Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 3029638, 18 pages
https://doi.org/10.1155/2018/3029638

Research Article

Shielding IoT against Cyber-Attacks: An Event-Based Approach Using SIEM

Daniel Díaz López,[1] María Blanco Uribe,[1] Claudia Santiago Cely,[1] Andrés Vega Torres,[1] Nicolás Moreno Guataquira,[1] Stefany Morón Castro,[1] Pantaleone Nespoli,[2] and Félix Gómez Mármol[2]

[1]*Computer Science Faculty, Colombian School of Engineering Julio Garavito, Colombia*
[2]*Department of Information and Communications Engineering, University of Murcia, Spain*

Correspondence should be addressed to Daniel Díaz López; daniel.diaz@escuelaing.edu.co

Received 28 May 2018; Revised 10 September 2018; Accepted 4 October 2018; Published 25 October 2018

Guest Editor: Constantinos Kolias

Due to the growth of IoT (Internet of Things) devices in different industries and markets in recent years and considering the currently insufficient protection for these devices, a security solution safeguarding IoT architectures are highly desirable. An interesting perspective for the development of security solutions is the use of an event management approach, knowing that an event may become an incident when an information asset is affected under certain circumstances. The paper at hand proposes a security solution based on the management of security events within IoT scenarios in order to accurately identify suspicious activities. To this end, different vulnerabilities found in IoT devices are described, as well as unique features that make these devices an appealing target for attacks. Finally, three IoT attack scenarios are presented, describing exploited vulnerabilities, security events generated by the attack, and accurate responses that could be launched to help decreasing the impact of the attack on IoT devices. Our analysis demonstrates that the proposed approach is suitable for protecting the IoT ecosystem, giving an adequate protection level to the IoT devices.

1. Introduction

The diversity of IoT devices has grown rapidly and attracted the attention of both the industry and academic society [1]. The Internet of Things (IoT) is considered as an emerging technology with considerable potential of development during this decade [2]. IoT is considered as a part of the Internet of the future, where billions of intelligent "things" will communicate to provide services to humans as an ultimate goal [3].

In [4], authors define the IoT as "*a world where physical objects are seamlessly integrated into the information network, and where the physical objects can become active participants in business processes. Services are available to interact with these 'smart object' over the Internet, query their state and any information associated with them, taking into account security and privacy issues*". Besides, IoT solutions are composed of a complex network of intelligent devices, sensors, and Internet connectivity, through which data can be collected, exchanged, and processed. These devices may be located in vehicles, buildings [5], home appliances [6], or cell phones, and they are controlled by software that allows them to be managed [7]. Under the IoT paradigm, some crucial application domains will be enhanced, such as healthcare and environmental and industrial plant monitoring [3]. To better perceive the IoT potential benefit, in 2016, Gartner estimated 6 billion things connected, forecasting additionally 21 billion smart objects in 2020 [2]. Moreover, Cisco Systems predicted that the IoT would create $14.4 trillion as a result of the combination of increased revenues and lower costs for companies from 2013 to 2022 [8].

IoT devices are equipped with sensors that capture data which must be transmitted to other places for processing and the IEEE and the IETF have defined a communication protocols stack for IoT designed to guarantee interoperability

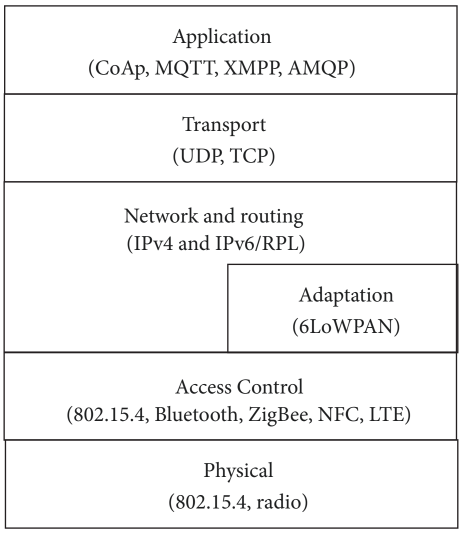

Figure 1: Communication protocols stack for IoT.

with other existing Internet devices [9]. The stack, illustrated in Figure 1, is composed of different interoperable protocols: (i) 802.15.4 (Low-Rate Wireless Personal Area Networks, LR-WPANs(low data rate solution with multimonth to multiyear battery life and very low complexity. It is operating in an unlicensed, international frequency band. Potential applications are sensors, interactive toys, smart badges, remote controls, and home automation)) and radio for the physical layer, (ii) Bluetooth, WiFi, NFC, 802.15.4, LTE, and Zigbee for the access control layer (MAC), (iii) IPv4 and 6LoWPAN (IPv6 Over Low-Power Wireless Personal Area Networks) [10] as a connector to the network layer (Adaptation) in 802.15.4 networks, (iv) RPL (IPv6 Routing Protocol for Low-Power and Lossy Networks) as a routing protocol, (v) UDP or TCP in transport layer, and (vi) CoAP (Constrained Application Protocol), MQTT (Message Queue Telemetry Transport), XMPP (Extensible Messaging and Presence Protocol), or AMQP (Advanced Message Queuing Protocol) for the application layer.

And to study the IoT solutions, an IoT architecture based on 5 components is proposed in [11]: (i) **perception or device** consisting of physical objects and sensor devices and their capacity to capture information, (ii) **network or transmission** responsible for transferring information from the device layer to the information processing system, (iii) **middleware** receiving the information from the network layer and storing it in a database, (iv) **application** consisting in the applications using the information gathered for the IoT devices, like smart health or home automation, and (v) **business** responsible for the management of the overall IoT system, including applications and services.

However, although IoT solutions are being implemented in a growing and fast way in almost all the aspects of humans' everyday life, their security exposition is not decreasing [12]. The integration of real-world objects with the Internet poses several security challenges, so that currently researchers and developers worldwide are struggling to find innovative solutions to address them [13, 14]. Specifically, the nature of the IoT ecosystem itself presents limitations from a security viewpoint. Due to the different standards and communication stacks involved, the resources constraints, and the massive amount and heterogeneity of interconnected devices, traditional security measures may not be employed efficiently in IoT systems [15]. Consequently, the process of reviewing different security alternatives for IoT devices becomes of crucial importance, so that one can spot the main challenges inherent to the IoT trend and possibly propose solutions [16].

Additionally, the large amount of data exchanged among the smart objects makes it unfeasible for a human to appropriately deal with all the events generated. In this direction, some proposals have been presented based on the management of security events generated by IoT devices. Concretely, these proposals suggest the use of a Security Information and Event Management (SIEM) system to centrally acquire knowledge stemming from IoT sources and, by doing so, to contribute with the development of IoT security [17, 18]. Although these proposals show promising features, none of them considers the possible correlations between IoT layers, security events, and attack surfaces, focusing mainly on suggesting methodologies to manage IoT security events coming into the SIEM. Furthermore, the reaction phase to the security incidents is commonly neglected, despite its vital importance due to the strong integration of the smart things into the real world. Thus, the proposed correlation is supportive for the security administrators, who can benefit from our studies to build strong correlation rules and to plan more robust reaction strategies against malicious threats.

The main contributions of this paper can be summarized as follows:

(i) Proposal of a security architecture for IoT adopting event management to deal with security incidents.

(ii) Detailed review of 11 IoT security event categories (defined by OWASP (https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project)) and proposal of a mapping between events, vulnerabilities, and attack surfaces in an IoT ecosystem.

(iii) Application of the proposed architecture to three common IoT scenarios, showing its feasibility through the generation of alerts and correlation rules.

(iv) Extension of the proposed solution through a defensive and offensive approach defining different incident responses.

The remainder of the paper is organized as follows: Section 2 describes the security problems for IoT devices, including the particularities of security event management. Section 3 exposes different SIEM solutions from a commercial and academic perspective. Following, Section 4 presents the details about the proposal of the security architecture using security events as a basis to protect IoT ecosystems. In Section 5 different alerts and correlation rules are implemented for the processing of security events of IoT devices allowing detecting and predicting security incidents. Different responses to security incidents from a defensive security strategy are considered in Section 6. Last but not least, Section 7 presents relevant conclusions and future works.

2. Problem Statement

2.1. IoT Attack Surfaces. An *attack surface* comprises the enablers (i.e., communication channels and protocols) and the targets (i.e., processes and data) that are required to perform an attack [19]. The attack surface generally depends on the interconnection between components of a system and the authorization definitions; i.e., privileges represented as access control policies. In an IoT architecture composed of a number of building blocks, it is reasonable to think about different attack surfaces. An attack surface also clusters all the different points that an attacker could use to get into a system and to steal/leak data out [20]. In this way, behind an attack surface there is a set of vulnerabilities that are the specific functions and/or elements of an IoT component that need to be reviewed from a security perspective.

Once an attack surface has been identified, it is possible to enumerate the security vulnerabilities and high-risk areas requiring defense-in-depth protection. Such protection refers to a commonly used strategy to shield critical resources on enterprise networks where security controls are established at multiple levels of the IT Information Technology infrastructure. In this context, the nine most relevant attack surfaces identified by OWASP in their Internet of Things Project (https://www.owasp.org/index.php/Attack_Surface_Analysis_Cheat_Sheet) for IoT ecosystems are listed as follows:

(1) Administrative interface: it consists of all the attack vectors (path or means by which an attacker can gain access to a computer or network server in order to deliver a malicious outcome [21]) related to the system's administrative web interface.

(2) Device web interface: it is composed of all the web application vulnerabilities of the IoT device web interface and credential management.

(3) Cloud web interface: the vulnerabilities involved in this attack surface are the standard web application vulnerabilities, credential management, transport encryption, and lack of two-factor authentication, applied to IoT cloud components like applications and web services.

