-source formal ISA semantics could be a first research direction in order to address a wide range of security testing objectives, while also taking into consideration various architecture features (e.g. speculation mechanisms, caching). The obtained model represents itself an open artifact to be used for IoT system designing and testing.

The attacks addressed by the analyzed primary studies are around 40% of the attacks defined in the taxonomy presented in [63]. This partial coverage of the common attack types represents another research gap worthy of further investigation in model-based security testing. Specifically, according to the taxonomy presented in [63], the network and software attacks seem to be the lesser considered attacks by the analyzed studies. This could be due to the widespread adoption of other popular test techniques such as penetration testing that makes possible to detect flaws in the system by executing different types of attacks against the network and software systems. However, penetration testing is not a systematic approach and can be applied after the system development. The idea of future research would be to leverage the advantages of MBT to complement the application of other testing solutions (such as penetration testing) in order to address security problems also during the early stages of IoT system development.

Finally, 3 of the analyzed studies revealed the need to apply the proposed MBST solutions to **concrete and real size case studies** or larger testbeds with a higher heterogeneity of IoT devices to better substantiate the efficiency and scalability of such solutions [76,81,82].

7. Threats to validity of our study

In this section, we present the threats to validity of this Rapid Review. We distinguish the internal, construct and external threats to validity. Regarding the internal validity, the expertise of the authors may have influenced the selection and classification of the primary studies. To reduce this risk, each paper has been randomly assigned to an author for selection and initial classification; moreover each of the selected primary studies has been read by at least two authors. Several meeting have been carried out during the selection process to discuss possible doubts. Finally, the initial classification of all primary studies has been discussed and finalized among all the authors. The adopted classification might not include all the dimensions characterizing the model-based security testing research in the IoT domain. To mitigate this risk, we defined our classification following a bottom-up approach, leveraging the data of the primary studies for defining the main dimensions of our classification.

With respect to the construct validity, i.e., the coherence of the adopted measures to the intended properties, the following potential threats are identified: (i) search string construction. A different set of primary studies may have been derived with a different search string. This threat characterizes all systematic surveys. To mitigate it, we adopted a very general and comprehensive search string; (ii) digital library. We selected the Scopus digital library. We might have missed some relevant papers not available in Scopus. The risk is mitigated since Scopus includes papers belonging to many of the most relevant software engineering digital libraries. Moreover, we followed the guidelines for Rapid Review proposed in [69] and complemented the automated search with backward and forward snowballing over Google Scholar to identify the primary studies; (iii) inclusion and exclusion criteria. There is the possibility that different inclusion/exclusion criteria might have provided different results. Moreover, considering for all the papers their entire content might have produced more accurate results. To mitigate this issue, in our research method we applied the guidelines for systematic reviews in software engineering [35]; (iv) quality of reviewed studies. As we did not conduct a quality assessment phase according to Rapid Review guidelines [69], our study could also cover primary studies that do not fully comply with quality criteria usually applied in literature selection in Systematic Literature Reviews. However, we excluded studies that had not undergone a peer-review process, which should anyhow guarantee the scientific quality of the selected papers.

Finally, external validity concerns potential issues related to the generalization of the results. This is an intrinsic threat of all the review studies, including this one. To the best of our knowledge, this study represents the first systematic review on model-based security testing in IoT domain. Moreover, the threat is mitigated since we considered all papers published until April 2022, thus well representative of current trends.

8. Discussion and conclusion

Devices heterogeneity and large scale objects and networks characterize more and more IoT systems development making the security assessment of the IoT platforms increasingly challenging. In this paper, we proposed the first Rapid Review of literature concerning MBST for IoT. Our systematic search on the Scopus digital library, complemented by snowballing on Google Scholar, led us to select 17 primary studies that have been classified along several dimensions. In the following we present our key findings and hint at interesting directions for future work.

