A Risk Assessment Methodology for the Internet of Things

Sabrina Sicari*‡, Alessandra Rizzardi*, Daniele Miorandi§, Alberto Coen-Porisini*
*Dipartimento di Scienze Teoriche e Applicate, Università degli Studi dell'Insubria,
via Mazzini 5 - 21100 Varese (Italy)
§U-Hopper, via R. da Sanseverino 95, 38121 Trento, Italy
‡Corresponding author
Email: {sabrina.sicari; alessandra.rizzardi; alberto.coenporisini}@uninsubria.it,
daniele.miorandi@u-hopper.com

***Abstract*—Letting both data producers and data consumers be aware of the levels of security and privacy guaranteed within an IoT-based system represents an important goal to be pursued. In fact, the presence of multiple and heterogeneous data sources, as well as wireless communication standards, increases the risk of violation in IoT scenarios. Besides controlling the behavior of data sources and regulating the access to resources by the interested parties, it is also fundamental to investigate how trustworthy is the platform that manages the provided information and services. To this end, risk assessment techniques can be adopted, with the aim of evaluating the reliability and the robustness towards malicious attacks of the components belonging to the IoT platform. In this paper, a general-purpose methodology for assessing the risk is proposed to be applied to end-to-end systems. More in detail, the proposed approach takes into account both static and dynamic features/components of an IoT system in an *objective* manner, following the whole data life cycle. Such an aspect represents the main advantage of the presented solution, which is concretely demonstrated within the real prototype implementation of an existing IoT middleware, in order to prove its feasibility.**

I. Introduction

Nowadays, most of our every-day and business activities depend on computer-based systems, which are even more moving towards mobile applications. Such an evolution is encouraged by the diffusion of Internet platforms that allow to be connected everywhere in the world. In this scenario, Internet of Things (IoT) technology has emerged thanks to the availability of "smart" devices. They are able to embed wireless sensors, actuators, RFID, NFC, and other similar technologies, which allows such devices to acquire information from the surrounding environment. In this way, it becomes easier the transmission of a huge amount of heterogeneous data that can be used with the final aim of providing customized services to the interested users. This is, in few words, the revolution carried out by the IoT paradigm [1].

Obviously, in order to deal with such a huge amount of information, a scalable platform has to be designed and put in act. It should be able to gather data from heterogeneous sources, process and structuring the data chunks in a uniform representation, and, finally, share them in the form of innovative and useful services. Hence, the IoT platform must be able to manage the whole data life cycle in each specific context.

Note that, to encourage the spreading of IoT applications, end-users (i.e., mainly data consumers) should trust the IoT system that manages both their information and data gathered from unknown sources, expecting that it will provide services with a degree of confidentiality, integrity, availability, and privacy compatible with their needs. Unfortunately, IoT platforms, in general, actually guarantee neither proper security violation controls nor a well-defined assessment of the risk to which the system could be exposed. Users typically know only the interface of the system, but have a little knowledge of how their information are treated, processed and shared. Such aspects are also important for software developers who design and implement the functionalities for the IoT platform itself. Therefore, it is fundamental to carry out a risk evaluation, with three final goals:

- Assessing how much users should believe in the system trustworthiness
- Revealing weaknesses of the existing platforms
- Evaluating possible countermeasures or improvements of the actual system components, in order to make the platform more resilient towards malicious attacks.

To cope with such issues, in this paper, we present a risk analysis methodology, targeted to end-to-end systems, since it comprehensively considers the whole data life cycle of an IoT platform. The proposed approach takes into account both static and dynamic features/components of an IoT system and aims to reveal the existing

risks at the different levels of the data flow. Note that what is important, in the adopted solution, is not the *absolute* value assigned to each level of risk, but the structure of the analyzed metrics.