(4) Mobile application: an attacker could use the vulnerabilities associated with a mobile application connected to an IoT device as attack vectors. Vulnerabilities can include username enumeration, weak passwords, lack of encryption and account lockout, among others.

(5) Device network services: insecure network services could allow Denial of Service (DoS) attacks, injections, Man in the Middle attacks, buffer overflow, and others.

(6) Update mechanism: lack of encryption or signature, writable locations, or missing update mechanisms could be used by an attacker to get into a system or introduce malicious scripts.

(7) Device physical interfaces: insecure elements involved in this attack surface could be used to tamper IoT devices, reset, get more privileges, extract device's firmware, or remove media storage. It also refers to the JTAG (Joint Test Action Group)/(Serial Wire Debug) interfaces that are used in chips for testing and debugging embedded applications. An unauthorized physical connection to these interfaces could introduce data injection, operation malfunction and also reverse engineering actions.

(8) Device firmware: firmware can expose sensitive data like credentials, encryption keys, authentication keys, firmware version, installed services, APIs, etc. This information could be used by the attacker to pivot toward another valuable service information. This attack surface includes vulnerabilities like backdoor accounts, firmware version display, vulnerable services, sensitive data exposure related to keys and accounts, among others.

(9) Local data storage: storage can be an attack surface when data is not encrypted, when it is encrypted using a compromised key, or when there are no integrity checks.

It is clear that the IoT ecosystem is exposed to security threats from different perspectives. The number of possible attack surfaces an attacker could leverage to perform their malicious activities represents unquestionably a motivation to build strong defense strategies. In addition, due to the resource-constraint nature of the IoT nodes, the traditional defensive schemes cannot be directly enforced, exposing the entire network to a higher risk. A clear example of such threats is the *Mirai botnet* that, together with its variants, was able to take control of the IoT devices and perform an incredibly powerful DDoS [22].

2.2. IoT Vulnerabilities. Information security in IoT devices, and in general in IT infrastructure, can be understood through three key concepts: confidentiality, integrity, and availability. In the context of IoT, availability would refer to the ability to use the information managed by the IoT device whenever it is required. Confidentiality would refer to the access to the information generated, stored, and processed by the IoT device allowed only to authorize users or entities. Integrity would consist of avoiding IoT data to be modified by unauthorized persons or entities. The above-mentioned security principles are better known as *CIA attributes*, and they represent a measure of the security level of the underlying system. Unfortunately, these attributes can be violated by exploiting vulnerabilities existing in the IoT devices. Therefore, the vulnerabilities are flaws in an element of a system that increase the attack surface. Specifically, an attacker may exploit the HW/SW breaches of the IoT system aiming at performing their malicious activities. Assuming that the detection of such violations is reliable, a security event is subsequently generated by the security monitors within the system.

Following the OWASP IoT project, the 16 most harmful IoT vulnerabilities grouped by security aspects are the following (https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project):

(i) Authentication:

  (a) lack of controls to avoid username enumeration: an attacker could collect valid usernames through authentication mechanisms.
  (b) use of weak passwords: this refers to passwords used in some authentication process against a component of the IoT service, which would be easy to guess because of its low complexity.
  (c) lack of an account lockout control after multiple failed attempts: it would allow an attacker to attempt authenticating many times without being blocked.
  (d) lack of two-factor authentication for critical functions: it claims that the authentication processes involved in the IoT service do not consider multiauthentication to access critical functions.

(ii) Cryptography

  (a) unencrypted network services allowing eavesdropping: communication between components of the IoT services are not protected allowing possible tampering.
  (b) failing in the implementation of encryption mechanisms: encryption methods are poorly implemented, improperly configured, or not being properly implemented.

(iii) Updating mechanism

  (a) update sent without encryption: updates at different architecture levels are transmitted over the network without using TLS or encrypting the files involved in the process.
  (b) remote update is done without security controls: there is no authentication method or secure remote control to regulate the remote device update.
  (c) the storage location for updates files is writable: storage location for update files has write and read permissions to any user, allowing firmware modification or distribution.

(iv) Physical access

  (a) firmware and data could be extracted allowing access to sensitive information: firmware could be extracted or gives more information than it should.
  (b) possible access to the device console because of lack of controls: it is possible to obtain full console access through serial interfaces, or lack of controls to avoid entering single user mode.
  (c) storage media is physically unprotected and could be removed.
  (d) lack of controls to avoid physical connection to the device causing manipulation of the code execution flow: it is possible to modify the execution code in order to bypass security controls or access to sensitive information.

(v) Device control

  (a) lack of controls against Denial of Service: it is possible to deny service through network or the device itself.
  (b) lack of controls to avoid command injection: there are no countermeasures to avoid injections, e.g., command or SQL, which can affect the data managed by the IoT service.
  (c) IoT service contains insecure third party components: applications or components developed by third-parties are out of date or they have security weaknesses.

One could easily argue that the task of preventing and reacting against the previously listed vulnerabilities is not trivial. In the IoT context, there is a particular need to support the system administrators, who bear the burden of controlling hundreds different events stemming from different sources.

In conclusion, the security analysis of the IoT ecosystem is complex, since it is exposed to different types of attacks, employing different attack surfaces and exploiting a variety of vulnerabilities present in IoT infrastructures. Nonetheless, the IoT environments generate an enormous amount of different events that should be leveraged to increase the security level of the entire system.

3. State of the Art

As previously stated, IoT ecosystems attracted the attention of both the academia and the information industry which are struggling to address a number of security challenges. On the one hand, IoT devices generate, process, and exchange vast amounts of security and safety-critical events, as well as privacy-sensitive information, making them appealing targets for ill-motivated entities [23]. And on the other hand, these devices can be involved as part of an attack, where vulnerable IoT devices are infected and become part of a botnet in order to subsequently reach further devices and compromise them [24]. Moreover, the attack surfaces that can affect traditional computer systems are extended in the case of IoT environments, adding some peculiarities due to their inherent characteristics [25]. For the ease of understanding, in this section we firstly introduce the concept of Security Information and Event Management (SIEM), and then we thoroughly review the related works of the field, highlighting their pros and cons.

3.1. Security Information and Event Management (SIEM). In order to understand what security event management means, it is necessary to define the concept of an information security event. Based on ISO 27000 [26], an *information security event* identifies the occurrence of a system, service or network state indicating a possible breach of information security policy or failure of controls or a previously unknown situation that may be security relevant. The security events can come from virtual private networks, firewalls, intrusion detection systems, intrusion prevention systems, routers, hosts, switches, and servers, among others [27]. Then, a security event management (SEM) is a system that allows real-time monitoring, reporting, normalization, correlation, and aggregation of security events. SEM combines with Security Information Management (SIM) allowing log management and reporting to conform a Security Information and Event Management (SIEM) [28]. Thus, SIEM supports threat detection and security incident response through the real-time collection and historical correlation and analysis of security events from a wide variety of events and contextual data sources across an IT infrastructure. It also supports compliance reporting and incident investigation through analysis of historical data from these sources (https://www.gartner.com/it-glossary/security-information-and-event-management-siem). A SIEM provides long-term analysis of security events and real-time reporting.

3.2. Related Works. Until now, some authors tried to tackle the IoT security weak points by investigating in hardware solutions [29, 30], while others suggested to solve the problem pinpointing to higher IoT layers, proposing innovative authentication methods included in MQTT, NFC [31], and smart card [32]. Above all, traditional defense mechanisms such as intrusion detection systems (IDSs) have been analyzed to further protect IoT devices. However, applying traditional IDS techniques is difficult due to the particular characteristics of IoT systems, such as constrained-resource devices, specific protocol stacks, and communication standards [33] as well as the network traffic generated [34]. Despite the motivations, the mentioned proposals do not consider the huge number of connected IoT devices in a real scenario with exception of works such as [34], where authors propose a mechanism to filter large volumes of traffic before performing analysis using a Bayesian inference method through an IDS. Consequently, the analysis of the generated volume of events is undervalued, as well as their correlation. To this extent, one could argue that the presence of a system which can centrally collect and analyze IoT events is crucial in a full-fledged IoT system, guiding the security experts to undertake optimal decisions.

Following this reasoning, some proposals have been presented among the research community to handle the above-mentioned drawbacks. From a design and functional perspective, [35] recommends the use of IoT events to determine the correct operational conditions; that is, the endpoints must be enabled with a warning threshold that helps to determine the state of a service. Reference [35] also suggests that the endpoints should log these events in a persistent memory to ensure that administrators can perform forensic analysis at a later stage. As the events are related to operational phases (e.g., power, temperature), the information included can be leveraged to detect possible malicious activities.

In addition, authors in [17] propose a system to analyze IoT security events as a “self-similar” system within the normal operation of its objects. Starting from the assumption that classical SIEM systems are incompatible to manage the vast amount of events generated by the IoT devices, authors suggest the study of an IoT architecture as a native network device. This network is represented as a graph with nodes (i.e., devices) and links (i.e., interconnection between devices) and the security analysis consists of the measurement of graph anomalies using fractal geometry.

Furthermore, in [18] authors propose integrating SIEM systems within IoT ecosystems using data aggregation, data digital signing, and swarm routing algorithm. Applying their methodology, they claim to tackle some problems related to this integration, such as data volume reduction, data integrity, and data delivery warranty for the messages exchanged between the devices and the SIEM. In particular, the last feature is enforced using a swarm algorithm, so that malicious network nodes are revised.

Another attempt of integration of SIEM technology into the IoT world is proposed in [36]. The authors present an IDS framework for IoT empowered by 6LoWPAN devices, which in turn sends the reported security events to an open-source SIEM (i.e., prelude (https://www.prelude-siem.com/en)). The feasibility of the proposed architecture is then demonstrated by conducting preliminary tests, where the performance of the implemented IDS framework is evaluated via penetration testing. Similarly, authors in [37] proposed a wireless IDS architecture applicable to a variety of IoT scenarios. Extending the work presented in [38], the authors suggest its application in the context of eHealth intrusion detection. In this case, the security events generated by open-source IDSs (i.e., Kismet (https://www.kismetwireless.net/) and Snort (https://www.snort.org/)) are sent to another open-source SIEM (i.e., OSSIM (https://www.alienvault.com/products/ossim)).

