Attack and System Modeling Applied to IoT, Cloud, and Mobile Ecosystems: Embedding Security by Design

JOÃO B. F. SEQUEIROS, FRANCISCO T. CHIMUCO, MUSA G. SAMAILA, MÁRIO M. FREIRE, and PEDRO R. M. INÁCIO, Universidade da Beira Interior and Instituto de Telecomunicações

Over the years, pervasive computing and communication technologies have enabled the emergence of new computing paradigms that have gained importance across a wide spectrum of domains. The three most notable that have witnessed significant advancements and have a solid track record of exponential growth in diverse applications are the Internet of Things (IoT), Cloud, and Mobile Computing. The ubiquity of these paradigms, their expandability, and applicability in different problem spaces have made them invaluable in modern computing solutions. Security becomes a real concern, especially when it comes to the development of applications in these environments, as numerous security issues may arise from potential design flaws. Secure application development across these three technologies can only be achieved when applications and systems are designed and developed with security in mind. This will improve the quality of the solutions and ensure that vulnerabilities are identified. It will also help in defining countermeasures against cyberattacks or mitigate the effects of potential threats to the systems. This article surveys existing approaches, tools, and techniques for attack and system modeling applicable to IoT, Cloud computing, and Mobile Computing. It also evaluates the strengths and limitations of the reviewed approaches and tools, from which it highlights the main existing challenges and open issues in the area.

CCS Concepts: • **Software and its engineering → Software system models**; **Software notations and tools**; • **Security and privacy** • **Computing methodologies** → *Modeling and simulation*;

Additional Key Words and Phrases: IoT, Cloud computing, Mobile computing, system modeling, attack modeling, security, survey

ACM Reference format:
João B. F. Sequeiros, Francisco T. Chimuco, Musa G. Samaila, Mário M. Freire, and Pedro R. M. Inácio. 2020. Attack and System Modeling Applied to IoT, Cloud, and Mobile Ecosystems: Embedding Security by Design. *ACM Comput. Surv.* 53, 2, Article 25 (March 2020), 32 pages.
https://doi.org/10.1145/3376123

This work is supported by Project SECURIoTESIGN, with funding by FCT-Fundação para a Ciência e Tecnologia (Portugal) through the research grant SFRH/BD/133838/2017, research grant BIM/n°32/2018-B00582, through projects UID/EEA/50008/2019 and POCI-01-0145-FEDER-030657, and by operation Centro-01-0145-FEDER-000019 - C4 - Centro de Competências em Cloud Computing, cofinanced by the European Regional Development Fund (ERDF) through the Programa Operacional Regional do Centro (Centro 2020), in the scope of the Sistema de Apoio à Investigação Científica e Tecnológica - Programas Integrados de IC&DT.

Authors' addresses: J. B. F. Sequeiros, F. T. Chimuco, M. G. Samaila, M. M. Freire, and P. R. M. Inácio, Universidade da Beira Interior, Departamento de Informática, Rua Marquês d'Ávila e Bolama, 6201-001 Covilhã, Portugal; emails: {jbfs, francisco.chimuco}@ubi.pt, mgsamaila@it.ubi.pt, {mario, inacio}@di.ubi.pt.

0360-0300/2020/03-ART25 $15.00
https://doi.org/10.1145/3376123

1 INTRODUCTION

In the early days of computing, computers were large, very expensive, immobile, only locally accessible, and have very limited capabilities. With the advancement of electronics technology and the consequent miniaturization of digital electronics, however, this scenery has significantly changed. Today, computers are portable, affordable, connected, and ubiquitous. In addition, the advent of Mobile technology has made it possible for us to hold minuscule computers in the palm of our hands, or in our pockets, and thus computers can be said to be Mobile [23].

Furthermore, technology trends in computer science and engineering have in recent years revolutionized many aspects of computing technologies, which introduced many new innovations and technologies. Three outstanding examples of these technologies are Cloud and Mobile computing, as well as the Internet of Things (IoT). While profoundly different in their structures and operations, the three have a complementary relationship and can work together to provide unique solutions. For example, Mobile computing enables the end-users of both IoT and the Cloud to have access to devices, services, or resources while in motion. Another important relationship exists particularly between IoT and Cloud computing: as IoT generates massive amounts of data, Cloud computing serves as a means to store or process all that data, the tasks that IoT end-nodes with limited resources are unable to do [50, 107].

The IoT is a network of interrelated devices other than computers, smartphones, or tablets, also referred to as *things*. IoT devices, which include kitchen appliances, cars, industrial machinery, animals, and even implantable medical devices for humans, are uniquely identified on a network and can communicate and transfer data over a network. IoT brings new possibilities for supporting a number of businesses in diverse application domains, which include healthcare, assisted living [54], smart cities [121], agriculture [70], environmental monitoring, and industrial applications [32]. Nonetheless, there are many constraints and challenges associated with the design, development, and deployment of IoT applications, including limited resources, device heterogeneity, and interoperability issues [5, 101, 117].

Unlike the IoT, Cloud computing often called simply *the Cloud* [12] is based on centralized computing resources that can be accessed remotely, and are virtually inexhaustible, being capable of scaling and adapting computing power, storage and access with minimal effort and no interaction from the user. It supports a per-use basis, making it an indispensable business model, removing complexity and acquisition costs. Cloud computing provides different services, such as Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS), or Infrastructure-as-a-Service (IaaS). The first can be described as the use of the Cloud to run software applications that are then interacted with through a standalone application or a web browser. PaaS is related to using the Cloud to create and maintain different applications or services. IaaS, on the other hand, relates to the provisioning of hardware resources, with the user controlling the operating systems, storage, networking, and the like. Cloud also has different deployment models, such as public, private, hybrid, or community clouds, each with its own specific benefits and drawbacks [7]. Figure 1 depicts the aforementioned deployment models. The main advantage of Cloud computing lies in its cost since users do not have to invest in the infrastructure and its maintenance, nor worry about how the infrastructure will sustain usage peaks, as the cloud can scale resources in those situations, reflecting the advantages of elasticity and scalability.

Mobile computing [97] is more focused on consumer technology, enabling users to have computing on-the-go. Mobile systems usually rely on telecommunication infrastructures to enable communication over a distance. Thus, it is essential that models exist that allow the development of systems and applications that are secure by design. This is beneficial not only in terms of development time but also in long-term support, as security issues or flaws are less likely to exist.

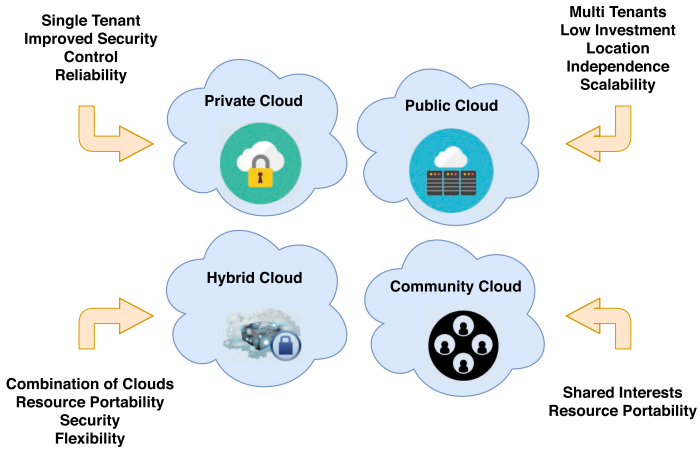

Fig. 1. Example of cloud deployment models (based on [8]).

Figure 2 represents these three ecosystems and how they can relate to one another [1, 25, 35]. As an example scenario, IoT devices can be used to gather data, which is then processed by a cloud service, that will, in turn, allow the end-user to access it on a Mobile device. Many of their characteristics are complementary, and communication between them for different purposes can exist [35]. Different communication technologies and standards (e.g., Wi-Fi, Bluetooth, Near-Field Communication (NFC), General Packet Radio Service (GPRS), Radio-Frequency Identification (RFID), 3G, 4G, ZigBee, etc.) can be used for such [5], with different security measures needing to be assured according to the different situations [102].

Hence, it is essential to have a specific and precise attack and system modeling strategy in place that can be used for analyzing system requirements, used technologies, implementation, and deployment. This will ensure that the system behaves as expected, even when subjected to an attacker.

Considering the aforementioned technologies, their particularities, and different focuses, developing for different platforms constitutes a major challenge, mainly in terms of security [47, 84, 111]. This is aggravated by the time factor since time is essential for introducing new products to the market. Hence, many companies tend to accelerate their product development and time-to-market, and thus security is often left as an afterthought or even nonexistent during the design and development processes [55]. This could cause different security issues in the devices or services, which can result in the exposure of sensitive information through backdoors, inadvertently created by the designer or developer. Therefore, as the use of these technologies becomes ubiquitous, there is a need for designers and developers to bake security into their applications and products right from the beginning, rather than battling it as an afterthought.

Furthering the issue at hand, several authors identify a wide number of attacks and vulnerabilities in these systems. IoT has several technological challenges [20], such as its heterogeneity, interoperability between devices, or energy consumption, and can suffer from a multitude of attacks, either passive (e.g., eavesdropping, node outage and destruction, or traffic analysis) or active (e.g., spoofing, node tampering, node replication, or routing attacks). Despite the existence of several security mechanisms, problems remain, such as the integration between IoT and cloud/fog [21], or the issue of botnets [17], as shown previously. Similarly, the same occurs for Cloud computing [38], with issues that can go from platform security to malware or Virtual Machine (VM) isolation, and spanning topics such as software, storage, virtualization, network, access, trust, and compliance.

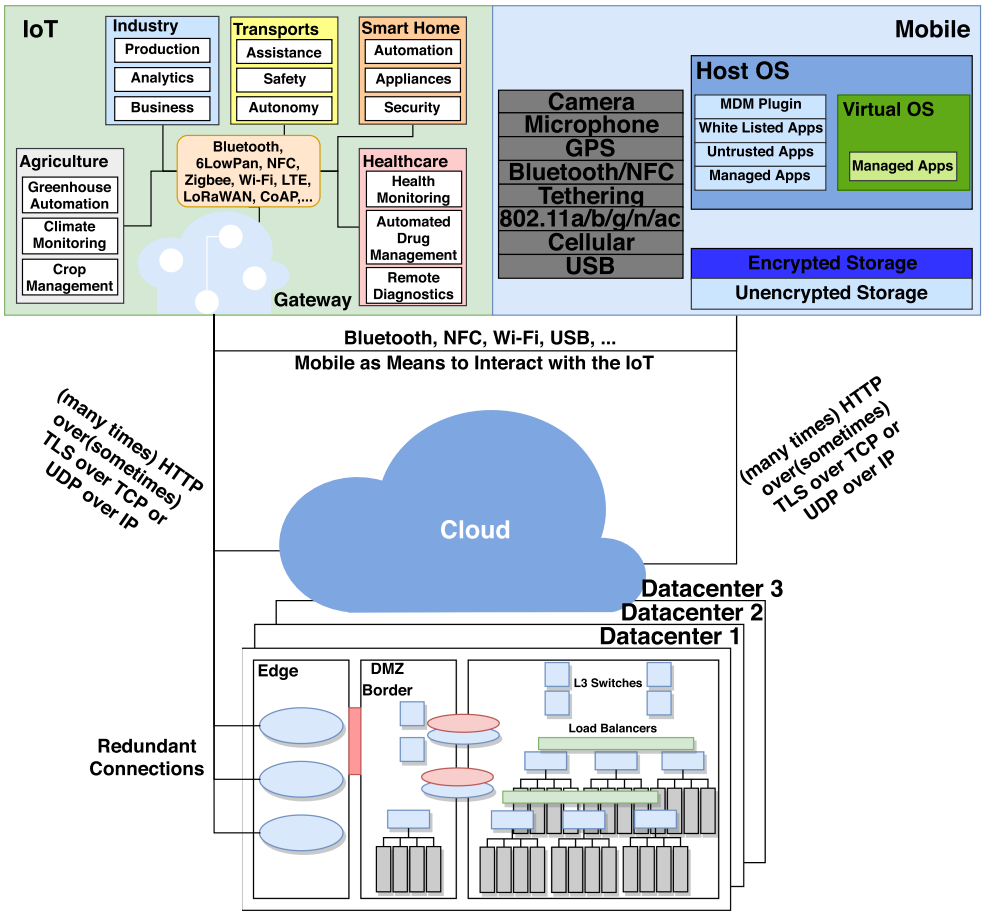

Fig. 2. Representation of the three ecosystems and their potential interactions [1, 25, 35].

Mobile follows a similar trend [84], with issues such as malware, application isolation or botnets. One characteristic common to all these studies is the lack of concise methods that can be applied by developers to resolve the majority of the presented issues. Different solutions exist and have been proposed (e.g., to solve privacy issues in the IoT [113], system models for the Cloud of things [66], or to prevent eavesdropping in IoT communications [118]), and have been verified, but their integration into projects is still not a common practice.

As seen in several threat reports [37, 92, 110], attacks on these systems continue to grow, with new methods and approaches taking advantage of different vulnerabilities. Distributed Denial of Service (DDoS) attacks, borne from malware such as Mirai, Lightaidra, or Kaiten target devices such as routers or IP cameras on a daily basis, taking advantage of default credentials (or lack of credentials). Lack of securing Cloud servers, database-targeting malware and unauthorized accesses have also been commonplace in Cloud environments, as well as malware infections in Mobile devices.

Below are presented some recent issues and attacks that have targeted IoT, Cloud and Mobile systems in recent years:

- *Mirai Botnet* [10, 57]. First appearing in 2016, the Mirai malware infects a large number of devices and transforms them into remote bots, and then uses them to create DDoS attacks.

A number of successful attacks using the Mirai botnet have disrupted the services many of big companies such as Netflix, Twitter, and Reddit, which rendered them unavailable for extended periods of time. The malware works by probing Internet Protocol (IP) addresses for an open Telnet port, and Transmission Control Protocol (TCP) 2323 port. It then performs a simple brute force attack from a set of hard-coded credentials and, if successful, it takes control of the device and reports it to a "controller", which adds it to the main botnet. While it has been around for several years, this malware remains one of the main IoT-directed malware and continues to infect devices on a daily basis.