8.1. Summary of key findings

We present below the key findings of our study that summarize the answers to RQ1, RQ2, RQ3, RQ4 and RQ5 respectively:

- Although different **formalisms** have been used for modeling the SUT (see Table 5), we could observe a slight preference for **UML class diagrams** that have been adopted in 7 of the selected works. In all of them but one UML has been complemented by the **OCL language** for specifying the expected dynamic behavior. Variants of **timed automata** are the second most prevalent formalism, used in 5 of the studies. Finally, some authors adopted specialized graph models.
- When performing MBST of IoT, the **test objectives** are of course closely related to security, but involving different aspects, such as **identifying potential vulnerabilities** or vice versa **certifying the provided security levels** against specified risks. In other studies MBST was conducted for verifying the **conformance of the security mechanisms against a formal specification** or for evaluating their performance.
- In MBST, the adopted **test case generation techniques are clearly related with the modeling formalism**, as they are driven by the aim of covering such a model. Hence, with most of the studies adopting UML+OCL, test generation aims at **coverage of the OCL code**, whereas for those using timed automata models, the generation is **based on model-checking techniques**, and most often using the UPPAAL tool. In few studies model-based testing is enforced by applying it in **combination with penetration testing or fuzz testing**. Concerning instead test execution, many different tools are adopted, with a slight prevalence of TITAN.
- MBST has been applied mostly to the domains of **smart city** and **automotive**, although several studies do not explicitly specify a targeted application domain.
- Last but not least we analyzed which types of attacks are addressed, and found that 8 of the studies tackled **Denial of Service**. However, as clearly shown in Table 6, an interesting finding was that almost all of the studies could actually address more than one type of attacks, thus testifying the flexibility and effectiveness of MBST.

Overall we can conclude that MBST can support the validation of IoT systems over a broad range of test objectives in emerging application domains. According to the results of this study, MBST has been applied to core IoT domains and has been proven adequate to assess the security of the system against several well-known IoT attacks.

As a final wrap up, model-based security testing in IoT is a very recent research topic, in fact all the analyzed papers have been published in the last five years. The papers analyzed in this review are in most cases exploratory studies. Therefore, the proposed MBST techniques have not been studied extensively and relevant empirical comparisons on real size IoT case studies are still missing in the literature.

8.2. Future research directions

In our study we also collected a set of open challenges that point to promising research directions. We summarize the research directions answering to RQ6 as follows:

- Probably the very first challenge towards facilitating a smoother adoption of MBST for IoT is that of **extending the existing formalisms** for adequately modeling the peculiarities and constraints of such systems, and thus improving the degree of **testing automation** with regard to the derivation of the test cases from the enhanced system model.
- Among the specific needs that a model should target, we identified as crucial ones the capability to capture **dynamic evolution** and to manage **context-awareness** of IoT systems, as well as ways to address complexity by supporting **modular test modeling**.
- The field of MBST of IoT will certainly mature through the introduction of **standardized technologies** and possibly also the adoption of **open source specifications**.
- Finally, to facilitate research progress more complex and **larger case studies/test-beds** should be available.

Triggered by current state of the art, as analyzed in this Rapid Review, and based on the rich set of remaining open challenges, we expect much more interesting and extensive research to appear in the next years, showing the benefits of applying MBST to IoT complex scenarios.

Moreover, the convergence of model-based security testing with other security test strategies, such as penetration testing or fuzz testing, seems a promising research direction to address the complexity and peculiarities of the IoT scenarios.

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.

Data availability

Data will be made available on request.

Acknowledgments

This work was partially supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU.

References

[1] P. Anand, Y. Singh, A. Selwal, M. Alazab, S. Tanwar, N. Kumar, IoT vulnerability assessment for sustainable computing: threats, current solutions, and open challenges, IEEE Access 8 (2020) 168825–168853, http://dx.doi.org/10.1109/ACCESS.2020.3022842.

[2] I. Nadir, Z. Ahmad, H. Mahmood, G.A. Shah, F. Shahzad, M. Umair, H. Khan, U. Gulzar, An auditing framework for vulnerability analysis of IoT system, in: Proceedings of the IEEE European Symposium on Security and Privacy Workshops, EuroS&PW, IEEE, 2019, pp. 39–47, http://dx.doi.org/10.1109/EuroSPW.2019.00011.

[3] Y. Atwady, M. Hammoudeh, A survey on authentication techniques for the internet of things, in: Proceedings of the International Conference on Future Networks and Distributed Systems, 2017, http://dx.doi.org/10.1145/3102304.3102312.