An approximation of SIEM suitable for IoT is presented in AMSEC [39, 40]. AMSEC is a proposal extending a traditional SIEM by adding interesting features to make it applicable to IoT scenarios. Specifically, authors propose a framework to model attack scenarios, which if deployed in tandem with a SIEM system is capable of (i) generating Attack Trees and Service Dependency Graphs based on the topological vulnerability analysis, (ii) applying algorithms to provide a near real-time attack modeling, (iii) analyzing attack models to predict future attacker’s steps, (iv) calculating security metrics, and (v) selecting the optimal security solution through an interactive decision support process.

Attacks over IoT devices have become a reality, as described in [41], where a complete taxonomy of Mirai botnet is presented. Mirai attacked different Internet targets through the control of nearly half a million IoT devices. The vulnerabilities exploited in this case over the IoT devices were “Use of weak password” and “lack of controls to avoid username enumeration”. In the same direction, [42] presents a threat analysis for smart home scenarios, which focuses on flaws caused by interactions among devices. Different kinds of IoT threats (mainly eavesdropping, DoS, impersonation, and software exploitation) are resumed and analyzed. Additionally, an analysis of how the attacks can be performed is developed for each threat.

All the above-mentioned solutions share a common background; that is, they propose the progressive integration of the SIEM technologies into the IoT systems to manage the events stemming from the IoT devices. Nevertheless, none of them focuses on the identification of proper IoT events and/or IoT event categories. Moreover, the existent connections between the IoT security events, vulnerabilities and attack surfaces is neglected. One could safely argue that a deep study in this direction constitutes an appealing research goal, since system administrators can leverage these connections to create stronger correlation rules and eventually plan strategic counteractions against suspicious alerts. Our work focuses on these challenges by highlighting the existing multirelations between symptoms (i.e., IoT events), causes (i.e., related vulnerabilities being exploited), and attack vectors (i.e., attack surfaces). The proposed multirelations can be used to identify the potential security risks to which IoT ecosystems are exposed and to make the correlation processes more efficient within the SIEM.

4. IoT Security Events Categories Analysis

The IoT events can be depicted as a representation or a symptom of something happening within the system which under certain circumstances may represent a security incident. Security events represent a comprehensive set of data which can be used in cyber security processes. However, it can be tremendously difficult to understand and use those events if there is not a clear understanding about their causes. For example, the causes can be related to the exploitation of some vulnerabilities on some components of the ecosystem. Additionally, since IoT ecosystems usually have a wide attack surface due to all the involved components, it is important to identify which particular attack vector or attack surface is directly related to that symptom and cause. Understanding the mapping between symptoms (i.e., IoT events), causes (i.e., vulnerabilities), and attack vectors is fundamental to develop different security tasks such as prevention, detection, and reaction against an IoT attack.

In this section, we present a security solution which uses a SIEM to process security events and protect IoT ecosystems. Figure 2 illustrates how the proposed solution is integrated in an IoT ecosystem. These security events are specially processed in the SIEM based on previously defined multirelations between security event categories, attack surfaces, and vulnerabilities.

The proposed multirelations help to contextualize the event, as it allows determining the vulnerabilities that could have been exploited and the related attack surfaces inside the IoT ecosystem. Reviewing the events bearing in mind the context in which they are generated is fundamental to achieve a security incident management process that can find the root cause of incidents and define proper remediation.

A schematic view of the proposed mapping between these concepts can be seen in Table 1, which is the result of the analysis developed around 11 security event categories (as we will see next) and can be utilized as a basis to understand the risks that IoT ecosystems face. Thus, an event belonging to an event category is generated upon the exploitation of one or more vulnerabilities present in one or more attack surfaces of an IoT ecosystem. Likewise, attack surfaces of an IoT ecosystem comprise one or more vulnerabilities, which after exploitation will generate events belonging to one or more event categories. In turn, when an attacker launches an ill-intentioned activity over one of the IoT attack surfaces, exploiting an IoT vulnerability, at least one security event belonging to an event category is generated.

It has to be stated that the analyzed security event categories are based on the event categories proposed by OWASP. Furthermore, the relation with the attack strategy, representing the techniques used by the attacker, was performed leveraging the taxonomy of Common Attack Pattern Enumeration and Classification (CAPEC) (https://capec.mitre.org/data/definitions/1000.html.), from the US-CERT at the US Department of Homeland Security. Next we analyze in depth each of these 11 security event categories.

4.1. Request Exceptions. This category clusters events related to abnormal requests addressed to the device, e.g., invocation of unsupported HTTP methods or reception of parameters in an unexpected way (quantity, type). The existence of these request events over a vulnerable IoT device could indicate that an attack is in progress. A number of vulnerabilities that, when exploited, would generate events of request exceptions are as follows:

(1) Lack of controls to avoid username enumeration: a username enumeration could be executed when there are not enough mechanisms that monitor and restrict the insertion of custom fields or the reception of maliciously prepared authentication requests, e.g., an attacker sends abnormal requests to an IoT device, using a fuzzer, to collect valid usernames (CAPEC-261). This vulnerability can be found in different parts of the IoT architecture, mainly in the following attack surfaces: mobile application and device web interface.

(2) Lack of two-factor authentication for critical functions: the bypass of a primary authentication method could be achieved by delivering abnormal requests, e.g., an attacker sends requests to a web server without a token/cookie, but containing an unexpected quantity of characters, avoiding the web interface validation (CAPEC-31). The IoT attack surface device web interface can expose this concrete vulnerability.

(3) Lack of controls against Denial of Service: a DoS attack may be generated sending abnormal requests against a target which could produce disruption in operations, e.g., an attacker floods the network to impede communications for the IoT device (CAPEC-482). This vulnerability might be contained in the IoT attack surface device network service.

(4) IoT service contains insecure third party components: an attack could exploit a third party component

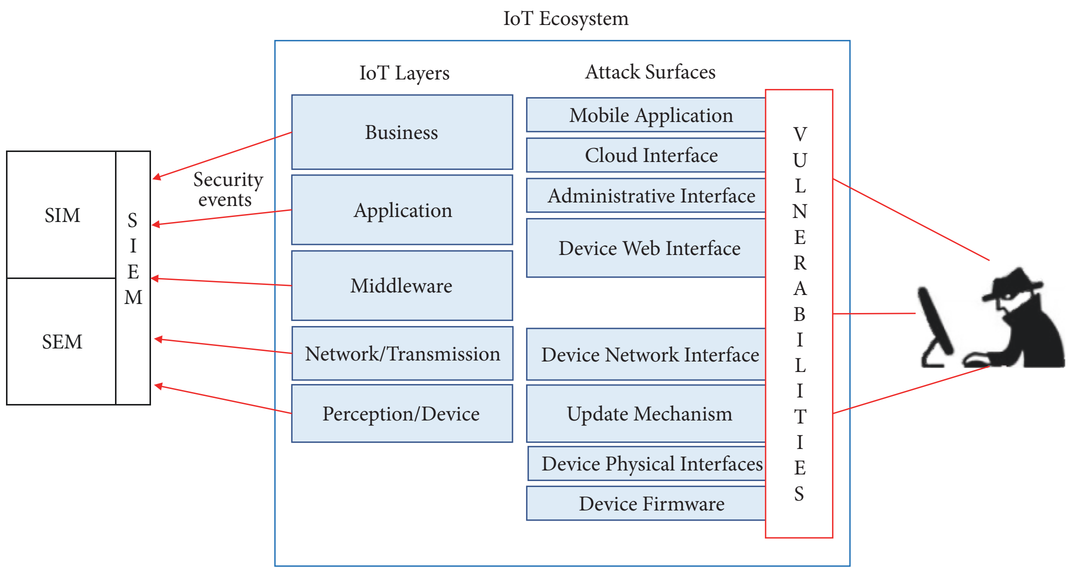

Figure 2: Proposal of integration of SIEM in an IoT ecosystem.

Table 1: Relations between event categories, vulnerability categories and attack surfaces on IoT ecosystems.

IoT Vulnerabilities	**Event Categories**											**IoT Attack Surfaces**								
	E_0	E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8	E_9	E_{10}	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
Lack of controls to avoid username enumeration	✘	✘		✘					✘			✘	✘	✘	✘			✘		
Lack of two-factor auth for critical functions	✘	✘		✘					✘			✘	✘	✘	✘			✘		
Lack of control against DoS attacks	✘	✘	✘	✘			✘		✘		✘	✘	✘	✘	✘			✘		✘
IoT service contains Insecure 3r party components	✘								✘					✘						
Use of weak password		✘		✘					✘			✘	✘	✘	✘			✘		
Lack of an account lockout after multiple failed attempts		✘		✘		✘			✘			✘	✘	✘	✘			✘		
Unencrypted network services allowing eavesdropping					✘				✘					✘				✘		
Lack of controls against manipulation of the code execution flow					✘	✘			✘											✘
Storage location for updates files is writable					✘				✘	✘				✘		✘			✘	
Lack of control for device console access				✘			✘	✘	✘		✘									✘
Update sent without encryption						✘			✘										✘	
Storage Media is physically unprotected							✘		✘		✘									✘
Possible Firmware and data extraction							✘		✘								✘			✘
Fail in the implementation of encryption mechanisms									✘	✘								✘		
Remote update is done without security controls									✘	✘									✘	
Lack of controls to avoid command injection									✘	✘				✘	✘					

Acronym and event categories: E_0, request exceptions; E_1, authentication exceptions; E_2, input exceptions; E_3, access control exceptions; E_4, session exceptions; E_5, ecosystem member exceptions; E_6, Device Access Events; E_7, admin mode events; E_8, honey trap exceptions; E_9, command injection exceptions; E_{10}, reputation exceptions. **acronym and IoT attack surfaces**: S_0, mobile application; S_1, cloud web interface; S_2, device web interface; S_3, admin interface; S_4, local data storage; S_5, Device firmware; S_6, device network services; S_7, update mechanism; S_8, device physical interfaces.

which has some functions for an IoT service, e.g., an attacker invokes abnormal HTTP methods to a third party component generating request events (CAPEC-86, CAPEC-460). The IoT attack surface device web interface may expose this specific vulnerability.