- *Accenture Data Exposure* [51]. Several Amazon Web Services (AWS) Cloud buckets (a public Cloud storage resource), belonging to Accenture, one of the largest management and consulting firms, were left unsecured, which exposed data of the Cloud platform used by Accenture, including sensitive data from different clients such as databases, credentials, and e-mails. The leaked data could be used to perform attacks on the clients, the company infrastructure, and others.
- *Physical Silicon-Level Vulnerabilities on Central Processing Units (CPUs)* [62, 68]. While not inherent to the Cloud, the recent discovery of hardware-level vulnerabilities, such as Meltdown and Spectre, on CPUs, mainly Intel ones from 2011 and onward, has caused concerns mainly in Cloud environments. These vulnerabilities derived, respectively, from out-of-order execution, a staple in modern processors for performance increase that can be exploited to allow reading kernel-reserved memory spaces (that may contain sensitive data, such as passwords); and speculative execution, another part of modern processor architectures that can be induced into performing instructions that then leak information through a side-channel. These vulnerabilities require a different approach since fixing them requires new silicon revisions, a task that is neither simple nor inexpensive. Patches were developed and implemented to fix these vulnerabilities through software, but with the downside of performance loss in specific operations (e.g., in some Cloud-centric tasks [89]).
- *Abbot Pacemakers* [2]. In 2017, almost 500,000 pacemakers were recalled for a firmware update due to a vulnerability that could allow attackers to either cause a drain of the device battery or alter the pacing given to the heart, eventually leading to health issues or even the death of a patient. No known exploitation of the vulnerability is known, but the risk to human lives was enough to cause the FDA to issue the recall.
- *Triada Android Malware* [3, 42, 64]. Triada, a malware for Android devices, is capable of gaining root access to a device, and then installs apps and inserts malicious code into web browsers to cause spam and show specific adverts to the user, besides stealing sensitive information and adding remote access to the device. While Google blocked Triada-infected apps on its Play Store, the malware has found its way to many devices from the factory because it was installed by third-party vendors in features and apps included in the system images used by Original Equipment Manufacturers (OEMs) to personalize the devices.

As it can be seen, these technologies have been a prime target for attacks over the last few years, with most of the attacks made possible due to either misconfiguration or vulnerabilities that were not either discovered or treated during the development process (despite the existence of security solutions, algorithms, defense mechanisms and applications of security for these ecosystems [43, 45, 50, 56]).

In this article, the state-of-the-art on system and attack modeling for the IoT, Cloud and Mobile computing is surveyed. To the best of our knowledge, other works do not seem to approach the topic based on specific technologies, but rather on a more generic and traditional computing sense, and hence most of their findings and strategies cannot be directly adapted to the specificities of these paradigms. The main contributions of this article, therefore, are:

- a comprehensive survey of different existing architectures, system models, and attack models specific to each paradigm and their applicability in each of them, and how general or specific, and adaptable they can be;
- an overview of the tool landscape that will help in the application of the models and architectures in the design process, their advantages and drawbacks, and limitations;
- an assessment on the main research challenges and work directions for a better design of secure applications and systems for IoT, Cloud and Mobile computing, with the main requirements for comprehensive tool suites of frameworks elucidated through these challenges.

The article is structured as follows: Section 2 presents a review of works that focused on system and attack modeling. Section 3 explores system modeling techniques and architectures, as well as available tools used in modeling processes. Section 4 examines many of the existing attack modeling techniques and tools. Section 5 summarizes the main challenges and potential issues that are identified, as well as highlights future research directions. Finally, Section 6 presents the main conclusions.

2 RELATED WORK

While there are several studies on attack modeling and system modeling techniques, there are no works that seem to focus on their applicability to specific ecosystems, such as IoT, Cloud, and Mobile computing. This section presents a discussion of several attack modeling and system modeling approaches, their main contributions, and limitations.

The OWASP [87] presents an approach to threat modeling in web application systems, which particularly focuses on analyzing web application security. The authors discuss the decomposition of an application to a level that enables the identification of countermeasures to the threats/attacks to the web applications. The STRIDE and DREAD taxonomies of Microsoft are used for this purpose for vulnerability identification and risk analysis, as well as attack trees for attack modeling. It is more of a set of guidelines on the advantages of threat and attack modeling, without a particular technical focus, or tool usage to guide the process.

Tidwell et al. [112] present an approach to modeling Internet attacks using attack trees. Their study proposes an Internet attack model for capturing compound attacks and providing a brief description of a distributed attack via a notification and visualization system. In addition, the model describes the concept of attack with parameters, prerequisites, and post-conditional statements. Dual-specification languages indicate network characteristics and exploits. The approach presented by the authors is geared towards the detection of attacks on an already implemented system, however, which makes its usefulness in the design and development processes limited.

Myagmar et al. [80] present an investigation on how to specify security requirements from threat models. After exploring and comparing distinctions between modeling software or complex systems, the authors introduce their method for identifying threats on network-enabled systems. Three approaches are then presented: a Software Defined Radio, and two tools, one for monitoring network traffic, dubbed VisFlowConnect, and another for monitoring cluster security, dubbed NVisionCC. The authors show how this process can aid in specifying security risks and requirements, and how to manage risk. While the work does not pursue the usage of tools for this process, it refers to a future possibility of the integration of these methodologies into security tools to aid in the definition of security requirements.

Ingalsbe et al. [52] present an overview of the considerations for organizations affected by Cloud computing, converging Mobile devices and consumption tendencies. The authors show how to identify, assess, and mitigate associated risks through threat modeling. They start by providing a definition for Cloud computing and presenting the necessary considerations for consumerization; the authors also discussed Mobile devices and consumption. In addition, threat modeling is

presented as a way to identify, assess, and mitigate risk. Although they recommend the use of the STRIDE and DREAD taxonomy in order to classify and analyze the threats, respectively, their work is presented in a very superficial and general way.

Çamtepe and Yener [22] describe a technique for representing and detecting complex attacks through formal methods. One of the main goals of their approach is the enabling of detection of complex attacks by Intrusion Detection Systems (IDS). The authors also discuss the application of their technique on IEEE 802.11 WLAN security. The presented results validate the claims, with the IDS being able to successfully detect attacks in large noisy input messages.

Another attack modeling technique worthy of note is the work of Pièvre-Cambacédès and Bouissou [90]. The authors present a formal graphic attack modeling technique. This approach adapts Boolean Logic Based on Markov Processes to the security realm, whereas they were previously commonly used in reliability and security engineering. The authors highlight a comparison of the proposed approach with the classic approaches of Petri-Net–based attack tree models.

Amini et al. [8] describe different threat modeling techniques to be applied to Cloud computing. In their study, several approaches to vulnerability and threat modeling were reviewed, where the authors found that none of them is fully suitable for the Cloud computing environment. Thus, the authors proposed a dynamic model for identifying vulnerabilities and threats in Cloud computing environments, capable of determining vulnerabilities and threats and analyzing security risk when using Cloud computing services in a systematic manner, through the identification of threats and their associated vulnerabilities. The proposed model is composed of four main steps, identification of assets, estimation of trustworthiness, threat and vulnerability identification and a final system rating step that identifies the threats and vulnerabilities that pose the greater risk. The work begins describing existing threats and security issues, vulnerabilities, and threat modeling frameworks for the ecosystem under consideration and followed by presenting the results and discussion of the proposed threat model. There is a limitation in proposed the threat model, as identified by the authors and left as future work, namely reliability, mobility, scalability, and flexibility, which are significant in Cloud environments.

Bergmayr et al. [16] present a systematic review on different modeling languages targeting Cloud computing, where the authors analyze the different proposals from the main characteristics, their main purposes, the capabilities that they provide to system engineers and developers, and toolsets associated with the languages made available. From the different surveyed works, the dispersion between different works, the heterogeneity of approaches, a focus on the time aspects of the design process, and the low interoperability between different modeling languages are noticeable, though some convergence is also starting to be observed. The authors finally identify three main future directions, continuous support for the languages (and interoperability between them), the creation of run-time models, and the possibility of simulation deployment configurations. This work is overall a strong in-depth and technical look on the landscape for Cloud modeling languages, but it does not include security in the modeling process.

The attack and system modeling scenarios for IoT, Cloud, and Mobile technologies are vast, and previous works focused only on generic and disperse areas. This work approaches these issues in a specific way, and address several security vulnerabilities and the numerous threats and attacks motivated by their role in IT (Information Technology) and cybersecurity.

Table 1 compares the different works reviewed based on different aspects, namely: focused topics, modeling landscape, tools, and research challenges. Several symbols are used for this comparison on the table: ✔is used to denote the description of a subject on the document, + or ++ are used to emphasize special attention given to a specific aspect, − and × are used for the opposite cases, to denote a less-detailed approach or the absence of the subject, respectively.

Table 1. Comparison of Related Attack and System Modeling Works for IoT, Cloud, and Mobile Ecosystems

Survey	Year	Topic Focused	Attacks Landscape	Modeling Landscape	Tools Landscape	Research Challenges
Tidwell et al. [112]	2001	Attack tree model of Internet attacks, vulnerability analysis, attack modeling	+	+	–	–
Myagmar et al. [80]	2005	Threat modeling, security requirements, networking	+	+	×	×
Çamtepe and Yener [22]	2007	Modeling complex attacks in three, detection complex attacks in three	✔	+	✔	×
Piètre-Cambacédès and Bouissou [90]	2010	Security, attack trees, BDMP, dynamic modeling	✔	++	–	×
Ingalsbe et al. [52]	2011	Threat modeling, Cloud computing, Mobile device security, consumerization	×	++	×	×
Elouali et al. [36]	2013	Multimodal mobile interaction, model-driven engineering, model-based approaches	×	✔	✔	×
Amini et al. [8]	2015	Cloud computing, data security, threat modeling, vulnerability	+	++	✔	×
Bojjagani and Sastry [18]	2017	Security threats, Mobile banking applications, MitM attacks	++	✔	×	✔
OWASP [87]	2017	Threat model, threat modeling	+	++	–	×
Bergmayr et al. [16]	2018	Cloud modeling languages	×	✔	✔	×

The study presented in this article differs from previous works in its broader, yet contextualized, scope. Instead of focusing on technical descriptions and solutions, it provides an ample, high-level description. It is the only work that proposes the discussion of modeling in the three technologies, given their importance in the IT context. It also joins the analysis of tools for IoT, Cloud and Mobile technologies. It allows for the attainment of basic concepts, panorama comprehension and analysis, and attack and system modeling on IoT, Cloud, and Mobile technologies. Furthermore, research challenges are presented, which can be used as a basis for future research.

3 TOOLS AND SYSTEM MODELING ON IOT, CLOUD, AND MOBILE

When designing a system, there is a need to organize and structure the process, and to create a layer of abstraction, so as to reduce the complexity of the process. This allows the specification of system modules, interactions, behavior, and requirements, among others, which ensures that user requirements are met, while increasing the odds of errors being detected early in the development process [74].

It then becomes necessary to integrate security engineering in the software development and engineering processes, to guarantee that systems of these ecosystems can guarantee security by design. In Software Security Engineering, the different security issues are methodically approached as a significant component of software development, with the main objective being the creation of secure software systems. According to Nunes et al. [83], the majority of attacks focus on the application layer, which makes software security issues the main focus of study for security professionals. This has worked as a catalyst for the betterment of software development processes. Devanbu and Stubblebine [33] considers changes in both architectures and development practices enabled through the usage of security engineering practices. Security engineering is then a viable alternative to reduce development time and costs [83]. Through security requirements

engineering, the initial development can be enriched and made less expensive, due to the different techniques rules and methods that it applies [59]. Security by design should then include the use of systematic and repeatable strategies in the development, as a way to ensure that requirements are complete, consistent and easy to understand [75]. Both functional (directly connected to the services) and non-functional (which encompasses characteristics such as performance or security requirements) should be a part of the requirements specification document. Considering how integrated software systems are in day-to-day life, it is vital to ensure that software systems are secure from their early conception [33].

The advent of the Unified Modeling Language (UML) was one of the hallmarks of system and software modeling development. It is a language capable of specifying artifacts that compose a software system by modeling the different objects that define it, and their interactions, which allows for the division of a system into smaller parts or modules that can be worked upon in a simpler fashion [95]. It is not highly specific nor formal, but it is generic, which permits its application in a wide range of different systems. While it is not a programming language, it is constructed in a way that code generators can generate the base structure of the defined models through it. Different variants and offshoots with additional features have evolved over the years, which find applications in different areas. One example is the SecureUML [69], specifically tailored for software security engineering. It combines the usage of UML with the Role-Based Access Control (RBAC) model, which allows for the inclusion of access control, e.g., roles, role permissions, and user-role assignments, into the system model, which in turns allows its usage in the defining of models for access control infrastructures.

Furthermore, other attempts and standards have been created for improving and facilitating system modeling. One example is the Model Driven Architecture (MDA) [11, 79, 114] proposed by the Object Management Group (OMG) as an integration of different standards, including UML, which work in tandem to aid in the designing of a system [61]. This approach attempts to separate design and architecture, so as to create independence and allow for choosing what best applies to the functional requirements more closely related to the system design, and nonfunctional ones, which the architecture will be responsible for resolving [46].

Most, if not all, of the commonly used system modeling tools, are directed towards more traditional computing systems. This section will present and discuss different tools and approaches specifically for IoT, Cloud, and Mobile environments, whose dissimilarities with the more traditional systems create challenges due to their specific architectures and limitations.

3.1 IoT

Morin et al. [78] present a model-based software engineering for the IoT based on a ThingML approach, a modeling language, based on UML, which comprises a modeling language, tools, and methodology. This allows the modeling of IoT systems and architectures without the need for specifying the used technologies or components, which usually are not defined in the early stages of a design process. The integrated tools also enable code generation through a framework that can be tinkered with to be adaptable to the needs and specificities of the system. While the integration of language and tools, including code generation, is one of the main advantages of ThingML, it still requires further work to be applicable to smaller IoT nodes, and applications that require resource sharing or software/firmware updating.