[4] R. Johari, I. Kaur, R. Tripathi, K. Gupta, Penetration testing in IoT network, in: Proceedings of 5th International Conference on Computing, Communication and Security, ICCCS, IEEE, 2020, pp. 1–7, http://dx.doi.org/10.1109/ICCCS49678.2020.9276853.

[5] S. Siboni, V. Sachidananda, Y. Meidan, M. Bohadana, Y. Mathov, S. Bhairav, A. Shabtai, Y. Elovici, Security testbed for Internet-of-Things devices, IEEE Trans. Reliab. 68 (1) (2019) 23–44, http://dx.doi.org/10.1109/TR.2018.2864536.

[6] M. Utting, A. Pretschner, B. Legeard, A taxonomy of model-based testing approaches, Softw. Test. Verif. Reliab. 22 (5) (2012) 297–312, http://dx.doi.org/10.1002/stvr.456.

[7] V. Garousi, A.B. Keleş, Y. Balaman, Z.Ö. Güler, A. Arcuri, Model-based testing in practice: An experience report from the web applications domain, J. Syst. Softw. 180 (2021) 111032, http://dx.doi.org/10.1016/j.jss.2021.111032.

[8] J. Peleska, J. Brauer, W.-l. Huang, Model-based testing for avionic systems proven benefits and further challenges, in: Proceedings of International Symposium on Leveraging Applications of Formal Methods, Springer, 2018, pp. 82–103, http://dx.doi.org/10.1007/978-3-030-03427-6_11.

[9] B. Morin, N. Harrand, F. Fleurey, Model-based software engineering to tame the IoT jungle, IEEE Softw. 34 (1) (2017) 30–36, http://dx.doi.org/10.1109/MS.2017.11.

[10] J.E. Siegel, S. Kumar, S.E. Sarma, The future internet of things: Secure, efficient, and model-based, IEEE Internet Things J. 5 (4) (2017) 2386–2398, http://dx.doi.org/10.1109/JIOT.2017.2755620.

[11] J.C. Kirchhof, B. Rumpe, D. Schmalzing, A. Wortmann, MontiThings: Model-driven development and deployment of reliable IoT applications, J. Syst. Softw. 183 (2022) 111087, http://dx.doi.org/10.1016/j.jss.2021.111087.

[12] I. Berrouyne, M. Adda, J.-M. Mottu, M. Tisi, A model-driven methodology to accelerate software engineering in the Internet of Things, IEEE Internet Things J. (2022) http://dx.doi.org/10.1109/JIOT.2022.3170500.

[13] G. Fortino, R. Gravina, W. Russo, C. Savaglio, Modeling and simulating Internet-of-Things systems: A hybrid agent-oriented approach, Comput. Sci. Eng. 19 (5) (2017) 68–76, http://dx.doi.org/10.1109/MCSE.2017.3421541.

[14] A. Ahmad, F. Bouquet, E. Fourneret, B. Legeard, Model-based testing for internet of things systems, in: Advances in Computers, Vol. 108, Elsevier, 2018, pp. 1–58, http://dx.doi.org/10.1016/bs.adcom.2017.11.002.

[15] M. Felderer, P. Zech, R. Breu, M. Büchler, A. Pretschner, Model-based security testing: a taxonomy and systematic classification, Softw. Test. Verif. Reliab. 26 (2) (2016) 119–148, http://dx.doi.org/10.1002/stvr.1580.

[16] M. Peroli, F. De Meo, L. Viganò, D. Guardini, MobSTer: A model-based security testing framework for web applications, Softw. Test. Verif. Reliab. 28 (8) (2018) e1685, http://dx.doi.org/10.1002/stvr.1685.

[17] S. Mahmood, H.N. Nguyen, S.A. Shaikh, Systematic threat assessment and security testing of automotive over-the-air (OTA) updates, Veh. Commun. 35 (2022) 100468, http://dx.doi.org/10.1016/j.vehcom.2022.100468.

[18] D.A. Robles-Ramirez, P.J. Escamilla-Ambrosio, T. Tryfonas, IoTsec: UML extension for internet of things systems security modelling, in: Proceedings of International Conference on Mechatronics, Electronics and Automotive Engineering, ICMEAE, IEEE, 2017, pp. 151–156, http://dx.doi.org/10.1109/ICMEAE.2017.20.