4.2. Authentication Exceptions. Those events related to the authentication process are grouped within this category and could include, for example, excessive login attempts or untrusted user location. Next we present some vulnerabilities that could exist in IoT ecosystems which when exploited would generate events of authentication exceptions type.

(1) Lack of controls to avoid username enumeration: a username enumeration could result successful through the delivery of probes to the authentication functions; e.g., an attacker starts a dictionary attack trying different common usernames (CAPEC-16). This vulnerability could be exposed across different parts of the IoT architecture, mainly in the following

attack surfaces: administrative interface, device web interface, coud web interface, and mobile application.

(2) Use of weak password: an attack against an IoT infrastructure can be performed by exploiting a weak password; e.g., an attacker starts a dictionary attack trying different common passwords. This vulnerability might be found throughout several components of the IoT architecture, specifically in the following attack surfaces: administrative interface, device web interface, cloud web interface, and mobile application.

(3) Lack of two-factor authentication for critical functions: a critical function in an IoT ecosystem should be protected by multifactor authentication so its access can not be exploited easily; e.g., an attacker introduces a valid username and password and tries to bypass a biometric control with a fake fingerprint (CAPEC-180). This vulnerability is contained in the following IoT attack surfaces: administrative interface, cloud web interface, and mobile application.

(4) Lack of an account lockout control after multiple failed attempts: an account lockout should be triggered upon a brute force attack; e.g., an attacker starts a brute force attack against an account that is not protected against multiple failed attempts, allowing an illegitimate access (CAPEC-49). We can find this vulnerability mainly in the next IoT attack surfaces: administrative interface, device web interface, cloud web interface, and mobile application.

(5) Lack of controls against Denial of Service: DoS attacks may succeed by exploiting the authentication functionalities of a IoT system; e.g., an attacker performs a DDoS attack through failed SSH connection attempts drowning the server or the IoT device (CAPEC-489). The IoT attack surface device network services exposes this particular vulnerability.

4.3. Session Exceptions. This category refers to those events stating that something abnormal in the creation, establishment or revocation of a session is happening. It could include, for example, a cookie being modified, receiving a request containing a valid session ID of a user different from the current one or unexpected changes of location of the user during the same session. In the following we introduce a few vulnerabilities from IoT ecosystems whose exploitation could generate events of session exceptions type.

(1) Unencrypted network services allowing eavesdropping: a session running under an unencrypted channel could affect seriously the operation of an IoT service, e.g., an attacker captures the unencrypted traffic transmitted in a session creation process and then they are able to sniff (CAPEC-158) and even modify packets (CAPEC-31) to send requests using a captured IoT device authentication token. We can observe this vulnerability in the IoT attack surfaces: device web interface and device network services.

(2) The storage location for updates files is writable: if a user is able to write in the file system of some of the components of the IoT ecosystem, she could affect the integrity of the service; e.g., the uploads folder has all permissions, or the uploads web interface is exposed to the Internet without authentication, so anyone could upload malicious files to the IoT device or cloud server (CAPEC-75). This vulnerability might be exposed by the next IoT attack surfaces: local data storage and device web interface.

(3) Lack of controls to avoid physical connection to the device causing manipulation of the code execution flow: the access to the physical components of an IoT ecosystem must also be protected to avoid affectations to confidentiality, integrity, or availability of the services; e.g., an attacker connects an external device to the JTAG interface of the IoT device and then starts an adulterated process to establish a session, which after being successful manipulates the code execution flow (CAPEC-391). The IoT attack surface device physical interfaces usually contain this specific vulnerability.

4.4. Access Control Exceptions. This category logs those events indicating that some access control functions are not working properly, specifically authorization functions. For example, a URL argument or POST parameter that has been modified in order to access a confidential object through an Insecure Direct Object Reference. The exploitation of the following vulnerabilities, present in IoT ecosystems, might generate events of access control exceptions type:

(1) Possible access to the device console because a lack of controls: console access should be restricted since an improper access to it could lead to the execution of an attack over the IoT service; e.g., an attacker gets legitimate accesses to an IoT device console but forces a switch out of users to get elevated privileges (CAPEC-233). This vulnerability is mainly found in the IoT attack surface device physical interfaces.

(2) Any vulnerability included in the authentication exception category could also generate an access control exception: a proper working of the authentication functions in IoT services is fundamental to support the execution of specific instructions; e.g., an attacker exploits an authentication vulnerability and then they get access to the privileges of the hacked account and they start invoking commands. The combination of authentication exception events and the later execution of commands could generate access control exception events. These vulnerabilities might be exposed by the following IoT attack surfaces: administrative interface, device web interface, cloud interface, mobile application, and device network services.

4.5. Ecosystem Membership Exceptions. This category encloses events related to the registration of the different components of the IoT ecosystems. For example, it puts on evidence traffic coming from an unregistered/revoked device and it also could include exceptions related to the registration process. Next we describe a number of vulnerabilities commonly found in IoT ecosystems, whose exploitation might imply the generation of events of ecosystem membership exceptions type.

(1) Lack of account lockout control after multiple failed attempts: a fake IoT device may try a registration process repeatedly using different combinations of parameters to cheat the IoT ecosystem; e.g., attacker starts a brute force attack. This vulnerability in particular may come along with the following IoT attack surfaces: administrative interface, device web interface, cloud web interface, and mobile application.

(2) Update sent without encryption: the transmission of an update through an unencrypted channel may yield confidentiality affectations; e.g., an attacker captures the unencrypted traffic related to the update and uses it to discover the mechanism that is being used to register IoT devices, so, it can be replicated to register fake IoT devices (CAPEC-609, CAPEC-615). The IoT attack surface update mechanism usually exposes this specific vulnerability.

(3) Lack of controls to avoid physical connection causing manipulation of the code execution flow: unauthorized physical connections can harm the operations of IoT devices and consequently their services; e.g., an attacker connects an external device to the JTAG interface of the IoT device and then forces a reregistration or a revocation (CAPEC-390). The IoT attack surface device physical interfaces commonly contains this concrete vulnerability.

4.6. Device Access Events. In this category we can cluster the events generated whenever an access to the physical device occurs, such as the removal of some protection element belonging to the device or the manipulation of the hardware. A number of vulnerabilities specific to IoT ecosystems and whose exploitation could yield events of device access type are as follows:

(1) Storage media is physically unprotected and could be removed: storage protection should be implemented to avoid extraction or modification of sensitive data; e.g., an attacker accesses an IoT device and extracts the media containing the operative system and potentially confidential data (CAPEC-547). The IoT attack surface device physical interfaces usually contains this vulnerability.

(2) Firmware and data could be extracted allowing access to sensitive information: repository for firmware and data should also be physically protected to avoid data exposition; e.g., an attacker accesses to an IoT device and extracts the media containing firmware and potentially confidential data (CAPEC-547). This vulnerability might be found in the next IoT attack surfaces: device physical interfaces and device firmware.

(3) Lack of controls against Denial of Service: a successful DoS attack might be achieved by the physical access to some components for which an IoT component depends on; e.g., an attacker disconnects the device power supply and the administrative interfaces notice that the IoT device is offline (CAPEC-547). This specific vulnerability might be exposed across the following IoT attack surfaces: device physical interfaces and device network services.

(4) Possible access to the device console because of a lack of controls: the absence of physical barriers could entail an unauthorized access to an IoT device; e.g., an attacker accesses to an IoT device console and logs in as a user with some preset privileges (CAPEC-40). The IoT attack surface device physical interfaces often manifests this vulnerability in particular.

4.7. Administrative Mode Events. This category of security events deals with the use of administration privileges to perform some action over the device. Events can be outputted due to the raise of privileges over the console (even if this is done in an authorized or unauthorized way) and of each administrative action that is invoked thereafter. The principal vulnerability in the context of IoT ecosystems whose exploitation might end up in the generation of events of administrative mode type is as follows:

(1) Possible access to the device console access because of a lack of controls: console access should be restricted and monitored to avoid the use of privileged functions; e.g., an attacker accesses to an IoT device console and logins as a user with some preset privileges, intending an elevation of privileges (CAPEC-40, CAPEC-249). The IoT attack surface device physical interfaces usually embraces this concrete vulnerability.

4.8. Input Exceptions. This category includes events that are generated upon unexpected input data coming to applications or devices. This includes, for example, coding or format errors. The exploitation of the next key vulnerability, commonly found in IoT ecosystems, might yield events of input exceptions type.

(1) Lack of controls against Denial of Service: DoS attacks can be originated introducing malformed inputs in some user fields exposed in the IoT components; e.g., an attacker sends device abnormal input data in requests addressed to an IoT device, with the intention of generating a failure in the device network services and achieve a successful DoS (CAPEC-469). The IoT attack surface device network services typically contains this vulnerability.