Fortino et al. [40] present a modeling and simulation approach for the IoT based on a framework developed by the same authors, ACOSO, which is capable of modeling smart *things* functionalities and requirements, and on OMNet++, an event simulator capable of simulating interactions and communications between entities in a distributed system. The framework is capable of modeling the behavior and goals of an agent of the system based on events occurrence and system states. The

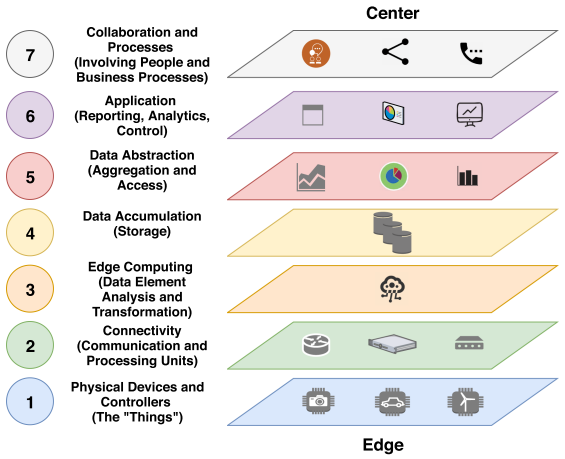

Fig. 3. Representation of the Cisco IoT reference model [27].

event simulator is supported by an extension, INET, capable of simulating wireless nodes in WSNs (Wireless Sensor Networks). This approach is capable of facilitating the design of both the smart objects, or *things*, and the network that they will be connected to, with the simulator facilitating the design process in terms of allowing the testing of different design decisions and their impact on the system. The main limitation of the system, however, is the fact that the system is designed for WSNs and wireless nodes, therefore, its simulations are not easily adaptable for other types of IoT systems and networks.

In [24], the authors propose a model for Smart Sensor Networks (SSN) for Industrial IoT (IIoT) applications. This is a hardware-based model with the purpose of designing efficient hardware controllers for the SSN. Its main advantages are in terms of achieved efficiency in both the design and final implementation. While it is not particularly directed towards designing a secure system, it takes fault detection and error reporting into consideration. Due to the implementation in Field-Programmable Gate Arrays (FPGAs), the proposed solution is not easily reconfigurable and alterable for reuse of the devices in different scenarios.

Cisco proposes an IoT Reference Model [27] with the intent of giving clear definitions and descriptions on what can be applied to IoT systems and applications. It is a 7-layer model: physical devices and controllers, connectivity, edge computing, data accumulation, data abstraction, application and collaboration, and processes. For each layer of the model, there is a detailed description of what it encompasses, as well as several examples of what functionalities the layer represents in the system. Some security considerations are also given, with the type of security required for each layer being described. Figure 3 presents a representation of the model and its layers. The model is, however, a high-level guideline to IoT systems, not focused on more specific implementation details that may be needed in certain scenarios, and the model does lack some greater description of interaction and data flows (and how to best protect them) between the different layers.

The IoT Architectural Reference Model [53] presents several principles and guidelines on the architecture and design of an IoT system, with the purpose of guiding the design process of a system to adhere to a reference model from the generation of the architecture and choice of development methodology to definition of requirements. This can help in achieving better and more cohesive system development, as well as provide economic benefits in the long run. A scheme of the model is present in Figure 4. Compliance with the architecture can allow for the simpler development of support tools and benchmark testing suites, and a comprehensive list of application domains are

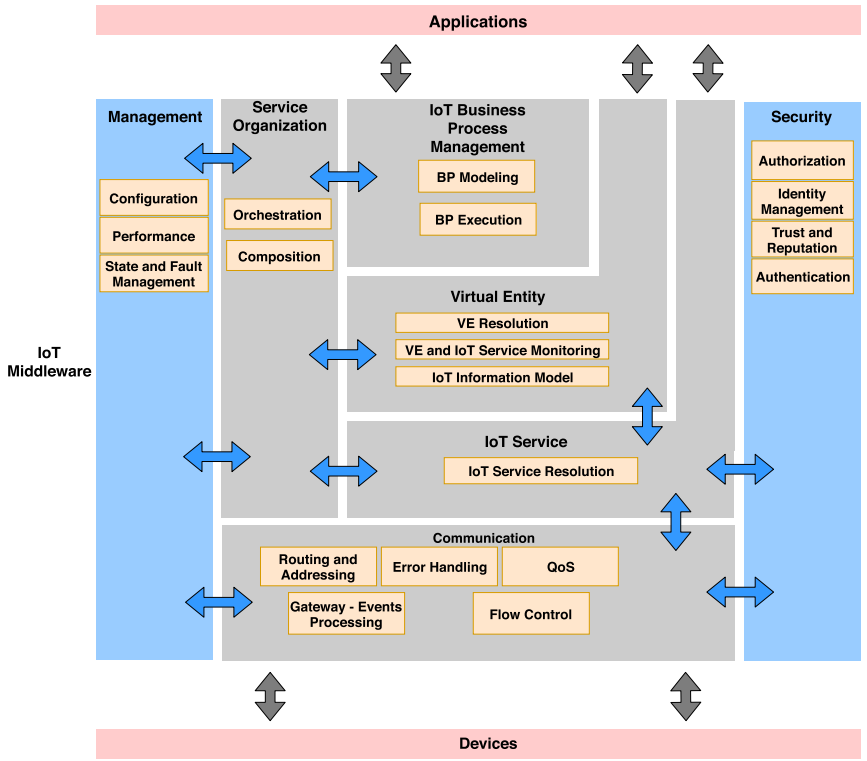

Fig. 4. Representation of the IoT architectural reference model [53].

two of the main advantages of this reference model, with the security emphasis being left as an architectural choice, and not a core part of the architecture / design, as one of the drawbacks of this model.

Breiner et al. [19] define an approach to modeling the IoT based on CT (Category Theory), a mathematical theory of abstract processes. This approach allows for the design of formal diagrams, being adaptable to, e.g., class diagrams, commonly used to design systems. It allows for the designing of logical diagrams capable of formally describing and defining a system and its structure. It is also possible to use it to define and analyze the relations between different components, or between different information models. It is also capable of generating axioms in dynamical systems. Another advantage is that CT can be directly related to functional programming, already used in software development. The main issue associated with the use of CT, according to the authors, is the steep learning curve. For example, making simple examples require advanced mathematical knowledge. This is mainly due to the complexity of the tools used, which are required independent of system complexity. There is also low support for CT in terms of software tools. However, as CT methods are uniform, the same tools can be used over a wide variety of problems and systems.

Nguyen et al. [81] present a framework to aid in the development of IoT applications, mainly focused on sensor-based networks. They propose an architecture for sensor nodes using a rule-based programming model and a DSL to help define the system architecture, which enables abstraction and separation from the chosen Operating System (OS), allowing for the application of the proposed architecture to different applications and scenarios, though mainly restricted to sensor node networks. The decoupling of the programming language from the model and the automated code generation are the two main driving advantages of the framework.

Yan et al. [119] present a system model to help in defining trust management in the IoT with three layers: physical perception, network, and application, which together enable the definition of the interactions between the different layers and associated technologies. From this model, the authors define objectives and properties to achieve trust in an IoT system. Trust is defined for data in different situations and with different characteristics, e.g., communication, processing or storage. The major drawback of this scheme is the apparent lack of integration of security with trust.

Babar et al. [14] propose an IoT security model, built on security requirements gathered from IoT system properties. The authors define the main properties of an IoT system as mobility, wireless, embedded use, diversity, and scale. Based on these properties, they define three main challenges: management and scalability of devices; networked knowledge, context, and privacy: and security and trust. The last challenge, which encompasses security, privacy, and trust, converges in a secure IoT system. In terms of requirements, the authors define several, such as resilience to attacks, data authentication, access control, client privacy, identity management, and tamper resistance. As previously mentioned, the application of these challenges in privacy, security, and trust concepts facilitates defining the secure model of an IoT system. The presented security model is designed around the concept of a cube, due to the intersection of these three dimensions. The model is, therefore, a concept still in the works with a need for a better definition.

Bau and Mitchell [15] present a framework that is capable of defining system security through security modeling. As auxiliary examples, the authors model the NS (Needham-Schroeder) public-key protocol, through a finite-state modeling approach, and other several system components, such as used cryptography protocols, security properties (logical expressions on the state of the model that need to be assured), hardware security (in this case, the Execute Only Memory (XOM) architecture) or web security (for this example, the authors utilize Alloy, a logical language that allows for the creation of models that can then check the properties of the model for their correctness, a more high-level approach than what is used in the NS protocol example). Threat behavior is also modeled for the different components, where attackers are defined on their capabilities and knowledge. From the modeling of each component, results are analyzed to discern the existence of vulnerabilities, which can still be corrected during the design process. The framework is more properly adapted for traditional computing and not for more specific types of computing, such as IoT, though the authors leave open the possibility of that development in the future.

The ETSI M2M [85] develops an architecture standard for M2M IoT communications based on three components: device, network, and application. The idea is based on the standardization of protocols, services, and networks using already existing technologies. Thus, the architecture defines several technical specifications, such as use cases and requirements, data collection, security, and privacy or remote entity management, to assure that a common M2M service layer can be implemented both on a software and hardware level. Figure 5 presents a scheme of the main architecture. Nonetheless, ETSI is an architecture specifically tailored for M2M communications, leaving no room for user interaction with the system.

Lazarescu [65] presents the design of a WSN platform for environmental monitoring. The author proposes a guideline for specifying and optimizing the development creation of WSN platforms for the IoT. From the purpose, objective, and conditions of deployment of the network, several requirements are lifted to create a structure of the platform, composed of four categories: sensor nodes, gateways and repeaters, application servers, and backend and alerts. Based on the requirements, the author proposes the designs for each component of the network, from the sensor nodes to the gateways, deployment devices, and application servers. Availability, reliability, and capacity for recovering from errors are given priority, as well as fast deployment, low cost and low maintenance requirements, properties that are important in the IoT. The design, however, does not take security into account in its design.

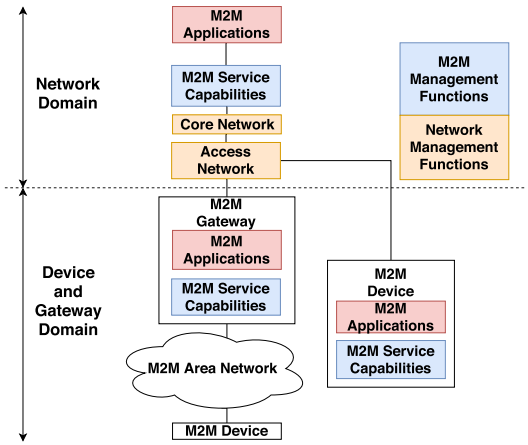

Fig. 5. ETSI M2M architecture [85].

Vučinić et al. [116] propose an architecture to achieve security in the IoT in terms of networking. The authors define a security architecture based on data producers and consumers (constructed on the Constrained Application Protocol (CoAP)), authorization servers, which are trusted entities that manage access to data producers from data consumers, and ensures the trust of producer entities, and proxy servers, which manage availability of producers to consumers when constraints occur. Through this system architecture, the overhead that security protocols imply is removed from the IoT devices, and placed upon external servers, ensuring better performance and maintaining security. Data streaming is the main use case that the proposed architecture does not conform as intended, with the main limitation in the architectural use of CoAP, not adaptable for other communication protocols.

Ye and Qian [120] developed a secure architecture for IoT, based on a security controller, which is capable of security auditing, secure network resource management, device authentication, or traffic encryption, among others, and a monitoring system, capable of monitoring communications between the IoT network of devices and the security controller. The architecture suffers some alterations depending on the deployment scenario (e.g., smart homes, or field-based WSNs), but the security controller remains a central characteristic of the architecture, capable of managing network resources, disconnecting compromised devices, securing communications between the devices and the controller, authentication, or creating a secure link between the network and the end-user. This system allows the assurance of security on a node cluster, for example, without requiring the cluster in itself to have cryptographic capabilities. The requirement of a separate monitoring system and security controller can be seen as an issue in some specific scenarios, especially in terms of costs and ease of configuration.

Sosa-Reyna et al. [104] propose a four-phase methodology for IoT application development, with the phases being system requirements analysis, business logic definition, integrated services solution design, and technological solution generation. This works by taking the initial requirements and modeling the business requirements, which will, in turn, allow for the design of the business solution model that feeds the system architecture model, which will finally generate the technological solution model. This approach also enables the generation of meta-models for more specific types of systems, serving as a structure or base architecture that can be adapted to different implementation platforms. The models are abstract enough to represent different domains of IoT systems and allow for the code generation in different languages, an advantage due to the heterogeneous nature of IoT. The main absence is the integration of security in the modeling process.

In [82], a system architecture for IoT is proposed based on three layers: perception, network, and application, where *things* exist as physical devices, and as cyberentities. These cyberentities are the focus of the work, as their management has a direct implication on the security of the system. Management of the domains in which these identities actuate and of the interactions between them is necessary to assure that security is achieved. As interactions between cyberentities are defined in three phases (preactive, active, and postactive), different security solutions are applied to each phase to achieve a final comprehensive security solution. The architecture defines countermeasures for each category of attacks, but these, as well as the cyberentities referred, are abstract concepts, not specifying which specific methods should be implemented, nor defining the technologies that a system being defined would use.

Robles-Ramirez et al. [94] defines a UML extension for IoT to aid in security modeling, enabling the addition of security modeling in the system modeling phase of development of the design of an IoT system. When compared to other UML extensions that aid in security modeling (such as UMLSec), IoTSec is designed specifically for IoT, being able to represent components, deployment scenarios, communications, activities or system layers. This allows for the integration of security in the design and system modeling processes of development of an IoT system, with the advantage, according to the authors, of not requiring extensive knowledge in cybersecurity in general, and IoT security, specifically, though the work still requires the extension of other tools for UML, such as, e.g., code generation from models, for it to be integrated into a development cycle process.