In order to demonstrate the feasibility of the presented solution, we concretely applied our risk assessment methodology to an existing IoT middleware, named *NetwOrked Smart objects (NOS)*, developed by the authors in [2]. NOSs are conceived as computationally powerful devices that are connected to create a distributed processing and storage layer able to manage the data acquired from large-scale IoT deployments, in a way closer to the actual data sources than a centralized solution. The potentialities of NOS' platform has been demonstrated by means of experiments conducted on a real prototype [3]. The following functionalities regarding security have just been integrated in the original middleware:

- Two key management systems by Dini et al. [4] and Di Pietro et al. [5] have been adopted, for securing the communications among NOSs and users/nodes with the distribution of well-defined keys [6]
- A novel algorithm for security level's assessment has been defined; it is able to perform an automatic evaluation of the information with respect to data sources behavior [2]
- A policy enforcement framework has been introduced; it provides a set of general-purpose rules aimed at regulating and controlling the actions performed by NOSs and reacting towards possible violation attempts [7]
- The enforcement mechanism just presented has been integrated with *AUPS (AUthenticated Publish&Subscribe system)*, a protocol able to effectively manage publications and subscriptions through MQTT interactions, securing the information sharing with users interested in the services provided by the IoT platform [8].

All these aspects will be further detailed in the paper. They just give an idea of the security modules included in the adopted IoT platform and point out that lots of attention has been paid to the security requirements. In fact, today security and privacy are central points of discussion, in the research field as well as in industries. To allow the growth and real diffusion of IoT paradigm, users must be protected against the violation of their privacy and of the confidentiality and integrity of their data [9]. Hence, the importance of conceiving a risk assessment analysis arises.

In the considered illustrative example of NOSs, the actual integrated functionalities, described above, are evaluated with respect of the proposed methodology, in order to point out their strengths and weaknesses and provide a vision to the users of the trustworthiness of this system. Note that, at the present time, NOSs perform, by means of the algorithms for the security assessment previously introduced, an analysis of the data on the basis of the behavior of the sources and evaluate the levels of confidentiality, integrity, robustness of the authentication mechanism, and privacy for each received information. Such a technique allows users to be aware of the trustworthiness of the services provided by the IoT platform, but, until now, we cannot infer anything about the reliability of the system itself. Such a work fills this lack and also demonstrates the feasibility of the approach adopted by NOSs. In addition, a concrete implementation of the analyzed methodology is provided by means of NOS's prototype. The integration of risk analysis with the algorithm for security assessment allows the distributed NOSs to perform a dynamic evaluation of the risk, on the basis of the actual data sources. The proposed solution will enable both data producers and consumers (e.g., stakeholders, organizations) to select the NOS to or from which send/receive information, according to the desired level of risk. Note that, if we do not consider the outcomes of the algorithm presented in [2], the proposed risk evaluation methodology could be also executed in offline mode, as clarified during the presentation of the analyzed case studies.

It is worth to remark that the main advantage brought by the use of a risk assessment methodology with respect to our previous work (which includes [6], [2], [7] and [8]) is that an impartial tool is introduced in the IoT platform, with the aim of evaluating its trustworthiness. To achieve such a goal, the risk assessment tool must identify the functionalities provided by the other security tools (as the ones just introduced [6], [2], [7] and [8]) and their interactions with respect to the possible threat towards the IoT system. Hence, without the presence of a risk assessment tool, the systems' administrator as well as the end-users may believe that the system is robust because they trust its single components; while the risk assessment tool guarantees a wider vision of the whole system, being able to discover possible weaknesses. Note that the proposed methodology aims to be applied to any general-purpose system.

Finally, we focus on NOSs platform, although there are other solutions available (such as openHab[1], Thread[2], AllJoyn[3], Amazon Web Service (AWS) IoT[4], IBM Watson IoT[5], FiWare[6]), due to its modular and cross-domain nature and built-in security functionalities, as described in detail during the paper.

The paper is organized as follows. Section II reviews

[1]https://www.openhab.org
[2]https://threadgroup.org
[3]https://allseenalliance.org/framework
[4]https://www.amazonaws.cn/en/iot-platform
[5]https://www.ibm.com/internet-of-things
[6]https://www.fiware.org

the relevant literature and state of the art about risk assessment in general and, in particular, in the IoT field; then the technique adopted in this work for evaluating the risk of NOS system is detailed. Section IV presents NOS architecture and functionalities. Section V presents the risk analysis of NOS system; while Section VI discusses a prototypical implementation of the proposed approach. Section VII ends the paper and provides directions for future research.