4.9. Command Injection Exceptions. This category of events embraces those attempts of injections that contain execution commands. In the case of IoT devices, a code injection could be harmful as it could extract, modify, or delete information kept within an IoT device. A number of potential vulnerabilities that could be exploited in order to generate events of command injection exceptions type are as follows:

(1) Failure in the implementation of encryption mechanisms: commands can be injected when a channel does not use encryption or the encryption implementation is weak; e.g., an attacker takes benefit of a failure in the implementation of the encryption algorithm, so they can decrypt, modify (inject commands), and encrypt again (CAPEC-20, CAPEC-463). This vulnerability could be found in the IoT attack surface device network services.

(2) Remote update is done without security controls: the updating process should be conducted under a ciphered channel to avoid affectations to the integrity; e.g., an attacker intercepts the unencrypted traffic related to an update and modifies it to inject a specific command that will be executed during a critical task (CAPEC-187, CAPEC-533). The IoT attack surface update mechanism commonly exposes this specific vulnerability.

(3) The storage location for updates files is writable: data stored in file systems should be protected to avoid modification and injection of commands; e.g., an attacker discovers a way to overwrite a data file in an IoT device, so they modify it to include a specific command (CAPEC-75). This file is executed later by the IoT device and the commands could get operative. The IoT device could detect the syntax of the command since such element is not expected in a data file. This vulnerability is typically exposed in the IoT attack surface update mechanism.

(4) Lack of controls to avoid command injection: appropriated controls to avoid command injection in all levels should be implemented to avoid a high impact risk; e.g., an attacker notices that the configuration web interface of an IoT device has insecure inputs and they inject command parameters to be executed on the IoT device, corrupting device data, and disabling the accountability functionality or information disclosure (CAPEC-500). This specific vulnerability is commonly found in the following IoT attack surfaces: device web interface and administrative interface.

4.10. Reputation Exceptions. This category contains those events related to situations where trust and reputation are involved. For instance, a user access from a new and remote location could be considered as dangerous and this could affect the trust that the IoT service would give to such access. Next we introduce some vulnerabilities in the context of IoT ecosystems whose exploitation could generate events of reputation exceptions type.

(1) Lack of controls against Denial of Service: reputation could be used to define the normal access to IoT services and avoid attacks such as DoS; e.g., an attacker programs a botnet to send multiple ill-intentioned requests to an IoT device (CAPEC-469). Such device detects requests coming from unusual sources and therefore it assigns a low trust to these connections. The IoT attack surface device network services commonly exposes this specific vulnerability.

(2) Storage media is physically unprotected and could be removed: IoT device integrity can be considered to avoid unauthorized hardware modifications and redefine reputation indicators; e.g., an attacker changes the device storage media to collect information (CAPEC-547). This vulnerability is frequently present in the IoT attack surface device physical interfaces.

(3) Possible access to the device console access because of a lack of controls: abnormal executions can also influence the reputation indicators of the IoT ecosystem; e.g., when a user gets access to the console under suspicious circumstances, the IoT ecosystem would trust less in such device until a confirmation of safety. The IoT attack surface device physical interfaces commonly includes this vulnerability in particular.

4.11. Honey Trap Exceptions. This category includes those events outputted by traps for the attackers. A trap could, for example, expose tempting but fake data, so that the attacker gets motivated to start a penetration activity toward the IoT ecosystem. This includes, for example, events generated from the access to a trap resource. The key vulnerability that could exist in IoT ecosystems whose exploitation would generate events of honey trap exceptions type is described next.

(1) The exploitation of all the IoT vulnerabilities that have been mentioned previously could entail the generation of honey trap exceptions. However, considering the nature of honey traps, the most commonly exploited vulnerabilities could be (i) lack of controls to avoid username enumeration, (ii) lack of two-factor authentication for critical functions, (iii) the storage location for updates files being writable, and (iv) remote update being done without security controls.

    Honey traps exposing obvious (but bogus) vulnerabilities can be useful to make intelligence over the adversaries; e.g., an attacker probes a system structure related to an IoT service to evaluate its security level, mapping the application, and exploiting an apparent update location writable vulnerability coping files to common resource locations and executing commands (CAPEC-40, CAPEC-150).

    Likewise, each of these vulnerabilities can be related to multiple IoT attack surfaces.

In this section we presented a security solution (Figure 2) employing a SIEM to receive, process, and manage security events coming from the different components of an IoT ecosystem. The key element to consider behind the process done within the SIEM is the proposed multirelation between security event categories, attack surfaces, and vulnerabilities, represented in Table 1. The multirelations were built as a result of the analysis done in this section around 11 security event categories. This proposed multirelations will be used in the next section to support building correlation rules in a SIEM, allowing getting a better knowledge about the IoT security risks.

It has to be stated that the process of integrating a SIEM solution within the IoT ecosystem is not trivial. As a matter of fact, some challenges have to be addressed in order to accommodate this procedure. Among them, one of the most relevant is represented by the heterogeneity exposed by the IoT devices. That is, the intelligent *things* can be any kind of device, using diverse protocols and data format to exchange information [43]. Consequently, the system administrator is in charge of understanding and manually managing the events stemming from the different sources in order to collect and organize them. To this extent, great effort is required in order to standardize the communication among the IoT nodes. Moreover, the scalability of the SIEM can be undermined as the amount of resources needed to handle the IoT data and then run analytics increases exponentially. To tackle this problem, it is indeed possible to exploit the benefits of Big Data technologies. By leveraging its powerful capabilities, it is possible to minimize the amount of data which the SIEM solution has to store and, therefore, analyze [44].

5. Alerts and Correlation Rules

In this section, alerts and correlation rules are built according to the security events emanating from the different components of an IoT ecosystem. In our idea, this task must help the security administrator of the IoT infrastructure to better understand the existing correlations between the events generated within the IoT system. Correlation rules support a prevention, detection, and reaction security strategy, as they identify anomalous or suspicious situations in an IoT ecosystem evidencing that something potentially harmful happened, is happening now, or will happen soon.

In turn, the IoT security events depend on the nature of the IoT device and are related to the IoT architecture components [11] (see Figure 2). Regarding the application and business components, there are security events related to the code and the operations performed by the IoT applications including local, web, cloud, and mobile operations. Examples can include events like input or output of data, execution of functions, and abnormally reported or measured values. In the middleware component, events are related to the database access, component/user security, and privacy and interoperation. Examples can include user or things authentication, database privileges, and inconsistencies between applications and platforms reported. In the network and transmission components we can find security events related to the communication between devices and through a server located in the Internet using specific network protocols. Examples of events are validation of data in the destination, transmission errors, and communication method. In the perception or device component we find events related to the electronic components that conform the IoT device. Examples of events could include the validation of integrity of the device using a *Trusted Platform Module* (TPM), which leverages the concept of root of trust [45]. Specifically in this Section, 3 security events will be further detailed.

Regarding the generation of correlation rules, favorably a number of different algorithms have been presented so far, which can be categorized according to their characteristics [46]: (i) similarity-based, where events are compared based on their similarity and grouped over time, (ii) knowledge-based, where events are compared against a database of attack patterns, and (iii) statistical-based, where statistical attributes in attacks are analyzed, specially their frequency of occurrence with respect to past statistical data.

Correlation rules are usually created in a manual way applying some of the aforementioned algorithms, and in practice different threat intelligence providers offer feed services to guarantee a regular update of rules. In order to help the security administrators to generate correlation rules, this paper proposes a mapping between events, vulnerabilities, and attack surfaces for IoT ecosystem (see Section 4). The approach proposed here is a knowledge-based algorithm, since the mapping is actually a taxonomy of events, vulnerabilities, and attack surfaces, constituting the baseline for an attack pattern. The generation of the correlation rules in the following subsections was straightforward for the security administrator thanks to the existence of such proposed mapping.

The rule generation process firstly consists of conducting a threat analysis for a specific IoT ecosystem. This analysis is made by reading the mapping as shown in Table 1, i.e., starting from an security event stemming from an IoT ecosystem, belonging to an event category, and indicating that the exploitation of an IoT vulnerability has occurred over a specific attack surface. So, the security administrator must generate at least one correlation rule for such situation capturing as well the identified security events under the affected attack surface, and additionally preparing an immediate remediation over the identified IoT vulnerabilities.

The threat analysis can also be made reading Table 1 in the following way: the IoT ecosystem exposes some attack surfaces comprising one or more vulnerabilities, whose exploitation will generate events belonging to one or more event categories. So, the security administrator must generate at least one correlation rule for such a situation that has an attack surface as a starting point and then define all the possible security events, with their corresponding event categories that can come from there. Additionally, the security administrator must also define the remediation to be enforced in order to patch the related IoT vulnerability.

In any case, the algorithms to generate correlation rules require the existence of a knowledge base with suspicious activities. Such dubious activities can be recorded and marked individually as alerts, and the correlation of alerts, done through a correlation rule in the SIEM, can proclaim an attack.

When the SIEM is running in a state where all incoming events are analyzed against the set of active correlation rules, and if one of the events matches some previously defined conditions for a rule, the SIEM will register it until a new related event appears. This situation is done until a threshold is reached confirming an attack scenario, so the system administrator is informed and an incident response as part of a defensive safety strategy is triggered. These algorithms can be very efficient for their accuracy, ability to avoid false alerts, and capacity to detect multistage attacks. However, these correlation rules can be difficult to define, since their

```
initialize bufferOfEvents to zero
initialize threshold to maximumTolerable
while bufferOfEvents is not empty
  Get newEvent from bufferOfEvents
  Analize newEvent against correlationRules
  If newEvent matches a correlationRule
    matchedRuleCounter = matchedRuleCounter + 1
    If matchedRuleCounter >= threshold
      print "Event confirms an attack in progress"
      Inform the system Admin
      Launch an Incident Response
    else
      print "Event matches a rule but is not even an attack"
  else
    print "Event did not match any rule"
```

Listing 1: Reception and processing of security events by a SIEM.

arguments must be set in a precise way and are also deficient against new attacks (see Listing 1).