[19] C. Bodei, P. Degano, G.-L. Ferrari, L. Galletta, Modelling and analysing IoT systems, J. Parallel Distrib. Comput. 157 (2021) 233–242, http://dx.doi.org/10.1016/j.jpdc.2021.07.004.

[20] F. Anwer, M. Nazir, K. Mustafa, Security testing, in: J.M. H. Mohanty, A. Balakrishnan (Eds.), Trends in Software Testing, Springer, 2017, pp. 35–66, http://dx.doi.org/10.1007/978-981-10-1415-4_3.

[21] G. Murad, A. Badarneh, A. Qusef, F. Almasalha, Software testing techniques in IoT, in: Proceedings of 8th International Conference on Computer Science and Information Technology, CSIT, IEEE, 2018, pp. 17–21, http://dx.doi.org/10.1109/CSIT.2018.8486149.

[22] J.P. Dias, F. Couto, A.C. Paiva, H.S. Ferreira, A brief overview of existing tools for testing the internet-of-things, in: Proceedings of International Conference on Software Testing, Verification and Validation Workshops, ICSTW, IEEE, 2018, pp. 104–109, http://dx.doi.org/10.1109/ICSTW.2018.00035.

[23] M. Cortés, R. Saraiva, M. Souza, P. Mello, P. Soares, Adoption of software testing in internet of things: A systematic literature mapping, in: Proceedings of the IV Brazilian Symposium on Systematic and Automated Software Testing, 2019, pp. 3–11, http://dx.doi.org/10.1145/3356317.3356326.

[24] T. Kh, I. Hamarash, Model-Based Quality Assessment of Internet of Things Software Applications: A Systematic Mapping Study, Int. J. Interact. Mob. Technol. (iJIM) 14 (2020) 128–152, http://dx.doi.org/10.3991/ijim.v14i09.13431.

[25] M. Bures, M. Klima, V. Rechtberger, X. Bellekens, C. Tachtatzis, R. Atkinson, B.S. Ahmed, Interoperability and integration testing methods for IoT systems: A systematic mapping study, in: Proceedings of International Conference on Software Engineering and Formal Methods, Springer, 2020, pp. 93–112, http://dx.doi.org/10.1007/978-3-030-58768-0_6.

[26] S. Mahmood, H.N. Nguyen, S.A. Shaikh, Automotive cybersecurity testing: Survey of testbeds and methods, in: Digital Transformation, Cyber Security and Resilience of Modern Societies, Springer, 2021, pp. 219–243, http://dx.doi.org/10.1007/978-3-030-65722-2_14.

[27] O.B. Tauqeer, S. Jan, A.O. Khadidos, A.O. Khadidos, F.Q. Khan, S. Khattak, Analysis of security testing techniques, Intell. Autom. Soft Comput. 29 (1) (2021) 291–306, http://dx.doi.org/10.32604/iasc.2021.017260.

[28] S. Zhu, S. Yang, X. Gou, Y. Xu, T. Zhang, Y. Wan, Survey of testing methods and testbed development concerning Internet of Things, Wirel. Pers. Commun. 123 (1) (2022) 165–194, http://dx.doi.org/10.1007/s11277-021-09124-5.

[29] F. Heiding, S. Katsikeas, R. Lagerström, Research communities in cyber security vulnerability assessments: A comprehensive literature review, Comp. Sci. Rev. 48 (2023) 100551, http://dx.doi.org/10.1016/j.cosrev.2023.100551.

[30] E. Ahmad, Model-based system engineering of the Internet of Things: A bibliometric literature analysis, IEEE Access (2023) http://dx.doi.org/10.1109/ACCESS.2023.3277429.

[31] F. Sommer, R. Kriesten, F. Kargl, Survey of model-based security testing approaches in the automotive domain, IEEE Access (2023) http://dx.doi.org/10.1109/ACCESS.2023.3282176.

[32] B. Cartaxo, G. Pinto, S. Soares, Rapid reviews in software engineering, in: Contemporary Empirical Methods in Software Engineering, Springer, 2020, pp. 357–384, http://dx.doi.org/10.1007/978-3-030-32489-6_13.