II. Related Work

Nowadays, it is fundamental to provide automatic methods for performing the risk evaluation of existing platforms, mainly due to the spreading of IoT devices, could computing, and social networks. Moreover, it is important to conceive mechanisms able to impartially judge the degree of trustworthiness of a structured platform, in order to improve the trust of the users towards the platform itself and also persuading new users to join. In this direction, several works deal with the problem of risk assessment in different application scenarios.

Fox example, as regards cloud services, the authors of [10] make use of three risk models for assessing the risk into their smart home architecture, namely: legal risk, which aims at evaluating if the privacy of users' data, stored in the cloud, is preserved; appliance failure risk, which is related to the compliance with the protocol established by the smart home and the cloud; resource security risk, which monitors possible external threats towards the devices that belong to the smart home. Such metrics may reveal to the owners of the smart home if one or more issues, that may have an impact on the security or on the correct functionality of the smart home system itself, are put in place. In this way, proper actions may be performed to restore the situation and avoiding to waste smart home resources or compromise users' security and privacy. However, a way to present (and communicate in real time) the calculated risk assessment to non-technical users has not been provided yet.

Another contribution which, instead, provides a dashboard for the safety management team, regards the risk analysis in Smart Work Environments (SWEs) [11]; they allow to monitor the activities performed by the workers, by the used tools, and also by the machinery in the workplaces. As a consequence, it is possible to obtain useful information from the smart objects, that interact in SWE by means of semantic services. Such continuous information may also help to early recognize, detect or prevent malfunctions or attacks to the system and, also, to assist the safety management team in decision making about risks. To this end, the authors introduce the dashboard named RAMIRES (Risk-Adaptive Management in Resilient Environments with Security), that implements the MAPE (Monitor-Analyze-Plan-Execute) methodology along with a new ontology for specifying the involved resources, the rules and the desired constraints. A proper methodology for analyzing the defined ontology and for introducing new ontology classes at run-time is planned to be investigated by the authors in the next future, in order to obtain a general-purpose solution.

Complementary to the provision of a dashboard, as RAMIRES, for visualizing the results of risk evaluation, the work in [12] shows the potentiality of virtual environments in being valuable tools to assess the security properties and to discover the vulnerabilities of IoT devices, in realistic scenarios. The authors propose the SmallWorld platform, which is able to support hardware virtualization technologies, cloud computing, and to simulate real IoT devices and malicious behaviors, thus allowing to test the design of an IoT system along with the possible security issues before the real deployment. The effectiveness of SmallWorld has been demonstrated by means of a case study regarding a smart home application.

Also, the work presented in [13] proposes a risk analysis conducted on a smart home automation system, connected to the cloud and with mobile and IoT devices, able to manage the smart home functionalities from a remote position. Note that, tracing the users' activity within a smart home may lead to the collection and misuse of personal data, that can make users and homes vulnerable to various kinds of attacks or intrusions. The well-known Information Security Risk Analysis (ISRA) method [14] has been adopted for performing the assessment. More in detail, the smart home automation system has been divided into five parts, namely: software, hardware, information, communication protocols, and human actors. Then, the risk exposure for each of them is estimated on the basis of its capability to fulfill the three basic goals of system security (i.e., confidentiality, integrity, and availability). Each of the five parts has also been analyzed in search for vulnerabilities and threats, with the help of a group of experts. A total of 32 risks were identified during the risk analysis process and a probability in the range [1:5] was assigned to each of them using the input provided by the experts. Four risks were classified as "high" and concerned human factor and software components. Thus, the authors concluded that the main risk for security derives from the software used by the components of the smart home system. Therefore, in the next future, particular attention has to be paid by smart home automation system's developers towards the development of security and privacy strategies by design (i.e., in the early phases of system's design). At the same time, appropriate automated risk analysis tools should be made available to the develop-

ers, in order to quickly reveal the vulnerabilities of the system.