Ge and Kim [41] present an IoT security modeling and assessment framework, capable of evaluating the security level of a given network when faced with an attack. The framework generates a security model based on the IoT network, generated from the architecture the user inputs, which must account for network topology, its nodes, and vulnerability information. From the security model, attack paths are generated, and security analysis is made by the framework, which outputs a security metric on the network. The last phase updates the network model based on the security analysis previously made, and different strategies can be applied to enable the reevaluation of their effectiveness in the model. The main limitation of the framework is the limited scope in terms of attack targets, defense strategies, and IoT network types, as well as network mobility (e.g., Mobile nodes in smart cities).

Kirichek et al. [60] present a model network, a prototype of a projected network for IoT systems. It consists of five segments, a WSN, a ubiquitous sensor network (composed of sensor nodes that move and collect data), an indoor location system, an SDN (Software Defined Network) to manage network resources, and a Cloud-IoT platform, which enables the end-user to interact with the IoT network, both in terms of configuration, data access, and processing. This platform allows for testing IoT networks with different scenarios and configurations, and different communication technologies. It does not include, from its description, methods for testing attack scenarios or exploiting vulnerabilities in the network, to observe potential different behaviors and defense mechanisms.

3.2 Cloud and Mobile

Parada and De Brisolara [88] present a modeling tool for Android application development using UML to generate models and a code generation tool based on those models. The tool generates models for two different types of views: the structural view, where the activities and services of the application are defined, and the behavioral view, which encompasses interactions and requests of an application, including intents and content provider components. Based on the GenCode tool, a code generation tool was developed for automatically generating code for each class defined in the models. The tool, while able to generate the code, does not generate optimized code, nor takes into account good practices, potentially creating performance or security issues in the generated code.

Ridene and Barbier [93] propose a modeling language, the MATel (Mobile Application Testing Language), for automation of Mobile application checking, which supports the description of tests and scenarios applicable to different Mobile phones, capable of managing the variability in different devices. It allows for the reuse of different scenarios or tests in different applications and is able to remotely control different devices in a testbed through a dedicated middleware that automates the testing process. The platform does not apply, though, to the earlier stages of system design and definition, as it's testing procedures require a working prototype to run through the automated testing.

Usman et al. [114] define a modeling methodology for Mobile applications capable of supporting different platforms. It implements both structural and behavioral modeling, which generate the main logic code for different platforms. This modeling is performed resorting to UML and Action Language for Foundational (ALF), a subset of UML. Through the UML diagrams and state machines, the Mobile Application Generator (MAG) tool generates the code for the specified platform (the current implementation supports Android and Windows Phone). The same diagrams and state machines can then be used to generate application code on every chosen platform. The developed platform is geared mainly towards the business logic and interface modeling of the Mobile application, with the main drawbacks of the system being the lack of security and privacy considerations in the modeling process, and not supporting iOS in its code generation (with it being the second most used Mobile operating system).

Vergara-Vargas and Umaña-Acosta [115] present the basis of a Domain-Specific Language (DSL) for software architecture modeling, specific to distributed architectures for Cloud computing. This language, Sarch, is introduced as a way to help in the design of software architectures. This allows for the definition of models that define an architecture. The models are created from a Model-Driven Deployment approach, which in turn generates transformations from which code can be automatically generated. The architecture design that the tool aids is focused on performance and scalability, important concepts in a Cloud environment, but security is not a part of this modeling process.

Manjunatha et al. [71] present an approach based on DSLs for the development of Cloud-Mobile hybrid applications. The objective is to mitigate many of the problems faced in the development process for hybrid applications in the Cloud and Mobile ecosystem, introducing a DSL-focused approach to generating applications. Generators, through DSL scripts, can build native applications for the distinct Cloud and Mobile platforms. Given the heterogeneous nature of the target platforms, this approach reduces complexity and allows developers to forego the need of understanding the inner working of the platforms, and reduces training time. In addition, the study presents a description of the proposed language and the main gatherings attained from the prototype generator. The main current limitation is in the user interface, with the generated ones not allowing for simple customization through the DSL.

Ranabahu et al. [91] present a DSL for enterprise hybrid applications. In the aforementioned study, the authors present a DSL that is used to generate hybrid applications for use in both Mobile and Cloud solutions. The DSL is then used through a modeling approach in the development process to provide programming abstractions on Cloud and Mobile capabilities. A description of the domain modeling strategy is presented together with the proposed language definition, details and auxiliary tools. The tool allows for quick prototyping without large design efforts, being easily adaptable to an agile development cycle. Currently, there is no support for HTML5 in front-end design, which would allow for simple cross-platform interface design.

Hamdaqa et al. [46] introduce a reference model for Cloud application development that allows developers to have a better understanding of Cloud applications, and that will be used as the foundation for a Cloud modeling language. The model is centered on Cloud Tasks, units that

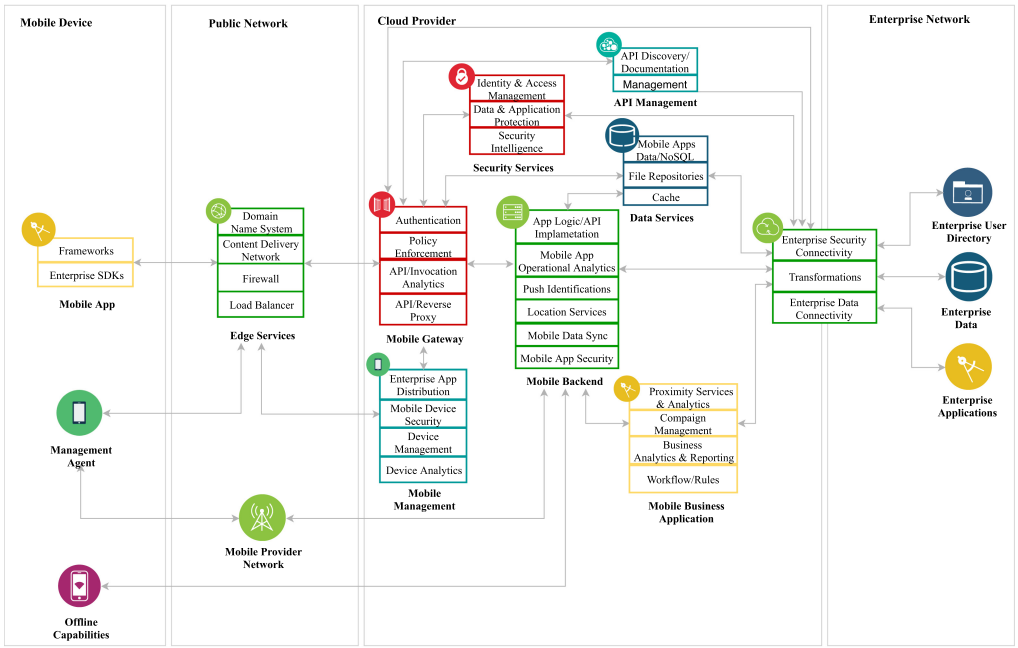

Fig. 6. Cloud customer mobile architecture components (based on [29]).

use services to solve a given problem through their functionalities. These tasks have properties, configurations, and definitions. Tasks can also be of different types, such as Front Tasks that handle user requests, and usually support interaction with users through web applications, or Rotor Tasks, which run in the background and serve as support to other tasks. The current main limitation is the lack of a modeling language to accompany the presented model, as well as the lack of integration of security, privacy, and trust in the reference model.

Heitkötter et al. [48] propose a framework for the development of Mobile applications through a model-driven approach. The authors devised their own DSL to define the application in a high-level abstract manner, which is interpreted by the framework to generate the native code of the application on the selected platform. The language takes a Model View Controller (MVC) approach to structure and organize the application. MVC allows the isolation of user interaction and data processing, allowing both to be separately changed. A detailed description of the MVC pattern is presented by Curry and Grace [30]. The generators, implemented with Xtend, translate the model of the application to object-oriented code, which has the advantage of generating readable code. Currently, the framework implements Android and iOS generators. It is best suitable for designing business-oriented apps, but it can easily be used in the development of a large scope of different apps. Specification of business logic is not yet implemented in the modeling process, and security and privacy are not taken into account in the development process.

Cloud Customer Architecture for Mobile [29] presents Cloud customer Mobile architecture components that illustrate the high-level architecture of a Mobile Cloud solution. The architecture has four tiers, each containing a subset of the components. In addition, each tier of the model, there is a detailed description of what it encompasses, as well as several examples of what functionalities the tier represents in the system. As for IoT, some security considerations are also given, with the type of security required for each layer being described. Figure 6 presents a representation of the model and its tiers.

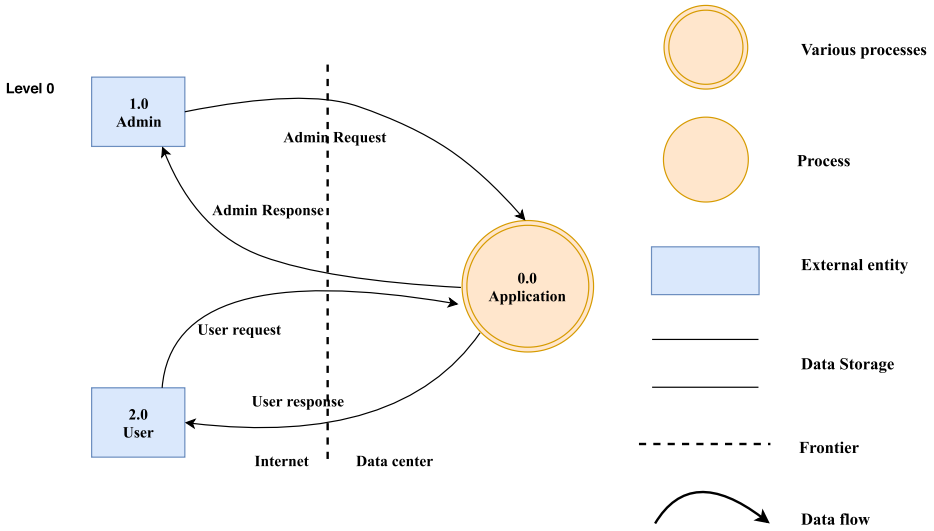

Fig. 7. Representation of the first level of Webmail application decomposition using DFD (based on [28]).

4 TOOLS AND ATTACK AND THREAT MODELING ON IOT, CLOUD, AND MOBILE

While system modeling aids in the design of a system, attack modeling is a structured approach that allows the identification and quantification of security risks in a given system. Its inclusion in the development process ensures that security is incorporated and prioritized from the beginning, and the documentation produced from the modeling process can provide a better overview of the system. Through the information gathered by attack modeling, one can prioritize revisions on the modules with higher threat risk [87]. This is done from the point of view of an attacker, as opposed to threat modeling, which is similar in its goals but works from the perspective of the system trying to defend itself from attacks. Generic risk models can be applied to security and software centered approaches with the purpose of creating a threat priority list to support a risk mitigation strategy, which allows for choosing which threats should be tackled first. Generic risk factors can be used to classify threats as High, Medium, or Low risk. Usually, threat risk models use multiple factors to model risk, as is described in [87].

The attack modeling process can be divided into three high-level stages [87]:

(1) *Application Decomposition.* The purpose of this stage is to understand the behavior of a system and its interaction with external entities. This is done through use cases, identifying areas or items in which an invader would be interested in, and which privileges will be granted by the system to the external entities based on the identified trust levels. Such information is usually included in the documentation of the attack model and then applied to create Data Flow Diagrams (DFD). DFDs display how data flows through a system, emphasizing the access limits [28, 87]. Figure 7 presents an example of a DFD and its respective symbols on level 0 or context diagram, that is, a simple decomposition of a webmail application, identifying the two different types of external entities that interact with a single process (application 0.0) [28]. Figure 8 presents a level-1 detailed decomposition, giving a good idea of the structure of the application, and identifying its principal constituents. The diagram allows for the identification of the most security-related relevant components in the system [28].

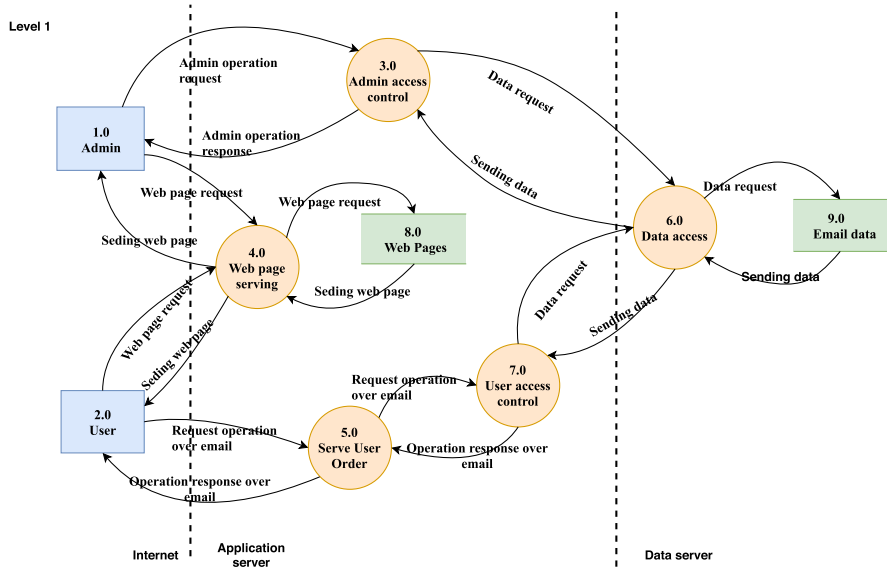

Fig. 8. Representation of the second level of Webmail application decomposition using DFD (based on [28]).