[33] C. Hamel, A. Michaud, M. Thuku, B. Skidmore, A. Stevens, B. Nussbaumer-Streit, C. Garritty, Defining rapid reviews: a systematic scoping review and thematic analysis of definitions and defining characteristics of rapid reviews, J. Clin. Epidemiol. 129 (2021) 74–85, http://dx.doi.org/10.1016/j.jclinepi.2020.09.041.

[34] M. Thelwall, P. Sud, Scopus 1900–2020: Growth in articles, abstracts, countries, fields, and journals, Quant. Sci. Stud. 3 (1) (2022) 37–50, http://dx.doi.org/10.1162/qss_a_00177.

[35] B. Kitchenham, Procedures for Performing Systematic Reviews, Vol. 33, No. 2004, Keele University, Keele, UK, (ISSN: 1353-7776) 2004, pp. 1–26.

[36] N. Medvidovic, D.S. Rosenblum, D.F. Redmiles, J.E. Robbins, Modeling software architectures in the unified modeling language, ACM Trans. Softw. Eng. Methodol. 11 (1) (2002) 2–57, http://dx.doi.org/10.1145/504087.504088.

[37] S.N. Matheu-García, J.L. Hernández-Ramos, A.F. Skarmeta, G. Baldini, Risk-based automated assessment and testing for the cybersecurity certification and labelling of IoT devices, Comput. Stand. Interfaces 62 (2019) 64–83, http://dx.doi.org/10.1016/j.csi.2018.08.003.

[38] J. Cabot, M. Gogolla, Object constraint language (OCL): a definitive guide, in: International School on Formal Methods for the Design of Computer, Communication and Software Systems, Springer, 2012, pp. 58–90, http://dx.doi.org/10.1007/978-3-642-30982-3_3.

[39] A. Ahmad, F. Bouquet, E. Fourneret, F. Le Gall, B. Legeard, Model-based testing as a service for IoT platforms, in: Proceedings of International Symposium on Leveraging Applications of Formal Methods, Springer, 2016, pp. 727–742, http://dx.doi.org/10.1007/978-3-319-47169-3_55.

[40] J. Jürjens, UMLsec: Extending UML for secure systems development, in: Proceedings of International Conference on the Unified Modeling Language, Springer, 2002, pp. 412–425, http://dx.doi.org/10.1007/3-540-45800-X_32.

[41] T. Lodderstedt, D. Basin, J. Doser, Secureuml: A UML-based modeling language for model-driven security, in: Proceedings of International Conference on the Unified Modeling Language, Springer, 2002, pp. 426–441, http://dx.doi.org/10.1007/3-540-45800-X_33.

[42] Y. Mahmoodi, S. Reiter, A. Viehl, O. Bringmann, W. Rosenstiel, Model-guided security analysis of interconnected embedded systems, in: Proceedings of International Conference on Model-Based Software and Systems Engineering, 2018, pp. 602–609, http://dx.doi.org/10.5220/0006724606020609.

[43] Y. Mahmoodi, S. Reiter, A. Viehl, O. Bringmann, W. Rosenstiel, Attack surface modeling and assessment for penetration testing of IoT system designs, in: Proceedings of 21st Euromicro Conference on Digital System Design, DSD, IEEE, 2018, pp. 177–181, http://dx.doi.org/10.1109/DSD.2018.00043.

[44] J. Arcile, É. André, Timed automata as a formalism for expressing security: A survey on theory and practice, ACM Comput. Surv. (2022) http://dx.doi.org/10.1145/3534967.

[45] K.G. Larsen, F. Lorber, B. Nielsen, 20 years of UPPAAL enabled industrial model-based validation and beyond, in: Proceedings of International Symposium on Leveraging Applications of Formal Methods, Springer, 2018, pp. 212–229, http://dx.doi.org/10.1007/978-3-030-03427-6_18.

[46] H.S. Lallie, K. Debattista, J. Bal, A review of attack graph and attack tree visual syntax in cyber security, Comput. Sci. Rev. 35 (2020) 100219, http://dx.doi.org/10.1016/j.cosrev.2019.100219.

[47] M. Krichen, R. Alroobaea, A new model-based framework for testing security of IoT systems in smart cities using attack trees and price timed automata, in: Proceedings of 14th International Conference on Evaluation of Novel Approaches to Software Engineering, SCITEPRESS-Science and Technology Publications, 2019, pp. 570–577, http://dx.doi.org/10.5220/0007830605700577.