In this paper, we underline such requirements in a more general IoT context and, at the same time, we apply an effective risk assessment approach, able to point out the robustness of the IoT system taking into account the involved entities and the possible threats. The final goal is to provide the users and the developers with a mean for checking during the time the level of trustworthiness of the various components of an IoT-based environment. It is worth to remark that our proposed treatment aims to pursue an *objective* risk assessment. Such an *objectivity* will be gained by considering values in the evaluated metrics which will not be *absolute* measures of risk, but, instead, relative ones. In fact, the focus is on the structure of the adopted metrics [15] [16]. Therefore, the information computed by our model can be used as a decision support, as we will demonstrate.

Other existing approaches for risk assessment are not specifically targeted to IoT or to smart environments. A complete survey that present a taxonomy of ISRA methods is provided in [17]. As pointed out in [18], it is difficult for the organizations that want to conduct risk assessment to choose a method able to meet their needs. To cope with such an issue, [18] proposed a framework of info-structure for ISRA that should help organizations in select the most suitable solution.

Nevertheless, establishing a standard is a very complex task, since risk analysis intrinsically depends on the application environments, that often show different security and privacy requirements. As an example, the IoT context is characterized by lots of heterogeneous devices with respect to a cloud-based system, thus the IoT requires more strict control over the access to the network resources. A recent attempt to fulfill this gap is provided in [19], which introduces a framework for online services security risk assessment and management, that was designed in accordance to the widely-accepted practices standardized by the ISO/IEC 31000:2009. It can be used by both service providers and service consumers and it is based on the definition of a threat model; then sets of vulnerabilities are identified starting from the defined threats. The effectiveness of the framework has been validated by means of a case study carried out within a large enterprise. However it is not mature enough to be integrated in wider environments, such as IoT.

What emerged from the analysis carried out about the state of the art is that no standard or wide-accepted methods have been, until now, agreed by the scientific community for risk assessment in large applications and general-purpose scenarios. For such reasons, in this paper, we decided to adopt an existing general-purpose approach whose feasibility was proven in several real case studies, such as the network infrastructure of a Department of Computer Science [16] and VoIP solutions [15]. As many risk assessment methods, the one proposed in [16] is based on a mathematical model, detailed in [20] and [21], and takes into account the vulnerabilities and the logical dependencies emerging from the analyzed system, as specified in the next section. In our opinion, the method proposed hereby would help to improve security management and attacks' detection in medium and small IoT systems, mainly in the presence of a clear information workflow or product life-cycle. In fact, the system's administrators could state the rules and access to the IoT resources, which could be integrated in the risk assessment and analysis, in order to easily detect violations. Other advanced systems, such as OWASP[7] (open source) and Symantec[8] (proprietary software) do not allow such a level of customization and fine-grained control by the IoT platform's administrator. Whereas, with regards to large-scale deployments, further functionalities should be probably provided, in order to automate the rules' definition in more complex systems, as we hint in the conclusions. This is due to the fact that it is fundamental, for the proposed approach, to be able to scale even in presence of a huge number of involved rules.

III. The adopted risk assessment method

This section describes the risk assessment method adopted in our solution and integrated into the IoT context. Such a mechanism has been proposed in [16] and measures the exploitability E of performing a certain kind of attack. To achieve such a value, the risk of a threat can be obtained following five steps. Table II summarizes the notations used in the paper.

1) **Step 1:** a threat towards the system to be examined has to be modeled by means of an attack tree. The root of the tree corresponds to the considered attack; while the children nodes represents the different ways that can be pursued to carry out the attack. Then, the leaves are the identified vulnerabilities v_i
2) **Step 2:** an initial exploitability value E_0 in the range (0:10) has to be associated to each vulnerability v_i belonging to the set of identified vulnerabilities V; E_0 measures how probable is that v_i is exploited to perform a successful attack. The evaluation of E_0 is done in a qualitative manner, considering the levels of ease and difficulty through which an attack against the system can be performed. Such qualitative values are then translated into quantitative ones (in the range (0:10), as

[7]https://www.owasp.org/

[8]https://www.symantec.com/it/it/solutions/internet-of-things

just said); Table I represents the conversion from qualitative to quantitative values used in this work. More in detail:

$$\begin{aligned} \forall v_i \in V \quad \text{assess} \quad E_0(v_i) \\ \text{where} \quad E : V \mapsto N \end{aligned} \tag{1}$$