Table 2. STRIDE Threat List (Based on [87])

Type	Examples	Security Control
Spoofing	Illegal access and usage of credentials of other user.	Authentication
Tampering	Modification of persistent data or data in transit in an open network.	Integrity
Repudiation	Performing illegal operations that cannot be tracked.	Non-repudiation
Information disclosure	Accessing information for which access was not allowed.	Confidentiality
Denial of service	Denying a valid user of access to a service.	Availability
Elevation of privilege	Obtaining unauthorized access by privilege escalation in an illegal manner.	Authorization

(2) *Determining and Classifying Attacks.* There are several categorization techniques for threats, such as STRIDE, from Microsoft, which only contains six vulnerability classes. STRIDE is a mnemonic, where each letter corresponds to the first letter of each vulnerability class [28, 98]: Spoofing identity, Tampering with data, Repudiation, Information disclosure, Denial of service, and Elevation of privilege. Classification of invader goals and threat identification is then simplified through the usage of threat categorization, such as STRIDE. Table 2 presents an organized list of generic threats with examples and affected security controls, as defined by STRIDE. Application Security Frame (ASF) is another classification technique, which incorporates different threat classes such as data validation, authorization, auditing and registry, configuration management, authentication, data protection, both in transit and storage or exception management. The main purpose of classifying threats is then the identification of attacks (STRIDE) and threats (ASF). The first-stage DFD helps in identifying potential attack targets from the perspective of the attacker, such as flow and source of data, and user interactions. These attacks can then be used as roots of an attack tree, where an attack goal gives birth to a new tree.

The ASF categorization helps identify attacks from the defender point of view, treating them as security control weaknesses. Use and abuse cases can mirror the way as existing protection measures can be ignored, or where the lack of protection exists. Through the usage of a value-based risk model, e.g., DREAD, or an objective and qualitative risk model (which takes risk factors, such as probability and impact into account), the security risk of each threat can be determined [28, 87]. For the DREAD model, damage and affected users are seen as technical impact risk factors and reproducibility and discovery as ease of exploration factors. The way these risks are factored allows for the assignment of different values to the different factors. In contrast to the attack model creation processes previously described, tools are created for automating those processes. Swiderski [108] created a tool that supports the attack modeling process, allowing the creation of several attack model documents.

(3) *Countermeasure Determining and Mitigation.* A vulnerability may be indicted from the lack of protection measures against an attack, and the employment of countermeasures can thus reduce risk exposure. Attack and countermeasure mapping lists can be used to identify these. As attacks are given a risk classification, it is possible to order threats on risk, and prioritize mitigation efforts on this assessment. A risk mitigation strategy may involve the commercial impact that a threat represents. Other options include risk assumption, taking into consideration acceptable commercial impact, informing the user of the existing threat, or the worse option, take no actions [28, 87]. In [87], a brief and limited verification list is presented which is not, in any way, an exhaustive list to identify countermeasures for specific web application threats.

There are several different threat/attack modeling methodologies, with two from Microsoft, namely, TAM (Threat Analysis and Modeling) and SDL-TM (Software Developed Life-Cycle Threat Modeling). There is also another methodology developed by Swiderki and Snyder [52]. Though these are methodologies more suited for software applications, they present limitations when dealing with interconnected systems [58]. The ETM (Enterprise Threat Modeling) is, therefore, presented in [52] with the purpose of dealing with threats and attacks in the Cloud and Mobile ecosystems. Al-Mohannadi et al. [6] describe several different attack modeling techniques. While not specific to any technology, these techniques can be applied to different network topologies and architectures. Their work focuses on three main modeling techniques, the *diamond model*, the *kill chain*, and the *attack graphs*. The first takes into account four elements: adversary, victim, capability, and infrastructure. The *diamond model* works by assessing the capabilities of the adversary and the victim, and the infrastructure of the victim and technical and logical knowledge of the adversary on that infrastructure, as shown in Figure 9. The kill chain model defines attacks as a chain of actions, followed by an attacker during its actions. The chain is composed of seven steps: reconnaissance, weaponization, delivery, exploitation, installation, command and control, and action on objectives. The first three are part of the first phase of an attack, while the last three being part of the second phase, with the exploitation phase being the focal point on the successfulness of an attack. Figure 10 shows the Kill Chain model. Attack graphs, on the other hand, are used to analyze how a target can be attacked through the usage of diagrams. They are designed as tree graphs, which are used to identify vulnerabilities, attack paths, and actions that can be enforced to stop an attacker from achieving their objective. The authors then apply the three methods to a case study and conclude that each model gives different, yet complementary information on an attack, which can be used to improve a system to prevent such attacks from occurring. Figure 11 shows the attack graphs model, inspired by that of Schneier [100], but the electronic logic gate symbols are used analogously to what is done in the fault trees. Swiderski [34] created a tool that

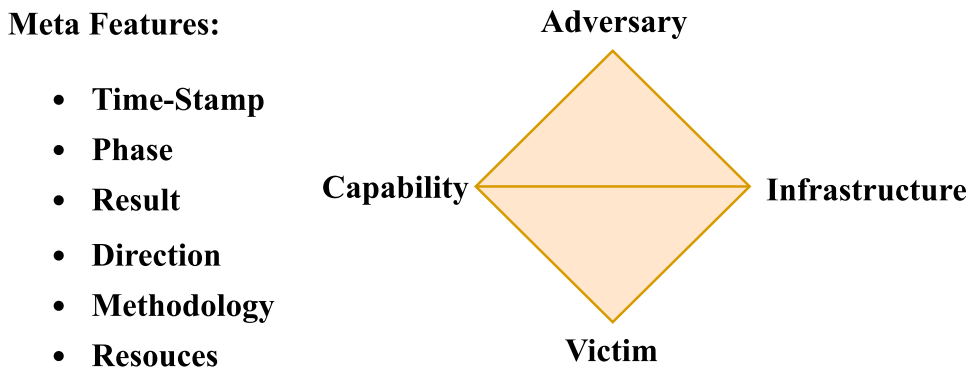

Fig. 9. Diamond model (based on [7]).

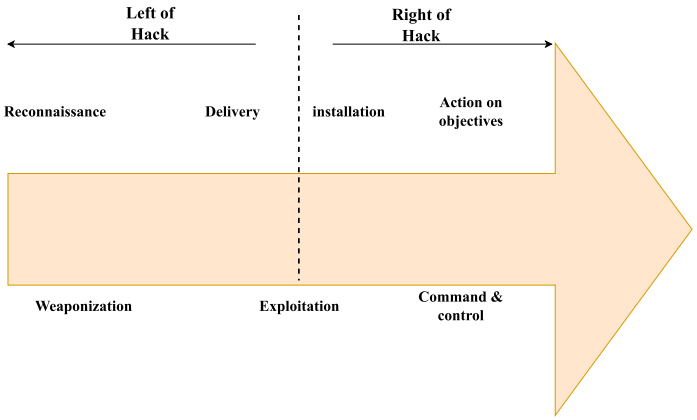

Fig. 10. Kill chain model (based on [7]).

supports the attack modeling process, allowing the creation of several attack model documents. It is freely available at [76], with an improved version with several additional resources, and focused on the use by software developers. IriusRisk [100] is an integrated solution for mainly threat modeling and specification of security requirements, capable of providing insight into all development phases, and even providing a test suite. This type of solution is what would be considered ideal to provide developers with tools to embed security by design, enabling them to create secure applications even without an in-depth knowledge of security. However, this approach can only be applied to more traditional and generic applications, and not to Cloud computing or IoT, which limits its application scope to some extent.

To determine the classification of a threat/attack in Microsoft DREAD, the threat/attack specialist should answer several questions for each risk factor, e.g.:

- *Damage*: What would the extent of the damage be for a successful attack?
- *Reproducibility*: If an attack works, how simple is it to reproduce it?
- *Exploitability*: How much needs to be spent, in terms of time and effort, and what technical experience is needed for the threat to be successfully explored?
- *Affected Users*: How many users would be affected by the exploitation of a threat?
- *Discoverability*: How simple is it for an attacker to uncover the existence of the threat?

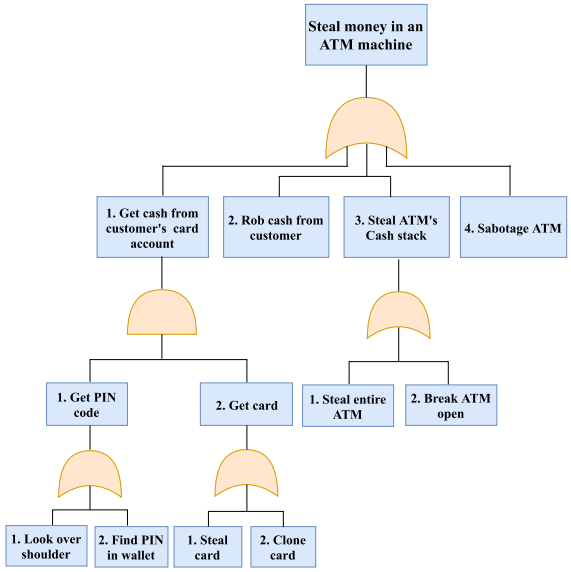

Fig. 11. Attack graphs model (based on [86]).

The risk value is the average of the values attributed to each of the parameters. The values are approximate, having to be coherent between the different vulnerabilities. Values are attributed taking into consideration the detailed description of the potential attacks against vulnerabilities given by the attack trees, introduced by Helmer in [49] ("Attack Tree" was first named by Schneier, making this approach the most known [99]). Other attack modeling mechanisms include *Vulnerability Cause Graphs*, *Misuse Cases* [103], *Petri Nets* [26, 105] (the "Attack Net" model was presented by McDermott in [73]), *Attack Patterns* [39] and, more recently, *Boolean Logic Based on Markov Processes* [90].

With the aforementioned methodologies not specific to any type of computing paradigm, this section will present and discuss threat and attack modeling approaches for IoT, Cloud, and Mobile ecosystems, which take into account the different attack surfaces and their specific limitations and characteristics, to attempt to model the potential attacks and threats that a given system may be subject to, in an effort to supply the design phase in its security modeling processes.

4.1 IoT

Li and Xin [67] present a threat modeling approach for IoT systems, specifically on the perception layer (sometimes referred to as the sensing layer, where sensing nodes operate), using the threat tree model. The authors list the main threats to the perception layer, with a specific focus on WSN, RFID devices, and Mobile terminals, based on which they established a relationship between the identified threats and the corresponding security goals. From the relationships, threat trees are defined for the affected security goals. An excerpt on the threat tree defined for WSNs is presented in Figure 12. This is done in greater depth for RFID and WSN, with preventive or nullifier techniques to be applied to each goal in order to prevent threat proliferation. The threat model does require greater depth in the transport and application layers and also greater detail on the threat model in other types of IoT devices.

Andrea et al. [9] propose an IoT security attack classification system where attacks can be defined as physical, network, software, or encryption-based. They started by describing the main

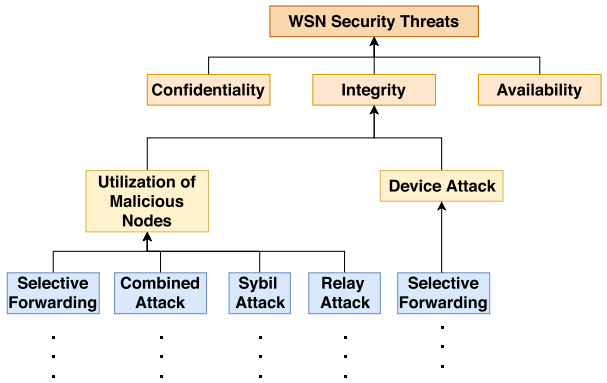

Fig. 12. Excerpt of the threat model scheme for WSNs, as presented by Li and Xin [67].

enabling technologies of IoT, followed by a classification of the basic structure of an IoT device in three layers, Application, Network, and Physical. The authors also define the main security goals, data security and privacy, which can be described as the protection of data collected and stored by a system, and trust, which has different properties, from the trust between the defined layers of a device, trust on the data at each layer, and trust of the user on the system, both in terms of collected data and of data presented to the user. A classification table is presented, which divides the attacks in physical, software, network or encryption attacks. Inside each category, different types of attacks are described (e.g., node tampering for physical attacks, or man-in-the-middle for network attacks). A recommendation is made in terms of trust and protection for all IoT layers, mainly assessing risks, physically securing the devices, managing trust and detect intrusions. The classification system is geared towards a single attack and does not encompass the communication and interactions between different devices (e.g., a node and a gateway).

Agadakos et al. [3] propose an approach to modeling attack paths in IoT systems, by modeling interactions between IoT devices on a network and identifying potential unexpected and harmful events. The system is composed of sniffers and a model checker. Real-Time identification of devices is performed, and the verification of interactions. When compared to defined user policies, it is also kept in real-time, which helps in detecting the impact of the addition or removal of devices in the network, and potential attack paths that may appear as a consequence. The model is verified with recourse to the Alloy model-checking tool. The system is able to discern potential attacks paths from the communication channels it listens to, but has limitations in terms of device detection and fingerprinting, and on the generalization of the threat model used, which models attackers with all types of resources, creating false-positive alerts due to this limitation. An example of an attack path on an IoT system is presented in Figure 13.

Ashraf and Habaebi [13] define autonomous schemes with the purpose of threat mitigation in the IoT. The authors used an autonomic taxonomy to examine threat mitigation approaches in IoT. The authors identify the five main information security goals of a system, confidentiality, integrity, availability, privacy and authenticity. From them, and through the definition of three layers, M2M (Machine-to-Machine), network, and Cloud, their approach suggests the use of autonomic computing and self-security as a way for the systems to autonomously be capable of identifying and mitigating threats, and even recover from attacks of faults (through either self-protection and/or self-healing). Through this approach, a given system can be kept secure even when deployed and left without direct maintenance for extended periods. The main restrictions lie in the required complexity in end devices, which will need enough processing power to be able to perform these

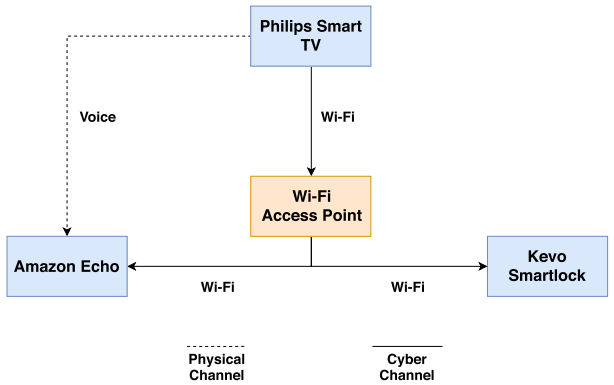

Fig. 13. Example of an attack path on a home IoT system, as described in [3].

autonomous defense tasks, as well as the complexity in terms of software design that such an approach requires.