Next, we will analyze three scenarios that are of special interest in IoT ecosystems, namely, geofencing, brute force, and command injection. In each scenario, we study some representative security events to define correlation rules implemented in a SIEM, which will aid the IoT architecture administrators in their security labors. The rules have been implemented on the Open-Source Security Information and Event Management System (OSSIM) which is one of the most popular open-source SIEMs, with a strong supporting community which in turn could be helpful to solve emerging issues.

5.1. Scenario 1: Geofence Attack in IoT Devices with Geolocation Functions. The first scenario is given by an IoT device moving within a delimited zone called *safe zone*. A device under this condition is called *geofenced*. Geofencing refers to a "virtual barrier or geographical border around a single point with a predefined set of boundaries on a geographical area mapped either with GPS or RFID". In this case, a safe zone has been defined for an IoT device and if the device exits the safe zone the SIEM will generate an alert.

As an example, a malicious user (attacker) takes possession of an IoT device (e.g., smart bike or vehicle) and intends to steal it. Within this scenario, the IoT device constantly validates its current position within the established geofence and when the device gets out of the safe zone, it sends a Device Access event to the SIEM informing about the theft. The Device Access event will be reported to the SIEM containing a correlation rule which validates the received information and generates an alert, so the theft can be prevented. In this scenario, the vulnerability exploited is Denial of Service, found in the attack surface device physical interface.

Hence, the correlation rule created for this scenario can be observed in Table 2, named, "IoT Geofencing Directive", comprising the following: (i) a "User data" defined in "GE1" which is the keyword used to notify the SIEM about a theft, (ii) a "Reliability" of 8 (in a scale up to 10, used internally to calculate the risk), (iii) a no defined "Timeout" (meaning that the frequency of the event in a time window is irrelevant), and (iv) an "Occurrence" of 1 (meaning that one occurrence of a Device Access event with a "User data" in "GE1" is enough to confirm the theft). This rule uses an OSSIM specially crafted plug-in that implements a regular expression to receive Device Access events from all the IoT devices being monitored.

5.2. Scenario 2: Brute Force Attack upon Failed Login in IoT Devices with Authentication Model N:N. The second scenario is represented by an attacker who attempts to perform a brute force attack against an application in an IoT device. The *N*:*N* authentication model is based on the interaction between an IoT device with more than one user and a user that could interact with a lot of components. The concurrent accesses or multiple accesses could affect the availability of an IoT device. Thus, a successful DoS may be achieved by the attacker trying multiple combinations of usernames and passwords through a brute force attack, until a valid username and password combination is found. It is important to avoid this type of attack because once the attacker enters the system as a valid user, she could exploit new vulnerabilities and would be considered as a trusted user within the system. The brute force attack should be detected through a correlation rule.

To this end, an authentication exception event is generated every time an invalid login is attempted. If any of these events constantly appears within a very short period of time, the correlation engine should generate an alert. In this scenario, the authentication exception events will be the evidence that any of the following vulnerabilities has been exploited: username enumeration, use of weak passwords, account lockout, or two-factor authentication. Additionally, these vulnerabilities can be applicable in the following attack surfaces here: administrative interface, device web interface, cloud interface, and mobile application.

Therefore, the correlation rule created for this scenario can be observed in Table 2 and is named "IoT rule", composed of (i) a "Reliability" of 1 (in a scale up to 10, used internally to calculate the risk), (ii) a no defined "Timeout"

Table 2: Correlation rules for different attacks scenarios implemented in OSSIM.

Scenario	Rule name	Reliability	Timeout [sec]	Occ	Security event	User data	Vulnerability exploited	Attack surface related
1	IoT GeoFencing Directive	8	None	1	Device Access event	GE1	Denial of Service	Device physical interface
2	IoT rule	1	None	1	Authentication Exception event	BF1	Username enumeration, Use of weak passwords, Account lockout or two-factor authentication	Administrative interface, Device web interface, Cloud interface and Mobile application
	Brute Attack	3	5	5				
3	IoT Command injection	3	None	1	Command Injection Exceptions event	AE1	Encryption mechanisms impl. fails, Remote update is done without security controls or Storage location is writable	Device network services and Update mechanism

(meaning that the frequency of the event in a time window is meaningless), and (iii) an "Occurrence" of 1 (meaning that one occurrence of an authentication exception event with a "User data" in "BF1" is enough to confirm the alarm). This rule uses an OSSIM specially crafted plug-in that implements a regular expression to receive authentication exception events from all the IoT devices being monitored.

Additionally, there is a second nested rule called "brute force attack" which is similar to the "IoT rule", where the reliability will increase from 1 to 3 when at least 5 events are received (occurrence) in a time window of 5 seconds (timeout). A bigger reliability means that the associated risk will be higher.

5.3. Scenario 3: Command Injection Attack in IoT Devices Deployed in Hostile Environments. In the third scenario, the attacker wants to access or modify data stored within an IoT device through a command injection attack, where a malicious code is interpreted and executed over the administrative or device web interface. The injected code would allow modifying, stealing, or even eliminating data. Even a DoS attack could be provoked by the injection leaving the resource inaccessible.

Here, a command injection exceptions event is generated as a consequence of the recognition of unusual characters, e.g., SQL sentences embedded in the data that the IoT device receives. In this scenario, this event will be an evidence that any of the following vulnerabilities have been exploited: fails in the implementation of the encryption mechanisms, remote update being done without security controls, or storage location for updates files being writable. Finally, the aforementioned vulnerabilities can be applicable in the following attack surfaces: device network services and update mechanism.

Thus, the correlation rule created for this scenario can be observed in Table 2 and is named "IoT command injection", including (i) a "User data" defined in "AE1" which is the keyword used to notify the SIEM about an injection on the IoT device, (ii) a "Reliability" of 3 (in a scale up to 10, used internally to calculate the risk), (iii) a no defined "Timeout" (meaning that the frequency of the event in a time window is irrelevant), and (iv) an "Occurrence" of 1 (meaning that one occurrence of a command injection exceptions event with a "User data" in "AE1" is enough to confirm the alarm). This rule uses an OSSIM specially crafted plug-in that implements a regular expression to receive command injection exceptions events from all the IoT devices being monitored.

6. Incident Responses and Defensive Safety

It is clear that managing large amounts of events is a burden in the work of system administrators and that using a SIEM facilitates their work, especially when correlation rules and countermeasures can not be easily elicited by a nonexpert professional. Hence, this section deals with the incident response and defensive security based on the threats identified in the previous section. A security incident is defined as "a violation or imminent threat of violation of informatics security policies, acceptable use policies, or standard security practices" [47]. The NIST's (National Institute of Standards and Technology) Incident Management Guide establishes four phases: (i) preparation; (ii) detection and analysis; (iii) containment, eradication, and recovery; and (iv) postincident activity. Assuming a correct detection and analysis phase, the security incident must be detected and consequently an optimal reaction against it must be triggered [48]. This reaction must balance the inherent trade-off between cost of the counteraction and its negative impact on the system [49].

There are five types of security controls which treat the risk associated with an attack [50]: (i) *preventive*, to avoid the incident, (ii) *detective*, to discover an incident in progress, (iii) *corrective*, to decrease the impact, (iv) *deterrent*, to discourage the incident, and (v) *compensatory*, to place controls instead of more desirable controls as an alternative safeguard. Additionally, security controls also have three kinds of implementation [50]: (i) *administrative*, based on procedures or policies, (ii) *technical*, based on hardware or software components, and (iii) *physical*, based on elements supporting physical security. To this extent, SIEM systems can help in the detection and analysis phase through correlation rules and alarms, but they can also support the containment, eradication, and recovery phase through the generation of responses as part of a defensive security strategy. Finally, SIEM systems can also contribute in the postincident activity phase based on its Digital Forensics capabilities [51].

Generally, the SIEM systems are equipped with a response-focused component that is able to execute actions triggered by a security incident (e.g., a confirmed intrusion is detected). These actions can be clustered in (i) active responses (technical controls intended to neutralize the attack) and (ii) passive responses (technical controls intended to alert the administrator about the detected incident) [52]. Therefore, a SIEM system could be used to support an incident response and defensive security strategy for an IoT ecosystem having the following considerations: IoT devices are generally implemented over computationally constrained hardware; there can exist transactions between IoT devices and also between the IoT device and an IoT platform remaining generally in the cloud, the SIEM system must be able to analyze trillions of IoT devices, get evidence from any kind of formats, scan different kinds of networks such as RFID or sensor networks, and handle exabytes of data, among others.

Additionally, a SIEM must secure all the gathered data as it could contain sensitive information about citizens, industrial processes, buildings, transportation means, etc. [53]. According to the types of technical security controls mentioned previously, some of them are defined for each one of the scenarios defined in the previous section. For each case, the affected IoT device or platform and the scenario are also described.

6.1. Scenario 1: Geofence Attack in IoT Devices with Geolocation Functions. As part of the defensive security strategy for this scenario, several technical security controls can be defined, as shown next.

(i) **Preventive**: create a correlation rule in the SIEM that identifies when the location of a device is suspiciously close to the boundary of the geofence, so an alarm can be triggered to avoid a possible theft.

(ii) **Detective**: implement a correlation rule in the SIEM that detects when the IoT device is out of the geofence and warn the security manager, security personal, or the owner to avoid the device theft.

(iii) **Corrective**: implement an action in the SIEM that turns off the device through a command, to avoid the use of the IoT device. Alternatively, implement an action that sends the current device location to its legitimate user and/or the system administrator to avoid the theft.

(iv) **Deterrent**: implement an action in the SIEM system that sends a command to the device to emit a sound that alerts the device theft and deters the thief from continuing with the robbery.

(v) **Compensating**: if the SIEM detects that the theft could not be avoided, execute an action to delete all the information, including a deep erase, so the configuration or operation information cannot be compromised.