[48] V. Gehlot, From Petri NETS to colored Petri NETS: A tutorial introduction to nets based formalism for modeling and simulation, in: Proceedings of Winter Simulation Conference, WSC, 2019, pp. 1519–1533, http://dx.doi.org/10.1109/WSC40007.2019.9004691.

[49] D.C.G. Valadares, Á.A. de Carvalho César Sobrinho, A. Perkusich, K.C. Gorgonio, Formal verification of a trusted execution environment-based architecture for IoT applications, IEEE Internet Things J. 8 (23) (2021) 17199–17210, http://dx.doi.org/10.1109/JIOT.2021.3077850.

[50] U. Khedker, A. Sanyal, B. Sathe, Data Flow Analysis: Theory and Practice, CRC Press, Taylor & Francis Group, 2009, http://dx.doi.org/10.1201/9780849332517.

[51] R. Wirtz, M. Heisel, A systematic method to describe and identify security threats based on functional requirements, in: Proceedings of International Conference on Risks and Security of Internet and Systems, Springer, 2019, pp. 205–221, http://dx.doi.org/10.1007/978-3-030-12143-3_17.

[52] S. Marksteiner, P. Priller, A model-driven methodology for automotive cybersecurity test case generation, in: Proceedings of IEEE European Symposium on Security and Privacy Workshops, EuroS&PW, IEEE, 2021, pp. 129–135, http://dx.doi.org/10.1109/EuroSPW54576.2021.00021.

[53] M. Rak, G. Salzillo, D. Granata, ESSecA: An automated expert system for threat modelling and penetration testing for IoT ecosystems, Comput. Electr. Eng. 99 (2022) 107721, http://dx.doi.org/10.1016/j.compeleceng.2022.107721.

[54] T. Allweyer, BPMN 2.0: Introduction to the Standard for Business Process Modeling, BoD–Books on Demand, ISBN: 978-3-8370-9331-5, 2016.

[55] S. Daoudagh, F. Lonetti, E. Marchetti, XACMET: XACML testing & modeling: An automated model-based testing solution for access control systems, Softw. Qual. J. 28 (1) (2020) 249–282, http://dx.doi.org/10.1007/s11219-019-09470-5.

[56] A. Lunkeit, I. Schieferdecker, Model-based security testing-deriving test models from artefacts of security engineering, in: Proceedings of International Conference on Software Testing, Verification and Validation Workshops, ICSTW, IEEE, 2018, pp. 244–251, http://dx.doi.org/10.1109/ICSTW.2018.00056.

[57] A. Miller, S. Maple, R. Powell, V. Danen, E. Papadopoulos, State of open source security report, Technical Report, Snyk, London, Tel Aviv, Boston, 2020, Retrieved on July 29th, 2023 from https://snyk.io/series/open-source-security/report-2020/.

[58] Y. Shah, S. Sengupta, A survey on classification of cyber-attacks on IoT and IIoT devices, in: Proceedings of the 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference, UEMCON, IEEE, 2020, pp. 406–413, http://dx.doi.org/10.1109/UEMCON51285.2020.9298138.

[59] N. Woolf, DDoS attack that disrupted internet was largest of its kind in history, experts say, Guardian 26 (2016) Retrieved on July 29th, 2023 from https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet.

[60] A. Greenberg, A hacker tried to poison a Florida city's water supply, officials say, 2021, Wired magazine. Retrieved on July 29th, 2023 from https://www.wired.com/story/oldsmar-florida-water-utility-hack.

[61] K. Lounis, M. Zulkernine, Attacks and defenses in short-range wireless technologies for IoT, IEEE Access 8 (2020) 88892–88932, http://dx.doi.org/10.1109/ACCESS.2020.2993553.

[62] S. Khanam, I.B. Ahmedy, M.Y.I. Idris, M.H. Jaward, A.Q.B.M. Sabri, A survey of security challenges, attacks taxonomy and advanced countermeasures in the Internet of Things, IEEE Access 8 (2020) 219709–219743, http://dx.doi.org/10.1109/ACCESS.2020.3037359.

[63] J. Sengupta, S. Ruj, S.D. Bit, A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT, J. Netw. Comput. Appl. 149 (2020) 102481, http://dx.doi.org/10.1016/j.jnca.2019.102481.