Stepanova and Zegzhda [106] propose the use of adaptive graphs to model IoT security, where network properties are defined as control, resilience, tenacity, scalability, and operational constancy. The authors provide a formal description of the network through adaptive graphs, based on each of the properties, with the objective of achieving balance between the different devices, so that in a hostile situation, the architecture of the system is capable of adapting itself, to maintain stability and be able to sustain itself, and manage risk and threat exposure. Efficiency is the main consideration to be taken from this approach, as it needs to ensure that connectivity and data integrity are maintained in a hostile environment.

Sarigiannidis et al. [96] introduce a framework for modeling IoT attacks. The model is adaptable to different system architectures and uses a G-network to model the network for two generic types of attacks: light and heavy attacks. This is done by taking into account what is defined as negative arrivals, which can be described as data losses that an attack creates (e.g., a jamming attack, which stops data flow from the sensor network to the gateway), and by calculating a threat impact metric to analyze the impact of an attack on the network. This framework models upon the devices or sensing layer and the communication layer, leaving the application layer for future development. Extensive testing and analysis are done on both the robustness and correctness and the performance of the modeling framework.

Kotenko and Chechulin [63] present a framework capable of modeling and assessing the impact of cyberattacks. The framework is capable of generating attack graphs, performing real-time event analysis, predicting future attacker steps, and assessing the impact of attacks. Attacks are added to the attack graph of the system when three conditions are verified: system vulnerabilities exist, the attacker has the knowledge and resource to perform the attack, and a high probability that the attacker is closer to achieving his/her goal. The attack modeling process on this part cannot be performed in real-time, but the generated graphs are representative during system runtime if no changes occur, which allows for real-time event analysis to be performed based on detecting if a determined event can fit the path of an attack graph. In case of a correct definition on the goal of the attacker, the framework then predicts the next steps the attacker will take, considering the successful attacks and the final goal of the attacker.

Mohsin et al. [77] describe a framework for performing a formal security analysis of IoT systems, capable of modeling how they behave through device configuration, network topology, user policy, and attack surface. The authors attempt to identify threat vectors, attack techniques, and

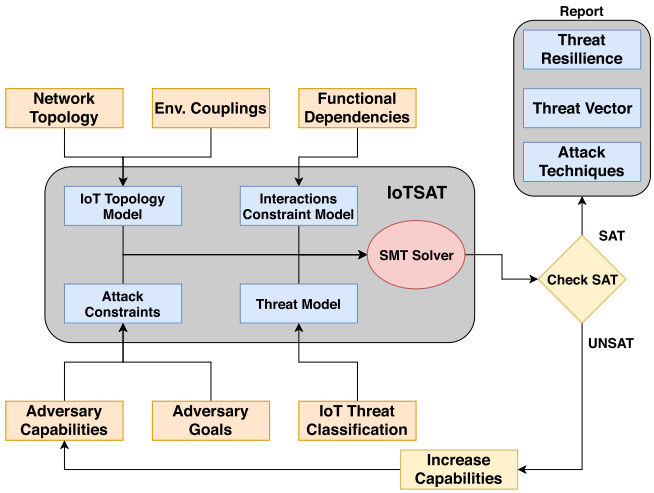

Fig. 14. IoTSAT framework (as described in [77]).

the threat resilience of a system based on the goals of an attacker. The properties of the IoT system consist of attack constraints, threat models, interaction constraints, and topology of the system. The goals are defined by the user based on the security requirements of the evaluated system. The framework utilizes SMT (Satisfiability Modulo Theories), a decision problem for logical formulas in first-order logic (formulas which use quantifiable variables) that attempts to verify that such formula is satisfiable through a logical theory, to formalize interaction between the different entities in the system, and classify system threats. The system scales the capabilities of the attacker until it is able to successfully attack the system. It also identifies the previously referred threat vectors, attack techniques, and threat resilience of the system. The behavior of a system is modeled taking into consideration the environment, the sensors, the controllers, the actuators, and the services of the system. A threat propagation system, which classifies threats in context, triggers and actuates threats, is defined and where threats are modeled taking into consideration the entry point of an attack vector and how it propagates in the system. The authors then present the security analysis their framework performs: first in the defined attack model, and then the verification scenario in which the framework verifies if an attacker achieves its goal, and the steps it takes and the scaling of the capabilities of the attacker needed to achieve its goal. This framework takes into account active threats, created by an attacker actively attempting to breach a system, and is capable of measuring the resilience of a system to attacks, and capable of finding complex attack vectors that a system can be exposed to. A scheme of the framework is presented in Figure 14.

4.2 Cloud and Mobile

Delac et al. [32] describe the emerging security threats for Mobile platforms. The authors present a threat model for two Mobile platforms, Android and iOS, centered on their invaders. The chosen approach takes three main security issues into account: attacker goals, attack vectors, and Mobile malware. Determining the benefits of the attacker and its potential targets is achieved by ascertaining the main motivations of the attacker. This is considered the first step. In the next step, the model comprises the attack vectors that define access points through which malicious content can enter a Mobile device. The last step is for the model to corroborate the application of the threat to Mobile platforms when the attack vectors are successful in their application. The authors then identify and investigate the security model of each operating system (example of

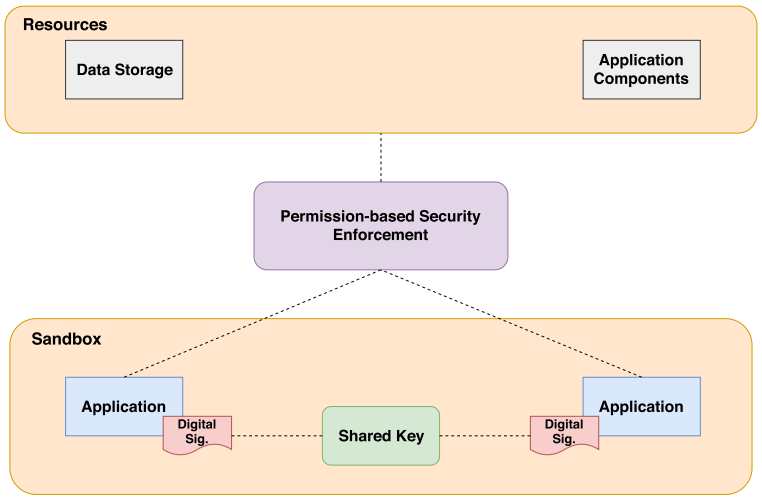

Fig. 15. Android security model (as described in [32]).

the Android security model, as presented by the authors, can be seen in Figure 15, and present an example of a malicious application for Android and how it circumvents the security model. The main part lacking in the document is a more in-depth exploration of the fractures in the two security models, and lack of a strong, similar example of malware for iOS devices.

Kazim and Evans [58] discuss threats and attack modeling on the Cloud. Their work presents threat models for Cloud services and data processing by services. Through this modeling, different threats were identified, as well as potential invaders and threat exploring mechanisms. The authors present different studies on Cloud threat modeling, the role of Cloud services, and the threats and threat-exploring mechanisms that can compromise services and data. The authors, finally, discuss generic threat modeling approaches for the different services on the Cloud ecosystem. Through the identification of attacks and their methods, one can use them as a basis to create or identify security mechanisms to counter these threats, though the paper leaves this approach untouched.

Bojjagani and Sastry [18] present a vulnerability evaluation and penetration testing threat model for Mobile banking apps (iOS and Android). The threat model presented by the authors has the capability to rigorously detect unknown vulnerabilities through a systematic study of unknown threats for Mobile Banking Apps (MBAs). Their study showed that MBAs are vulnerable to Man-in-the-Middle (MitM) attacks. The main contributions of their work are the proposal of a threat model for detection and mitigation of unknown threats, a framework capable of identifying security issues that open up opportunities for MitM attacks, and an analysis on security risks in Mobile banking applications, from those referenced in OWASP, to ascertain their acquiescence with them, and their documentations.

Manzoor et al. [72] present an approach to modeling and analyzing threats to the Cloud ecosystem. The objective of this research was to obtain a holistic analysis of threats in the Cloud, designing a new multilayer Cloud model through Petri Nets, to gauge the behavior during service activities involved in Cloud operations. To achieve this, various tasks were performed, such as threat modeling, to identify threats both inside and in between the various layers of Cloud operations, as well as variants of possible vulnerabilities in ways that broadly presume the attack surface of the Cloud. The major contributions of their work includes the development of a comprehensive multilayer Cloud operational model and rules of information flow between services, and to develop and design a Matrix-based approach to investigate security threats and the attack surfaces

that derive from them through vulnerabilities and vulnerability variants found both inside and in between the different layers of the Cloud model.

The taxonomy of Mobile platform security attacks, as well as their mitigation countermeasures, is dealt with in great detail in [122]. The work focuses on the study of malicious attacks through Bluetooth and malware on Mobile operating systems, mainly Blackberry OS, iOS, Android OS, and Windows Phones. Additionally, the authors propose some countermeasures to security vulnerabilities in Mobile devices.

Suarez-Tangil et al. [109] describe how malware has evolved and can be detected in smartphone devices. The authors presented an exhaustive and complete study of how malware has evolved in recent years to the most popular platforms. They also explored exhibited behaviors, goals pursued, and strategies of infection and distribution of malware. The authors also provided a number of examples through case studies of the most important models. In addition, their work describes security models for smartphones, attack detection taxonomy, how to analyze and detect malware, and how detection systems can be constructed through monitoring of several features of a device. The main issue with these latter approaches, as identified by the authors, is user privacy; because the analysis of a device features could reveal user behaviors and data.

Amara et al. [7] present threats and attacks, along with their mitigation techniques, for Cloud computing environments. Research highlights include Cloud computing architectural principles and main security requirements, for private, hybrid, and public Clouds. Security threats are then presented for different Cloud services that they can affect (some examples include data breaches, malicious insiders or abusive use of Cloud services), as well as the mitigation techniques that can be employed to minimize them. A definition of different security attacks (e.g., phishing, MitM, DoS, DDoS) is presented, with the security level on which they occur and, once again, the mitigation techniques that can be applied to counter them. The definition of security requirements in different Cloud delivery models, and therefore the threats and attacks, as well as the mitigations, that are presented for each, and their distinction, are one the best qualities of the document.

5 RESEARCH CHALLENGES

Several challenges and issues can be deduced from the foregoing discussion and literature review, which can additionally embody bottlenecks to the design of more robust and secure systems. Both in terms of attack and system modeling, in the three environments, several issues consistently appear:

(1) *Lack of Specific Modeling Tools and Designs for IoT, Cloud, and Mobile.* While some attempts have been made to create at least general guidelines and reference models, such as A-IoT and ETSI M2M for IoT, the existence of several different standards hampers the design process and the standardization of the architecture of these systems. Several models are proposed for different types of systems, causing fragmentation on standardization. While several proposals exist, some of which adapt already existing software engineering tools used in more traditional software, as can be seen in IoTSec, which is an UML extension specifically for IoT, they suffer from the same problem already discussed above, with most tools and frameworks being based on models created by the authors, making each tool disparate with one another, as they follow different design philosophies. This is seen as well in Cloud and Mobile environments, with the lack of tools that either approach all Mobile operating systems or Cloud delivery models, or are not complete, especially when the projected system escaped the more general and common types.

(2) *Heterogeneity of Attack Vectors and Taxonomy.* The heterogeneous nature of these computing paradigms exacerbates the existence of threats and different attack vectors, as it

increases the difficulty in developing mechanisms capable of mitigating them that are also applicable and/or scalable to different types of architectures. The way countermeasures can be developed is also affected by the type of each system, running OS, or even communications protocols, processing capacity, and even secondary support services.

(3) *Stronger Focus on Attack Modeling Rather than Threat Modeling.* There is a stronger focus on attack modeling, as in the detection of potential attacks and vulnerability exploitation, and less on threat modeling, which would be more beneficial for embedding security in the design process, as it is what enables developers to properly detect potential security threats during the design process, and design, develop, and adapt mitigation techniques in these early stages of development.

Future research will focus on defining general architectures, as well as providing solutions to the issues raised. The aim will be to facilitate the proper development of solutions that can be used across the three technologies discussed above. The implications of the findings and suggestions for future research directions are:

(1) *Creation of a Framework for Modeling and Application Construction for Cloud, Mobile, and IoT Ecosystems.* While there are different modeling languages, the absence of communication and compatibility between them and the different layers of each referred ecosystem creates the necessity to have standards that can mitigate such faults. The creation of a framework capable of solving these issues is therefore necessary.

(2) *Security Engineering for IoT, Cloud, and Mobile Ecosystems.* This research challenge is critical to eliminate the chasm that exists between software engineering and security engineering, so as to enable security by design in these ecosystems. This challenge should have as its main target the description and proposal of workflows for engineering and development of applications and systems where security modeling and engineering are fully integrated into the software engineering processes.

6 CONCLUSIONS

The growing adoption of Cloud-based Mobile applications and the deployment of IoT systems is becoming commonplace in businesses, industries, and homes. The advantages are many, but security remains one of the main challenges in the adoption of these systems. As it was already discussed, there is currently a large number of opposite or complementary work, with the lack of standardization being the main driver for further development in the area. If concise system architectures are defined, and tools can be developed around it, it will be a great improvement to the design process, which will enable developers to properly incorporate security in their design and development processes. It will also enable the development of threat and attack modeling tools that will allow for more secure environments as a whole, boosting the growth and spread of these technologies without one of its major deterrents. Much of what exists is either afterthoughts on security during the design and development processes, or too specific to be easily adopted by developers on different strains of systems. Finally, there is a need to standardize, simplify and generalize models and tools, so as to allow for more widespread adoption.