N is a total ordered set of degrees of exploitability, as just hinted. "0" means that the vulnerabilities is "not exploitable at all"

3) **Step 3:** a graph that highlights the dependencies d_i in the set D (i.e., the edges) among the identified vulnerabilities has to be drawn. Note that a vulnerability v_1 depends on a vulnerability v_2 if and only if when v_2 has been already exploited, then v_1 is easier to be exploited
4) **Step 4:** an exploitability value $E(d_i)$ has to be assigned to the edges of the graph, as done for the E_0 values at step 2
5) **Step 5:** the exploitability values E_i has to be updated taking into account the dependencies just individuated, according to the following iterative formula, starting from the values E_0, previously calculated:

$$\begin{aligned} \forall v_i \in V, \forall d_i \in D : \\ E_{i+1} = max(E_0(v_i), min(E(d_i), E_i(v_i))) \end{aligned} \tag{2}$$

TABLE I: Conversion between qualitative and quantitative evaluations

very easy	easy	on average	difficult	very difficult
9	7	5	3	1

Therefore, high values of exploitability (i.e., 7, which means that it "easy" to exploit the attack, and 9, which means that it "very easy" to exploit the attack) suggest that the system is very vulnerable towards a particular kind of attack. Such an exploitability value gives an idea of the degree of trustworthiness associated to the evaluated system. In that sense, if this value is lower (i.e., 3, which means that it "difficult" to exploit the attack, and 1, which means that it "very difficult" to exploit the attack), than the system's trustworthiness degree is higher.

Note that an advantage of this approach is that it can be also used to evaluate the impact of adding or removing a component to/from an existing system. In fact, vulnerabilities and dependencies would be updated and, then, new exploitability values will be obtained. Such an aspect is particularly relevant for the kind of system we want to analyze in this paper. In fact, as introduced in Section I, NOS platform is conceived as a modular middleware. Hence, it is very interesting to assess the contribution of each security module that composes the system. Another important feature is that the adopted schema is general-purpose and, thus, fits the needs of IoT applications. In the following sections, NOS middleware will be detailed, before presenting the risk analysis methodology.

TABLE II: Notations

Acronym	Meaning
E	Exploitability of performing a certain kind of attack
E_i	*i-th* exploitability
V	Vulnerability of the system
v_i	*i-th* vulnerability
D	Dependency among vulnerabilities
d_i	*i-th* dependency
a_n	Set of threats/attacks
a_i	*i-th* threat/attack
c_m	Set of security countermeasures
c_i	*i-th* countermeasure
w_{a_i,c_i}	Weight associated with the relatioship among a_i and c_i

IV. Illustrative Example: IoT Platform

This work demonstrated the feasibility of the proposed risk assessment methodology by applying it in an existing flexible and cross-domain IoT-based middleware, named *NetwOrked Smart object (NOS)*, defined the first time in [3], using the technique presented in Section III. NOSs are able to manage in a distributed way the data provided by heterogeneous sources and evaluate, by means of proper algorithms [2], the security and data quality of the information, in order to allow the users to be aware of the levels of reliability and trustworthiness of the services gathered by NOSs themselves. NOSs also provide a lightweight and secure information exchange process, based on an authenticated publish and subscribe mechanism [8] using the MQTT protocol. In the next sections, the architectural components of NOSs will be detailed, as long as the actual enforcement framework, defined in [7] and, in particular, the algorithm used for the security assessment and AUPS. In fact, in order to understand the risk analysis, presented in Section V, it needs to provide necessary details to the reader on how NOSs work.