[64] L. Xiao, X. Wan, X. Lu, Y. Zhang, D. Wu, IoT security techniques based on machine learning: How do IoT devices use AI to enhance security? IEEE Signal Process. Mag. 35 (5) (2018) 41–49, http://dx.doi.org/10.1109/MSP.2018.2825478.

[65] A.R. Chandan, V.D. Khairnar, Security testing methodology of IoT, in: Proceedings of International Conference on Inventive Research in Computing Applications, ICIRCA, IEEE, 2018, pp. 1431–1435, http://dx.doi.org/10.1109/ICIRCA.2018.8597192.

[66] Z. Gui, H. Shu, F. Kang, X. Xiong, Firmcorn: Vulnerability-oriented fuzzing of IoT firmware via optimized virtual execution, IEEE Access 8 (2020) 29826–29841, http://dx.doi.org/10.1109/ACCESS.2020.2973043.

[67] W. Xie, Y. Jiang, Y. Tang, N. Ding, Y. Gao, Vulnerability detection in IoT firmware: A survey, in: Proceedings of 23rd International Conference on Parallel and Distributed Systems, ICPADS, IEEE, 2017, pp. 769–772, http://dx.doi.org/10.1109/ICPADS.2017.00104.

[68] C.-K. Chen, Z.-K. Zhang, S.-H. Lee, S. Shieh, Penetration testing in the IoT age, Computer 51 (4) (2018) 82–85, http://dx.doi.org/10.1109/MC.2018.2141033.

[69] B. Cartaxo, G. Pinto, S. Soares, The role of rapid reviews in supporting decision-making in software engineering practice, in: Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering, 2018, pp. 24–34, http://dx.doi.org/10.1145/3210459.3210462.

[70] B. Cartaxo, G. Pinto, B. Fonseca, M. Ribeiro, P. Pinheiro, M.T. Baldassarre, S. Soares, Software engineering research community viewpoints on rapid reviews, in: Proceedings of ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, IEEE, 2019, pp. 1–12, http://dx.doi.org/10.1109/ESEM.2019.8870144.

[71] E. Reynen, R. Robson, J. Ivory, J. Hwee, S.E. Straus, A.C. Tricco, et al., A retrospective comparison of systematic reviews with same-topic rapid reviews, J. Clin. Epidemiol. 96 (2018) 23–34, http://dx.doi.org/10.1016/j.jclinepi.2017.12.001.

[72] C. Wohlin, Guidelines for snowballing in systematic literature studies and a replication in software engineering, in: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, 2014, pp. 1–10, http://dx.doi.org/10.1145/2601248.2601268, Art. no. 38.

[73] R. Jabbar, M. Krichen, M. Kharbeche, N. Fetais, K. Barkaoui, A formal model-based testing framework for validating an IoT solution for blockchain-based vehicles communication, in: Proceedings of 15th International Conference on Evaluation of Novel Approaches to Software Engineering. SCITEPRESS-Science and Technology Publications, 2020, pp. 595–602, http://dx.doi.org/10.5220/0009594305950602.

[74] M. Krichen, S. Mechti, R. Alroobaea, E. Said, P. Singh, O.I. Khalaf, M. Masud, A formal testing model for operating room control system using internet of things, Comput. Mater. Continua 66 (3) (2021) 2997–3011, http://dx.doi.org/10.32604/cmc.2021.014090.

[75] S.N. Matheu, J.L. Hernández-Ramos, S. Pérez, A.F. Skarmeta, Extending MUD profiles through an automated IoT security testing methodology, IEEE Access 7 (2019) 149444–149463, http://dx.doi.org/10.1109/ACCESS.2019.2947157.

[76] R. Neisse, G. Baldini, G. Steri, A. Ahmad, E. Fourneret, B. Legeard, Improving internet of things device certification with policy-based management, in: Proceedings of Global Internet of Things Summit, GIoTS, IEEE, 2017, pp. 1–6, http://dx.doi.org/10.1109/GIOTS.2017.8016273.

[77] S. Marksteiner, R. Ramler, H. Sochor, Integrating threat modeling and automated test case generation into industrialized software security testing, in: Proceedings of the Third Central European Cybersecurity Conference, 2019, pp. 1–3, http://dx.doi.org/10.1145/3360664.3362698.