ACKNOWLEDGMENTS

The authors wish to thank the Centre for Geodesy and Geodynamics, National Space Research and Development Agency, Toro, Bauchi State, Nigeria for supporting this work, and to thank Édi Marques Aires for his suggestions and help during the elaboration of this work.

REFERENCES

[1] Mohammad Aazam, Imran Khan, Aymen Abdullah Alsaffar, and Eui-Nam Huh. 2014. Cloud of Things: Integrating Internet of Things and cloud computing and the issues involved. In *11th International Bhurban Conference on Applied Sciences and Technology (IBCAST'14)*. IEEE, 414–419.

[2] U.S. Food & Drug Administration. 2017. Firmware Update to Address Cybersecurity Vulnerabilities Identified in Abbott's (formerly St. Jude Medical's) Implantable Cardiac Pacemakers: FDA Safety Communication. https://www.fda.gov/medical-devices/safety-communications/firmware-update-address-cybersecurity-vulnerabilities-identified-abbotts-formerly-st-jude-medicals.

[3] Ioannis Agadakos, Chien-Ying Chen, Matteo Campanelli, Prashant Anantharaman, Monowar Hasan, Bogdan Copos, Tancrède Lepoint, Michael Locasto, Gabriela Felicia Ciocarlie, and Ulf Lindqvist. 2017. Jumping the air gap: Modeling cyber-physical attack paths in the Internet-of-Things. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy*. ACM, 37–48.

[4] Milad Taleby Ahvanooey, Qianmu Li, Mahdi Rabbani, and Ahmed Raza Rajput. 2017. A survey on smartphones security: Software vulnerabilities, malware, and attacks. *Int. J. Adv. Comput. Sci. Appl* 8, 10 (2017), 30–45.

[5] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. 2015. Internet of Things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials* 17, 4 (2015), 2347–2376.

[6] Hamad Al-Mohannadi, Qublai Mirza, Anitta Namanya, Irfan Awan, Andrea Cullen, and Jules Disso. 2016. Cyber-Attack Modeling Analysis Techniques: An Overview. http://hdl.handle.net/10454/10703 Accessed: 2018-07-15.

[7] Naseer Amara, Huang Zhiqui, and Awais Ali. 2017. Cloud computing security threats and attacks with their mitigation techniques. In *2017 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*. 244–251. DOI: https://doi.org/10.1109/CyberC.2017.37

[8] A. Amini, N. Jamil, A. R. Ahmad, and M. R. Z'aba. 2015. Threat modeling approaches for securing cloud computing. *Journal of Applied Sciences* 15, 7 (2015), 953.

[9] Ioannis Andrea, Chrysostomos Chrysostomou, and George Hadjichristofi. 2015. Internet of Things: Security vulnerabilities and challenges. In *IEEE Symposium on Computers and Communication (ISCC'15)*. IEEE, 180–187.

[10] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, et al. 2017. Understanding the Mirai Botnet. In *26th {USENIX} Security Symposium ({USENIX} Security 17)*. 1093–1110.

[11] Danilo Ardagna, Elisabetta Di Nitto, Giuliano Casale, Dana Petcu, Parastoo Mohagheghi, Sébastien Mosser, Peter Matthews, Anke Gericke, Cyril Ballagny, Francesco D'Andria, et al. 2012. Modaclouds: A model-driven approach for the design and execution of applications on multiple clouds. In *4th International Workshop on Modeling in Software Engineering*. IEEE Press, 50–56.

[12] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. 2010. A view of cloud computing. *Commun. ACM* 53, 4 (2010), 50–58.

[13] Qazi Mamoon Ashraf and Mohamed Hadi Habaebi. 2015. Autonomic schemes for threat mitigation in Internet of Things. *Journal of Network and Computer Applications* 49 (2015), 112–127.

[14] Sachin Babar, Parikshit Mahalle, Antonietta Stango, Neeli Prasad, and Ramjee Prasad. 2010. Proposed security model and threat taxonomy for the Internet of Things (IoT). In *International Conference on Network Security and Applications*. Springer, 420–429.

[15] Jason Bau and John C. Mitchell. 2011. Security modeling and analysis. *IEEE Security and Privacy* 9, 3 (2011), 18.

[16] Alexander Bergmayr, Uwe Breitenbücher, Nicolas Ferry, Alessandro Rossini, Arnor Solberg, Manuel Wimmer, Gerti Kappel, and Frank Leymann. 2018. A systematic review of cloud modeling languages. *ACM Comput. Surv.* 51, 1, Article 22 (Feb. 2018), 38 pages. DOI: https://doi.org/10.1145/3150227

[17] Elisa Bertino and Nayeem Islam. 2017. Botnets and internet of things security. *Computer* 2 (2017), 76–79.

[18] S. Bojjagani and V. Sastry. 2017. VAPTAi: A threat model for vulnerability assessment and penetration testing of Android and iOS mobile banking apps. In *2017 IEEE 3rd International Conference on Collaboration and Internet Computing (CIC)*. 77–86. DOI: https://doi.org/10.1109/CIC.2017.00022

[19] Spencer Breiner, Eswaran Subrahmanian, and Ram D. Sriram. 2016. Modeling the Internet of Things: A foundational approach. In *7th International Workshop on the Web of Things*. ACM, 38–41.

[20] Ismail Butun, Patrik Österberg, and Houbing Song. 2019. Security of the Internet of Things: Vulnerabilities, attacks and countermeasures. *IEEE Communications Surveys & Tutorials* (2019).

[21] Ismail Butun, Alparslan Sari, and Patrik Österberg. 2019. Security implications of fog computing on the Internet of Things. In *2019 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 1–6.

[22] S. A. Camtepe and B. Yener. 2007. Modeling and detection of complex attacks. In *2007 3rd International Conference on Security and Privacy in Communications Networks and the Workshops - SecureComm 2007*. 234–243. DOI: https://doi.org/10.1109/SECCOM.2007.4550338

[23] Paul E. Ceruzzi. 2003. *A History of Modern Computing*. MIT press.
[24] Ching-Han Chen, Ming-Yi Lin, and Xing-Chen Guo. 2017. High-level modeling and synthesis of smart sensor networks for Industrial Internet of Things. *Computers & Electrical Engineering* 61 (2017), 48–66.
[25] Mung Chiang and Tao Zhang. 2016. Fog and IoT: An overview of research opportunities. *IEEE Internet of Things Journal* 3, 6 (2016), 854–864.
[26] Farida Chowdhury and Md Sadek Ferdous. [n.d.]. MODELLING CYBER ATTACKS. ([n.d.]).
[27] Cisco. 2014. Cisco IoT Reference Model. http://cdn.iotwf.com/resources/72/IoT_Reference_Model_04_June_2014.pdf. Accessed: 2018-06-24.
[28] Miguel Pupo Correia and Paulo Jorge Sousa. 2017. *Segurança no Software* (2nd ed.). FCA - Editora da Informática, Lda.
[29] Cloud Standards Customer Council. 2015. Cloud Customer Architecture for Mobile. https://www.omg.org/cloud/deliverables/cloud-customer-architecture-for-mobile.htm. Accessed: 2019-02-15.
[30] E. Curry and P. Grace. 2008. Flexible self-management using the model-view-controller pattern. *IEEE Software* 25, 3 (May 2008), 84–90. DOI: https://doi.org/10.1109/MS.2008.60
[31] Li Da Xu, Wu He, and Shancang Li. 2014. Internet of Things in industries: A survey. *IEEE Transactions on Industrial Informatics* 10, 4 (2014), 2233–2243.
[32] G. Delac, M. Silic, and J. Krolo. 2011. Emerging security threats for mobile platforms. In *2011 Proceedings of the 34th International Convention MIPRO*. 1468–1473.
[33] Premkumar T. Devanbu and Stuart Stubblebine. 2000. Software engineering for security: A roadmap. In *Proceedings of the Conference on the Future of Software Engineering (ICSE'00)*. ACM, New York, NY, USA, 227–239. DOI: https://doi.org/10.1145/336512.336559
[34] Mark Dowd, John McDonald, and Justin Schuh. 2006. *The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities*. Pearson Education.
[35] Hanan Elazhary. 2018. Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: Disambiguation and research directions. *Journal of Network and Computer Applications* (2018).
[36] Nadia Elouali, José Rouillard, Xavier Le Pallec, and Jean-Claude Tarby. 2013. Multimodal interaction: A survey from model driven engineering and mobile perspectives. *Journal on Multimodal User Interfaces* 7, 4 (01 Dec 2013), 351–370. DOI: https://doi.org/10.1007/s12193-013-0126-z
[37] F-Secure. 2019. 2019 Attack Landscape Report. https://blog.f-secure.com/attack-landscape-h1-2019-iot-smb-traffic-abound
[38] Diogo A. B. Fernandes, Liliana F. B. Soares, João V. Gomes, Mário M. Freire, and Pedro R. M. Inácio. 2014. Security issues in cloud environments: A survey. *International Journal of Information Security* 13, 2 (2014), 113–170.
[39] Eduardo Fernandez, Juan Pelaez, and Maria Larrondo-Petrie. 2007. Attack patterns: A new forensic and design tool. In *IFIP International Conference on Digital Forensics*. Springer, 345–357.
[40] Giancarlo Fortino, Raffaele Gravina, Wilma Russo, and Claudio Savaglio. 2017. Modeling and simulating internet-of-things systems: A hybrid agent-oriented approach. *Computing in Science & Engineering* 19, 5 (2017), 68–76.
[41] Mengmeng Ge and Dong Seong Kim. 2015. A framework for modeling and assessing security of the internet of things. In *IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS'15)*. IEEE, 776–781.
[42] Google. 2019. PHA Family Highlights: Triada. https://security.googleblog.com/2019/06/pha-family-highlights-triada.html.
[43] Brij Gupta, Dharma P. Agrawal, and Shingo Yamaguchi. 2016. *Handbook of Research on Modern Cryptographic Solutions for Computer and Cyber Security*. IGI global.
[44] B. B. Gupta and Omkar P. Badve. 2017. Taxonomy of DoS and DDoS attacks and desirable defense mechanism in a cloud computing environment. *Neural Computing and Applications* 28, 12 (2017), 3655–3682.
[45] Brij B. Gupta. 2018. *Computer and Cyber Security: Principles, Algorithm, Applications, and Perspectives*. CRC Press.
[46] Mohammad Hamdaqa, Tassos Livogiannis, and Ladan Tahvildari. 2011. A reference model for developing cloud applications. In *CLOSER*. 98–103.
[47] Tobias Heer, Oscar Garcia-Morchon, René Hummen, Sye Loong Keoh, Sandeep S. Kumar, and Klaus Wehrle. 2011. Security challenges in the IP-based Internet of Things. *Wireless Personal Communications* 61, 3 (2011), 527–542.
[48] Henning Heitkötter, Tim A. Majchrzak, and Herbert Kuchen. 2013. Cross-platform model-driven development of mobile applications with MD 2. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. ACM, 526–533.
[49] Guy Helmer, Johnny Wong, Mark Slagell, Vasant Honavar, Les Miller, and Robyn Lutz. 2002. A software fault tree approach to requirements analysis of an intrusion detection system. *Requirements Engineering* 7, 4 (1 Dec 2002), 207–220. DOI: https://doi.org/10.1007/s007660200016