6.2. Scenario 2: Brute Force Attack upon Failed Login in IoT Devices with Authentication Model N:N. In this scenario, we propose the following technical security controls following the taxonomy proposed in [50]:

(i) **Preventive**: create a correlation rule in the SIEM that identifies when a brute force attack is in progress, due to the unusual characters being received, so an alarm can be triggered to avoid the attack being successful.

(ii) **Detective**: implement a correlation rule in the SIEM that detects when a brute force attack in progress discovers a set of user credentials. Warn immediately the security manager to avoid that the user account can be exploited.

(iii) **Corrective**: implement an action in the SIEM that enables the resolution of a captcha or the application of a two-step authentication process for new user authentication attempts [54].

(iv) **Deterrent**: implement an action in the SIEM system that, through a command, takes a picture with the front face camera (to photograph the attacker) and shows it in the device's display with a message asking to stop the attack. This control could be valid in case that the attack was performed locally from the IoT device, not remotely.

(v) **Compensating**: if other controls do not work, the SIEM could execute a script to enable rules in a WAF (Web Application Firewall) available in the network that avoids the brute force attack.

Any of these responses could be accompanied by an email to the IoT device owner, notifying about the possible attack, the device information, and where the attack comes from.

6.3. Scenario 3: Command Injection Attack in IoT Devices Deployed in Hostile Environments. As part of the defensive security strategy, different technical security controls can be defined for this scenario, as shown next.

(i) **Preventive**: create a correlation rule in the SIEM that identifies when a command injection attack is in progress, detecting that unusual characters are being received in a user input field, so an alarm can be triggered to avoid the attack being successful.

(ii) **Detective**: implement a correlation rule in the SIEM that detects when a command injection attack in progress manages to execute a special command in the device of the victim. Warn immediately the security manager to avoid malicious operations.

(iii) **Corrective**: implement an action in the SIEM that increases the level for the security policies running in the operative system of the victim, so special commands cannot be run immediately.

(iv) **Deterrent**: when unusual characters are detected in the security events coming to the SIEM, it can start an action that sends a warning message to the attacker informing that the attack will be reported and its consequences.

(v) **Compensating**: if other controls do not work, the SIEM could execute a script to enable rules in a WAF (Web Application Firewall) available in the network that avoid the command injection attack.

6.4. Evaluation. The evaluation of our proposal was conducted through the emulation of an IoT-secure device built under the premise of security by designed. Our IoT-secure device, running Ubuntu IoT OS, implements a number of security functions through the incorporation of some security modules, specifically (i) a module to detect security anomalies around the Ethernet traffic, i.e., Suricata IoT IDS (https://github.com/decanio/suricata-IoT), (ii) a module to detect anomalies in wireless access networks like implementations based on 802.11 specifications, i.e., Kismet IDS (https://www.kismetwireless.net/), and (iii) a vulnerability scanning engine that builds and sends vulnerabilities reports, i.e., OpenVAS (http://www.openvas.org/). Each one of these modules is able to send security events toward a SIEM server implemented using OSSIM, which has a set of correlation rules, designed and implemented leveraging the proposed security architecture and multirelations mapping described in Section 4. Security events are sent using the Syslog standard.

Suricata IoT required 3 configurations, namely, definition of a common identity for all security events reported by the IoT device, definition of internal/external network addresses, and definition of IDS rules to be used (ours use just emerging-dos and emerging-webserver rules). Additionally, on the server side, an specific plug-in was set containing a regular expression for OSSIM to receive and understand the security events coming from Suricata. The security events received by the OSSIM contain the following fields: malicious source IP address, malicious source port, malicious destination IP address, malicious destination port, matched Suricata rule ID, and rule priority.

In turn, Kismet IDS required a wireless antenna able to work on monitor mode (TP-LINK WN722N). Kismet configuration required to set the wireless adapter in monitor mode and define the name of the interface that scan the media. Furthermore, it was required to implement a client to forward to OSSIM all messages generated and stored locally by Kismet IDS. Besides, on the server side, a specific plug-in was set containing a regular expression for OSSIM to receive and understand the security events coming from Kismet. The security events received by OSSIM contain the following fields: malicious source IP address, malicious source MAC address, malicious destination MAC address, wireless channel, and matched Kismet rule ID.

In the case of OpenVAS, it required the development of a client sending all the vulnerability reports toward OSSIM. This client was implemented using the API OpenVAS Management Protocol (OMP). Moreover, on the server side, a specific plug-in was set containing a regular expression for OSSIM to receive and understand security events coming from OpenVAS. The security events received for the OSSIM contain the following fields: vulnerability ID, IP address, vulnerability severity, and the Common Vulnerabilities and Exposures (CVE) code.

The IoT-secure device was a Raspberry Pi3 (1 GB RAM, Quad Core 1.2 GHz Broadcom BCM2837, 100 base Ethernet, 10 GB ROM) with an integrated wireless antenna TP-LINK WN722N. On the other hand, the SIEM server was implemented with OSSIM 5.4.1 (8 GB RAM, Intel(R) Xeon(R) CPU E5-2620 v3, 2.40GHz, 8 cores, 1 GG Ethernet, 100 GB ROM) with a total set of 16 plug-ins. OSSIM does not have an initial restriction regarding the creation of correlation rules. Yet, it does not have directives like DoS, network, and SCADA, which are only available for the commercial version. Correlation rules were implemented as indicated in the scenarios described in Section 5. Last but not least, the OSSIM server was successful to receive the security events form the IoT device, perform a match with the correlation rules, and generate an alert.

7. Conclusions and Future Work

The IoT world represents unquestionably a great opportunity to benefit our everyday life [55]. Several applications have been already developed to integrate IoT devices within the existing network infrastructures.

Along this paper, a comprehensive study around security aspects in IoT devices has been conducted, reviewing the most common security events, vulnerabilities, and attack surfaces. Specifically, we proposed a multirelations mapping among these three categories, which we believe may be helpful to support the security administrator to better understand the causes, symptoms, and attack vectors of security incidents. Specifically, security events have been used to generate correlation rules which are used by a SIEM system to detect security incidents.

Defensive security has also been explored describing different possible actions that could be implemented from a SIEM in order to decrease the impact of an incident (if it has already occurred) or avoid the progress of the incident (if it has been detected that an adversary is trying to exploit a vulnerability).

Future works include the definition of new techniques that allow the generation of correlation rules in a more automatic way, so the SIEM system may be able to face different combinations of events, vulnerabilities, and threats.

Automatic rules generation can be grounded on techniques like semantic web to build and maintain a complete taxonomy applicable for different components of an IoT ecosystem. This initiative would require a complete characterization of the IoT ecosystem to identify all the applicable attack surfaces and guarantee a proper security event generation coming from IoT devices.

Moreover, a concrete evaluation of the overhead needed to effectively introduce a SIEM solution within the IoT ecosystem is worth of investigating. Finally, we are currently working in the deployment of a SIEM solution over an IoT device thinking in situations where it is necessary to have a security solution just inside the IoT ecosystem.

Data Availability

The IoT vulnerabilities, attack surfaces, and event categories data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work has been supported by the European Commission Horizon 2020 Programme under Grant Agreement no. H2020-ICT-2014-2/671672-SELFNET (*Framework for Self-Organized Network Management in Virtualized and Software Defined Networks*), by a Ramón y Cajal Research Contract (RYC-2015-18210) granted by the MINECO (Spain) and cofunded by the European Social Fund, by a Leonardo Grant 2017 for Researchers and Cultural Creators awarded by the BBVA Foundation, and by the Colombian School of Engineering Julio Garavito (Colombia) through the project "Developing of a SIEM solution (Security Information and Event Management System) for information asset protection", approved by the Internal Research Opening 2016.

References

[1] K. Yelamarthi, M. S. Aman, and A. Abdelgawad, "An application-driven modular IoT architecture," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 1350929, 16 pages, 2017.

[2] Gartner, "Gartner's 2016 Hype Cycle for Emerging Technologies," 2016. [Online]. Available: https://www.gartner.com/newsroom/id/3412017.

[3] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[4] S. Haller, S. Karnouskos, and C. Schroth, "The Internet of Things in an Enterprise Context," in *Future Internet – FIS 2008*, vol. 5468 of *Lecture Notes in Computer Science*, pp. 14–28, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[5] A. Abdelgawad and K. Yelamarthi, "Internet of things (IoT) platform for structure health monitoring," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 6560797, 2017.

[6] B. R. Stojkoska, K. Trivodaliev, and D. Davcev, "Internet of things framework for home care systems," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 8323646, 2017.

[7] B. Gomes, L. Muniz, F. J. da Silva e Silva, L. E. Rios, and M. Endler, "A comprehensive cloud-based IoT software infrastructure for Ambient Assisted Living," in *Proceedings of the 2015 International Conference on Cloud Technologies and Applications (CloudTech)*, pp. 1–8, Marrakech, Morocco, June 2015.

[8] S. Charmonman and P. Mongkhonvanit, "Special consideration for Big Data in IoE or Internet of Everything," in *Proceedings of the 13th International Conference on ICT and Knowledge Engineering, ICT and KE 2015*, pp. 147–150, Thailand, November 2015.

[9] J. Granjal, E. Monteiro, and J. Sá Silva, "Security for the internet of things: a survey of existing protocols and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[10] A. Zanella, N. Bui, A. P. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.

[11] R. Khan, S. U. Khan, and R. Zaheer, "Future internet: the internet of things architecture, possible applications and key challenges," in *Proceedings of the 10th International Conference on Frontiers of Information Technology (FIT' 12)*, pp. 257–260, December 2012.

[12] V. Beltran, A. F. Skarmeta, and P. M. Ruiz, "An ARM-Compliant Architecture for User Privacy in Smart Cities: SMARTIE—Quality by Design in the IoT," *Wireless Communications and Mobile Computing*, vol. 2017, Article ID 3859836, 13 pages, 2017.