[78] A. Ahmad, G. Baldini, P. Cousin, S.N. Matheu, A. Skarmeta, E. Fourneret, B. Legeard, Large scale IoT security testing, benchmarking and certification, in: Cognitive Hyperconnected Digital Transformation: Internet of Things Intelligence Evolution, ISBN: 9781003337584, 2017, pp. 189–220.

[79] B.K. Aichernig, E. Muškardin, A. Pferscher, Learning-based fuzzing of IoT message brokers, in: Proceedings of 14th Conference on Software Testing, Verification and Validation, ICST, IEEE, 2021, pp. 47–58, http://dx.doi.org/10.1109/ICST49551.2021.00017.

[80] S.N. Matheu, S. Pérez, J.L.H. Ramos, A. Skarmeta, On the automation of security testing for IoT constrained scenarios, in: Proceedings of International Conference on Information Security Applications, Springer, 2019, pp. 286–298, http://dx.doi.org/10.1007/978-3-030-39303-8_22.

[81] M. Krichen, O. Cheikhrouhou, M. Lahami, R. Alroobaea, A.J. Maâlej, Towards a model-based testing framework for the security of internet of things for smart city applications, in: Proceedings of International Conference on Smart Cities, Infrastructure, Technologies and Applications, Springer, 2017, pp. 360–365, http://dx.doi.org/10.1007/978-3-319-94180-6_34.

[82] M. Krichen, M. Lahami, O. Cheikhrouhou, R. Alroobaea, A.J. Maâlej, Security testing of internet of things for smart city applications: A formal approach, in: Smart Infrastructure and Applications, Springer, 2020, pp. 629–653, http://dx.doi.org/10.1007/978-3-030-13705-2_26.

[83] R. Wang, L.M. Kristensen, H. Meling, V. Stolz, Automated test case generation for the Paxos single-decree protocol using a Coloured Petri Net model, J. Log. Algebraic Methods Program. 104 (2019) 254–273, http://dx.doi.org/10.1016/j.jlamp.2019.02.004.

[84] J. Tretmans, On the existence of practical testers, in: ModelEd, TestEd, TrustEd, Springer, 2017, pp. 87–106, http://dx.doi.org/10.1007/978-3-319-68270-9_5.

[85] B. Legeard, A. Bouzy, Smartesting certifyIt: Model-based testing for enterprise IT, in: Proceedings of IEEE Sixth International Conference on Software Testing, Verification and Validation, ICST, IEEE, 2013, pp. 391–397, http://dx.doi.org/10.1109/ICST.2013.55.

[86] C. Willcock, T. Deiß, S. Tobies, S. Keil, F. Engler, S. Schulz, An introduction to TTCN-3, John Wiley & Sons, 2011, http://dx.doi.org/10.1002/9780470977903.

[87] D. Marijan, M. Liaaen, A. Gotlieb, S. Sen, C. Ieva, Titan: Test suite optimization for highly configurable software, in: Proceedings of the IEEE International Conference on Software Testing, Verification and Validation, ICST, IEEE, 2017, pp. 524–531, http://dx.doi.org/10.1109/ICST.2017.60.

[88] R. Lohiya, A. Thakkar, Application domains, evaluation data sets, and research challenges of IoT: A systematic review, IEEE Internet Things J. 8 (11) (2020) 8774–8798, http://dx.doi.org/10.1109/JIOT.2020.3048439.

[89] C. Rudin, Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, Nat. Mach. Intell. 1 (5) (2019) 206–215, http://dx.doi.org/10.1038/s42256-019-0048-x.

[90] N. Carlini, Poisoning the unlabeled dataset of Semi-Supervised learning, in: Proceedings of 30th USENIX Security Symposium, USENIX Security 21, USENIX Association, ISBN: 978-1-939133-24-3, 2021, pp. 1577–1592, Retrieved on July 29th, 2023 from https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-poisoning.

[91] M. Sabbagh, Y. Fei, D. Kaeli, Secure speculative execution via RISC-V open hardware design, in: Proceedings of Fifth Workshop on Computer Architecture Research with RISC-V, CARRV 2021, 2021, pp. 1–7, http://dx.doi.org/10.1145/1122445.1122456.