[50] M. Shamim Hossain, Ghulam Muhammad, Wadood Abdul, Biao Song, and B. B. Gupta. 2018. Cloud-assisted secure video transmission and sharing framework for smart cities. *Future Generation Computer Systems* 83 (2018), 596–606.
[51] Upguard Inc. 2019. System Shock: How a Cloud Leak Exposed Accenture's Business. https://www.upguard.com/breaches/cloud-leak-accenture.
[52] Jeffrey A. Ingalsbe, Dan Shoemaker, and Nancy R. Mead. 2011. Threat modeling the cloud computing, mobile device toting, consumerized enterprise-an overview of considerations. In *AMCIS*.
[53] IoT-A. 2013. Introduction to the Architectural Reference Model for the Internet of Things. http://iotforum.org/wp-content/uploads/2014/09/120613-IoT-A-ARM-Book-Introduction-v7.pdf. Accessed: 2018-07-03.
[54] S. M. Riazul Islam, Daehan Kwak, M. D. Humaun Kabir, Mahmud Hossain, and Kyung-Sup Kwak. 2015. The Internet of Things for health care: A comprehensive survey. *IEEE Access* 3 (2015), 678–708.
[55] Ajit Jha and M. C. Sunil. 2014. Security considerations for Internet of Things. *L&T Technology Services* (2014).
[56] Feng Jiang, Yunsheng Fu, Brij B. Gupta, Fang Lou, Seungmin Rho, Fanzhi Meng, and Zhihong Tian. 2018. Deep learning based multi-channel intelligent attack detection for data security. *IEEE Transactions on Sustainable Computing* (2018).
[57] Georgios Kambourakis, Constantinos Kolias, and Angelos Stavrou. 2017. The Mirai Botnet and the IoT zombie armies. In *MILCOM 2017-2017 IEEE Military Communications Conference (MILCOM)*. IEEE, 267–272.
[58] M. Kazim and D. Evans. 2016. Threat modeling for services in cloud. In *2016 IEEE Symposium on Service-Oriented System Engineering (SOSE)*. 66–72. DOI: https://doi.org/10.1109/SOSE.2016.55
[59] Jintae Kim, Minseong Kim, and Sooyong Park. 2006. Goal and scenario based domain requirements analysis environment. *Journal of Systems and Software* 79, 7 (2006), 926–938. DOI: https://doi.org/10.1016/j.jss.2005.06.046 Selected papers from the 11th Asia Pacific Software Engineering Conference (APSEC2004).
[60] Ruslan Kirichek, Andrei Vladyko, Maxim Zakharov, and Andrey Koucheryavy. 2016. Model networks for Internet of Things and SDN. In *18th International Conference on Advanced Communication Technology (ICACT'16)*. IEEE, 76–79.
[61] Anneke G. Kleppe, Jos Warmer, Jos B. Warmer, and Wim Bast. 2003. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Professional.
[62] Paul Kocher, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. 2018. Spectre attacks: Exploiting speculative execution. *arXiv preprint arXiv:1801.01203* (2018).
[63] Igor Kotenko and Andrey Chechulin. 2013. A cyber attack modeling and impact assessment framework. In *5th International Conference on Cyber Conflict (CyCon'13)*. IEEE, 1–24.
[64] AO Kaspersky Lab. 2016. Kaspersky Threats - Triada. https://threats.kaspersky.com/en/threat/Trojan.AndroidOS.Triada/.
[65] Mihai T. Lazarescu. 2013. Design of a WSN platform for long-term environmental monitoring for IoT applications. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 3, 1 (2013), 45–54.
[66] Wei Li, Igor Santos, Flavia C. Delicato, Paulo F. Pires, Luci Pirmez, Wei Wei, Houbing Song, Albert Zomaya, and Samee Khan. 2017. System modelling and performance evaluation of a three-tier Cloud of Things. *Future Generation Computer Systems* 70 (2017), 104–125.
[67] Zhang Li and Tong Xin. 2013. Threat modeling and countermeasures study for the Internet of Things. *Journal of Convergence Information Technology* 8, 5 (2013).
[68] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. 2018. Meltdown. *arXiv preprint arXiv:1801.01207* (2018).
[69] Torsten Lodderstedt, David Basin, and Jürgen Doser. 2002. SecureUML: A UML-based modeling language for model-driven security. In *International Conference on the Unified Modeling Language*. Springer, 426–441.
[70] Junyan Ma, Xingshe Zhou, Shining Li, and Zhigang Li. 2011. Connecting agriculture to the Internet of Things through sensor networks. In *2011 IEEE International Conferences on Internet of Things, and Cyber, Physical and Social Computing*. IEEE, 184–187.
[71] Ashwin Manjunatha, Ajith Ranabahu, Amit Sheth, and Krishnaprasad Thirunarayan. 2010. A domain specific language based method to develop cloud-mobile hybrid applications. *Kno. e. sis Center Wright State University* (2010), 50–60.
[72] S. Manzoor, H. Zhang, and N. Suri. 2018. Threat modeling and analysis for the cloud ecosystem. In *2018 IEEE International Conference on Cloud Engineering (IC2E)*. 278–281. DOI: https://doi.org/10.1109/IC2E.2018.00056
[73] J. P. McDermott. 2000. Attack net penetration testing. In *2000 Workshop on New Security Paradigms (NSPW'00)*. ACM, New York, 15–21. DOI: https://doi.org/10.1145/366173.366183
[74] Nenad Medvidovic and Richard N. Taylor. 2010. Software architecture: Foundations, theory, and practice. In *32nd ACM/IEEE International Conference on Software Engineering, Volume 2*. ACM, 471–472.
[75] Daniel Mellado, Carlos Blanco, Luis E.Sánchez, and Eduardo Fernández-Medina. 2010. A systematic review of security requirements engineering. *Computer Standards & Interfaces* 32, 4 (2010), 153–165. DOI: https://doi.org/10.1016/j.csi.2010.01.006

[76] Microsoft. 2019. Microsoft Threat Modeling Tool 2016. https://www.microsoft.com/en-us/download/details.aspx?id=49168. Accessed: 2019-02-04.
[77] Mujahid Mohsin, Zahid Anwar, Ghaith Husari, Ehab Al-Shaer, and Mohammad Ashiqur Rahman. 2016. IoTSAT: A formal framework for security analysis of the Internet of Things (IoT). In *IEEE Conference on Communications and Network Security (CNS'16)*. IEEE, 180–188.
[78] Brice Morin, Nicolas Harrand, and Franck Fleurey. 2017. Model-based software engineering to tame the IoT jungle. *IEEE Software* 34, 1 (2017), 30–36.
[79] Francesco Moscato, Beniamino Di Martino, and Rocco Aversa. 2012. Enabling model driven engineering of cloud services by using mosaic ontology. *Scalable Computing: Practice and Experience* 13, 1 (2012), 29–44.
[80] Suvda Myagmar, Adam J. Lee, and William Yurcik. 2005. Threat modeling as a basis for security requirements. In *Symposium on Requirements Engineering for Information Security (SREIS)*. Citeseer, 1–8.
[81] Xuan Thang Nguyen, Huu Tam Tran, Harun Baraki, and Kurt Geihs. 2015. FRASAD: A framework for model-driven IoT application development. In *IEEE 2nd World Forum on Internet of Things (WF-IoT'15)*. IEEE, 387–392.
[82] Huansheng Ning, Hong Liu, and Laurence Yang. 2013. Cyber-entity security in the Internet of Things. *Computer* (2013), 1.
[83] Francisco José Barreto Nunes, Arnaldo Dias Belchior, and Adriano Bessa Albuquerque. 2010. Security engineering approach to support software security. In *2010 6th World Congress on Services*. 48–55. DOI: https://doi.org/10.1109/SERVICES.2010.37
[84] Jon Oberheide and Farnam Jahanian. 2010. When mobile is harder than fixed (and vice versa): Demystifying security challenges in mobile environments. In *11th Workshop on Mobile Computing Systems & Applications*. ACM, 43–48.
[85] Hersent Olivier, Boswarthick David, and Omar Elloumi. 2011. *The ETSI M2M Architecture*. Wiley-Blackwell, Chapter 14, 237–267. DOI: https://doi.org/10.1002/9781119958352.ch14 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119958352.ch14
[86] Andreas L. Opdahl and Guttorm Sindre. 2009. Experimental comparison of attack trees and misuse cases for security threat identification. *Information and Software Technology* 51, 5 (2009), 916–932.
[87] Open Web Application Security Project OWASP. 2017. Application Threat Modeling. https://www.owasp.org/index.php/Application_Threat_Modeling.
[88] Abilio G. Parada and Lisane B. De Brisolara. 2012. A model driven approach for android applications development. In *Brazilian Symposium on Computing System Engineering (SBESC'12)*. IEEE, 192–197.
[89] Phoronix. 2019. The Performance Impact of MDS / Zombieload Plus the Overall Cost Now of Spectre/Meltdown/L1TF/MDS. https://www.phoronix.com/scan.php?page=article&item=mds-zombieload-mit&num=1.
[90] L. Piètre-Cambacédès and M. Bouissou. 2010. Beyond attack trees: Dynamic security modeling with Boolean Logic Driven Markov Processes (BDMP). In *2010 European Dependable Computing Conference*. 199–208. DOI: https://doi.org/10.1109/EDCC.2010.32
[91] Ajith H. Ranabahu, Eugene Michael Maximilien, Amit P. Sheth, and Krishnaprasad Thirunarayan. 2011. A domain specific language for enterprise grade cloud-mobile hybrid applications. In *Compilation of the Co-located Workshops on DSM'11, TMC'11, AGERE! 2011, AOOPES'11, NEAT'11, & VMIL'11 (SPLASH'11 Workshops)*. ACM, New York, 77–84. DOI: https://doi.org/10.1145/2095050.2095064
[92] Check Point Research. 2019. Securing the Cloud, Mobile and Internet of Things. http://snt.hr/news/pressroom/pressreleases/CP2019SecurityReportVolume03.pdf.
[93] Youssef Ridene and Franck Barbier. 2011. A model-driven approach for automating mobile applications testing. In *5th European Conference on Software Architecture: Companion Volume*. ACM, 9.
[94] David Alejandro Robles-Ramirez, Ponciano Jorge Escamilla-Ambrosio, and Theo Tryfonas. 2017. IoTsec: UML extension for Internet of Things systems security modelling. In *International Conference on Mechatronics, Electronics and Automotive Engineering (ICMEAE'17)*. IEEE, 151–156.
[95] James Rumbaugh, Ivar Jacobson, and Grady Booch. 2010. *Unified Modeling Language Reference Manual* (2nd ed.). Addison-Wesley Professional.
[96] Panagiotis Sarigiannidis, Eirini Karapistoli, and Anastasios A. Economides. 2017. Modeling the Internet of Things under attack: A G-network approach. *IEEE Internet of Things Journal* 4, 6 (2017), 1964–1977.
[97] Mahadev Satyanarayanan. 1996. Fundamental challenges in mobile computing. In *15th Annual ACM Symposium on Principles of Distributed Computing*. ACM, 1–7.
[98] Riccardo Scandariato, Kim Wuyts, and Wouter Joosen. 2015. A descriptive study of Microsoft's threat modeling technique. *Requirements Engineering* 20, 2 (2015), 163–180.
[99] Bruce Schneier. 1999. Attack trees. *Dr. Dobb's Journal* 24, 12 (1999), 21–29.
[100] Continuum Security. 2019. IriusRisk - threat modeling tool. https://continuumsecurity.net/threat-modeling-tool/. Accessed: 2019-01-22.

[101] Zhengguo Sheng, Shusen Yang, Yifan Yu, Athanasios Vasilakos, Julie Mccann, and Kin Leung. 2013. A survey on the IETF protocol suite for the Internet of Things: Standards, challenges, and opportunities. *IEEE Wireless Communications* 20, 6 (2013), 91–98.
[102] Sabrina Sicari, Alessandra Rizzardi, Luigi Alfredo Grieco, and Alberto Coen-Porisini. 2015. Security, privacy and trust in Internet of Things: The road ahead. *Computer Networks* 76 (2015), 146–164.
[103] Guttorm Sindre and Andreas L. Opdahl. 2005. Eliciting security requirements with misuse cases. *Requirements Engineering* 10, 1 (2005), 34–44.
[104] Claudia M. Sosa-Reyna, Edgar Tello-Leal, and David Lara-Alabazares. 2018. Methodology for the model-driven development of service oriented IoT applications. *Journal of Systems Architecture* 90 (2018), 15–22.
[105] Jan Steffan and Markus Schumacher. 2002. Collaborative attack modeling. In *2002 ACM Symposium on Applied Computing (SAC'02)*. ACM, New York, 253–259. DOI: https://doi.org/10.1145/508791.508843
[106] Tatiana Stepanova and D. Zegzhda. 2014. Applying large-scale adaptive graphs to modeling Internet of Things security. In *7th International Conference on Security of Information and Networks*. ACM, 479.
[107] Christos Stergiou, Kostas E. Psannis, Byung-Gyu Kim, and Brij Gupta. 2018. Secure integration of IoT and cloud computing. *Future Generation Computer Systems* 78 (2018), 964–975.
[108] Frank Swiderski and Window Snyder. 2004. *Threat Modeling*. Microsoft Press.
[109] Guillermo Suarez-Tangil, Juan E. Tapiador, Pedro Peris-Lopez, and Arturo Ribagorda. 2014. Evolution, detection and analysis of malware for smart devices. *IEEE Communications Surveys Tutorials* 16, 2 (2014), 961–987. DOI: https://doi.org/10.1109/SURV.2013.101613.00077
[110] Symantec. 2019. 2019 Internet Security Threat Report. https://www.symantec.com/en/uk/security-center/threat-report.
[111] Hassan Takabi, James B.D. Joshi, and Gail-Joon Ahn. 2010. Security and privacy challenges in cloud computing environments. *IEEE Security & Privacy*6 (2010), 24–31.
[112] T. Tidwell, R. Larson, K. Fitch, and J. Hale. 2001. Modeling internet attacks. In *2001 IEEE Workshop on Information Assurance and Security*, Vol. 59. United States Military Academy West Point, NY.
[113] Ikram Ullah, Munam Ali Shah, Abdul Wahid, Amjad Mehmood, and Houbing Song. 2018. ESOT: A new privacy model for preserving location privacy in Internet of Things. *Telecommunication Systems* 67, 4 (2018), 553–575.
[114] Muhammad Usman, Muhammad Zohaib Iqbal, and Muhammad Uzair Khan. 2014. A model-driven approach to generate mobile applications for multiple platforms. In *Software Engineering Conference (APSEC), 2014 21st Asia-Pacific*, Vol. 1. IEEE, 111–118.
[115] Jeisson Vergara-Vargas and Henry Umaña-Acosta. 2017. A model-driven deployment approach for scaling distributed software architectures on a cloud computing platform. In *8th IEEE International Conference on Software Engineering and Service Science (ICSESS'17)*. IEEE, 99–103.
[116] Mališa Vučinić, Bernard Tourancheau, Franck Rousseau, Andrzej Duda, Laurent Damon, and Roberto Guizzetti. 2015. OSCAR: Object security architecture for the Internet of Things. *Ad Hoc Networks* 32 (2015), 3–16.
[117] Andrew Whitmore, Anurag Agarwal, and Li Da Xu. 2015. The Internet of Things - A survey of topics and trends. *Information Systems Frontiers* 17, 2 (2015), 261–274.
[118] Qian Xu, Pinyi Ren, Houbing Song, and Qinghe Du. 2016. Security enhancement for IoT communications exposed to eavesdroppers with uncertain locations. *IEEE Access* 4 (2016), 2840–2853.
[119] Zheng Yan, Peng Zhang, and Athanasios V. Vasilakos. 2014. A survey on trust management for Internet of Things. *Journal of Network and Computer Applications* 42 (2014), 120–134.
[120] F. Ye and Y. Qian. 2017. A security architecture for networked Internet of Things devices. In *GLOBECOM 2017-2017 IEEE Global Communications Conference*. 1–6. DOI: https://doi.org/10.1109/GLOCOM.2017.8254021
[121] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. 2014. Internet of Things for smart cities. *IEEE Internet of Things Journal* 1, 1 (2014), 22–32.
[122] Ting Zhao, Gang Zhang, and Lei Zhang. 2014. An overview of mobile devices security issues and countermeasures. In *International Conference on Wireless Communication and Sensor Network (WCSN'14)*. IEEE, 439–443.

Received May 2019; revised December 2019; accepted December 2019