[13] Y. H. Hwang, "IoT security & privacy: Threats and challenges," in *Proceedings of the 1st ACM Workshop on IoT Privacy, Trust, and Security, IoTPTS 2015*, p. 1, Singapore.

[14] F. Gómez Mármol, M. Gil Pérez, and G. Martínez Pérez, "I Don't Trust ICT: Research Challenges in Cyber Security," in *Trust Management X*, vol. 473 of *IFIP Advances in Information and Communication Technology*, pp. 129–136, Springer International Publishing, Cham, 2016.

[15] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: the road ahead," *Computer Networks*, vol. 76, pp. 146–164, 2015.

[16] I. Yaqoob, E. Ahmed, M. H. U. Rehman et al., "The rise of ransomware and emerging security challenges in the Internet of Things," *Computer Networks*, vol. 129, pp. 444–458, 2017.

[17] D. S. Lavrova, "An approach to developing the SIEM system for the Internet of Things," *Automatic Control and Computer Sciences*, vol. 50, no. 8, pp. 673–681, 2016.

[18] P. Zegzhda, D. Zegzhda, M. Kalinin, A. Pechenkin, A. Minin, and D. Lavrova, "Safe integration of SIEM systems with Internet of Things: Data aggregation, integrity control, and bioinspired safe routing," in *Proceedings of the 9th International Conference on Security of Information and Networks, SIN 2016*, pp. 81–87, USA, July 2016.

[19] G. Ho, D. Leung, P. Mishra, A. Hosseini, D. Song, and D. Wagner, "Smart locks: Lessons for securing commodity internet of things devices," in *Proceedings of the 11th ACM Asia Conference on Computer and Communications Security, ASIA CCS 2016*, pp. 461–472, Xi'an, China, June 2016.

[20] M. Woschek, "Owasp cheat sheets," *pp*, vol. 315, p. 4, 2015, https://www.owasp.org/images/9/9a/OWASP_Cheatsheets_Book.pdf.

[21] ISO/IEC, "ISO/IEC 27032:2012 - Information technologyâ€"Security techniquesâ€"Guidelines for cybersecurity," https://www.iso.org/standard/44375.html, 2012.

[22] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: mirai and other botnets," *IEEE Computer Society*, vol. 50, no. 7, pp. 80–84, 2017.

[23] A.-R. Sadeghi, C. Wachsmann, and M. Waidner, "Security and privacy challenges in industrial internet of things," in *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC '15)*, pp. 1–6, IEEE, San Francisco, Calif, USA, June 2015.

[24] A. O. Prokofiev, Y. S. Smirnova, and V. A. Surov, “A method to detect Internet of Things botnets,” in *Proceedings of the 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 105–108, Moscow, January 2018.

[25] V. Adat and B. B. Gupta, “Security in Internet of Things: issues, challenges, taxonomy, and architecture,” *Telecommunication Systems*, vol. 67, no. 3, pp. 423–441, 2018.

[26] ISO/IEC, “ISO/IEC 27000:2018 - Information technology-Security techniques-Information security management systems-Overview and vocabulary,” https://www.iso.org/standard/73906.html, 2018.

[27] S. Gupta, B. S. Chaudhari, and B. Chakrabarty, “Vulnerable network analysis using war driving and Security intelligence,” in *Proceedings of the 2016 International Conference on Inventive Computation Technologies, ICICT 2016*, India, August 2016.

[28] J. R Vacca, *Network and system security*, S. Elliot, Ed., Syngress - Elsevier, 2014.

[29] Y. Chahid, M. Benabdellah, and A. Azizi, “Internet of things security,” in *Proceedings of the 2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems, WITS 2017*, Morocco, April 2017.

[30] R. Van Rijswijk and E. Poll, “Using trusted execution environments in two–factor authentication: comparing approaches,” ser. Lecture Notes in Informatics. 1em plus 0.5em minus 0.4em Bonn, Germany: Gesellschaft for Informatik, 9 2013, pp. 20–31.

[31] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, and G. Vassilacopoulos, “Enabling data protection through PKI encryption in IoT m-Health devices,” in *Proceedings of the 12th IEEE International Conference on BioInformatics and BioEngineering, BIBE 2012*, pp. 25–29, November 2012.

[32] W.-I. Bae and J. Kwak, “Smart card-based secure authentication protocol in multi-server IoT environment,” *Multimedia Tools and Applications*, pp. 1–19, 2017.

[33] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, “A survey of intrusion detection in Internet of Things,” *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017.

[34] W. Meng, “Intrusion Detection in the Era of IoT: Building Trust via Traffic Filtering and Sampling,” *The Computer Journal*, vol. 51, no. 7, pp. 36–43, 2018.

[35] I. Smith and D. Bailey, “IoT Security Guidelines for Endpoint Ecosystem,” GSM Association, Tech. Rep., 2016. [Online]. Available: https://www.gsma.com/iot/wp-content/uploads/2016/02/CLP.13-v1.0.pdf.

[36] P. Kasinathan, G. Costamagna, H. Khaleel, C. Pastrone, and M. A. Spirito, “Demo: An ids framework for internet of things empowered by 6lowpan,” in *Proceedings of the 2013 ACM SIGSAC Conference on Computer; Communications Security*, pp. 1337–1340, 2013.

[37] P. Nespoli and F. Gómez Mármol, “e-Health Wireless IDS with SIEM integration,” in *IEEE Wireless Communications and Networking Conference (WCNC'18)*, Barcelona, Spain, 2018.

[38] A. Sforzin, F. G. Marmol, M. Conti, and J. Bohli, “RPiDS: Raspberry Pi IDS — A Fruitful Intrusion Detection System for IoT,” in *Proceedings of the 2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pp. 440–448, Toulouse, France, July 2016.

[39] I. Kotenko and A. Chechulin, “Computer attack modeling and security evaluation based on attack graphs,” in *Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems, IDAACS 2013*, pp. 614–619, Germany, September 2013.

[40] I. Kotenko and A. Chechulin, “Common Framework for Attack Modeling and Security Evaluation in SIEM Systems,” in *Proceedings of the 2012 IEEE International Conference on Green Computing and Communications (GreenCom)*, pp. 94–101, Besancon, France, November 2012.

[41] G. Kambourakis, C. Kolias, and A. Stavrou, “The Mirai botnet and the IoT Zombie Armies,” in *Proceedings of the 2017 IEEE Military Communications Conference, MILCOM 2017*, pp. 267–272, USA, October 2017.

[42] D. Geneiatakis, I. Kounelis, R. Neisse, I. Nai-Fovino, G. Steri, and G. Baldini, “Security and privacy issues for an IoT based smart home,” in *Proceedings of the 40th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2017*, pp. 1292–1297, Croatia, May 2017.

[43] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, “IoT security: ongoing challenges and research opportunities,” in *Proceedings of the 7th IEEE International Conference on Service-Oriented Computing and Applications (SOCA '14)*, pp. 230–234, IEEE, Matsue, Japan, November 2014.

[44] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. Ullah Khan, “The rise of ‘big data’ on cloud computing: review and open research issues,” *Information Systems*, vol. 47, pp. 98–115, 2015.

[45] G. Shpantzer, “Implementing hardware roots of trust: The trusted platform module comes of age,” *SANS Whitepaper*, 2013. [Online]. Available: https://trustedcomputinggroup.org/wp-content/uploads/SANS-Implementing-Hardware-Roots-of-Trust.pdf.

[46] S. A. Mirheidari, S. Arshad, and R. Jalili, “Alert Correlation Algorithms: A Survey and Taxonomy,” in *Cyberspace Safety and Security*, vol. 8300 of *Lecture Notes in Computer Science*, pp. 183–197, Springer International Publishing, Cham, 2013.

[47] P. Cichonski, T. Millar, T. Grance, and K. Scarfone, “Computer Security Incident Handling Guide : Recommendations of the National Institute of Standards and Technology,” National Institute of Standards and Technology NIST SP 800-61r2, 2012.

[48] D. Díaz-López, G. Dólera-Tormo, F. Gómez-Mármol, and G. Martínez-Pérez, “Dynamic counter-measures for risk-based access control systems: An evolutive approach,” *Future Generation Computer Systems*, vol. 55, pp. 321–335, 2016.

[49] P. Nespoli, D. Papamartzivanos, F. G. Marmol, and G. Kambourakis, “Optimal countermeasures selection against cyber attacks: A comprehensive survey on reaction frameworks,” *IEEE Communications Surveys & Tutorials*, 2017.

[50] P. H. Gregory, *ISSP guide to security essentials*, vol. 12, Cengage Learning, 2014.

[51] Alienvault, “Insider's guide to Incident Response,” https://www.alienvault.com/resource-center/ebook/insider-guide-to-incident-response-download, 2017.

[52] E. Tittle, J. M. Stewart, and M. Chapple, *CISSP: Certified Information Systems Security Professional Study Guide*, vol. 7, John Wiley Sons, 2012.

[53] S. Perumal, N. Md Norwawi, and V. Raman, “Internet of Things(IoT) digital forensic investigation model: Top-down forensic approach methodology,” in *Proceedings of the 5th*

International Conference on Digital Information Processing and Communications, ICDIPC 2015, pp. 19–23, Switzerland, October 2015.

[54] RSA, "Two-Factor Authentication Is a Must for Mobile," 2016. [Online]. Available: https://www.rsa.com/en-us/blog/2016-06/two-factor-authentication-is-a-must-for-mobile.

[55] A. Soro, A. H. Ambe, and M. Brereton, "Minding the gap: Reconciling human and technical perspectives on the IoT for healthy ageing," *Wireless Communications and Mobile Computing*, vol. 2017, 2017.