A. Networked Smart Object Architecture

Two main entities compose a typical IoT system: (i) the nodes, conceived as heterogeneous devices (e.g., RFID, NFC, actuators, sensors etc.) which generate data for the IoT platform; (ii) the users, who interact with the IoT system through services making use of such IoT-generated data, typically accessing them by means of a mobile device (e.g., smartphone, tablet) connected

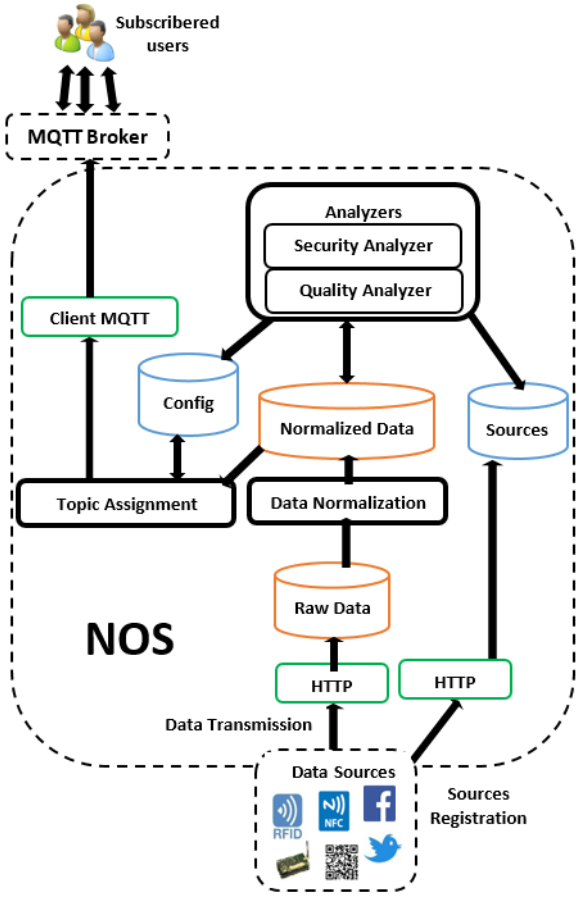

Fig. 1: NOS architecture

to the Internet (e.g., through WiFi, 3G, or Bluetooth technologies).

Therefore, interfaces for the communications of NOSs with the data sources (i.e., the nodes) and with the users, respectively, have been defined.

HTTP protocol is adopted by NOSs for collecting the data from the IoT devices and for allowing sources' registration. In fact, NOSs deal both with registered and non-registered sources. The registration is not mandatory, but it provides various advantages in terms of security, since registered sources may specify an encryption scheme for their interactions with NOSs, thus increasing the level of protection of their communications (encryption keys' distribution is made by the algorithms presented in [6]). The information related to the registered sources are put in the storage unit, named *Sources*. Instead, for each incoming data, both from registered and non-registered sources, the following information is gathered: (i) the kind of data source, which describes the kind of node; (ii) the communication mode, that is, the way in which the data are collected (e.g., discrete or streaming communication); (iii) the data schema, which represents the type (e.g., number, text) and the format of the received data; (iv) the data itself; (v) the reception timestamp.

Since the received data are of different types and formats, NOSs initially put them in the *Raw Data* storage unit. Data in such collection are periodically processed, in a batch way, by the *Data Normalization* and *Analyzers* phases, in order to obtain an uniform representation and add useful metadata regarding the security (i.e., level of confidentiality, integrity, privacy and robustness of the authentication mechanism) and data quality (i.e., level of accuracy, precision, timeliness and completeness) assessment. Such an assessment is based on a set of rules stored in a proper format in another storage unit, named *Config*, and are detailed in Section IV-C.

Such a mechanism allows users who access the IoT services to filter directly by themselves the data processed by NOSs according to their personal preferences. Note that data sharing to the interested users is performed by means of Message Queue Telemetry Transport (MQTT) protocol [22]. To this end, a topic is assigned by NOSs to each processed data. Figure 1 summarizes the NOS's components just introduced.

NOSs modules interact among themselves through *RESTful* interfaces; they have been implemented in a real prototype, which is openly accessible at https://bitbucket.org/alessandrarizzardi/nos.git. *Node.JS* platform [23] has been used for developing NOSs' core operations, *MongoDB* [24] has been adopted for the data management, and Mosquitto [25] has been chosen for realizing the open-source MQTT broker. For more details about the implementation, we refer to [2].

B. Networked Smart Object Enforcement System

The enforcement framework integrated within NOSs [7] is responsible for properly managing the available resources and handling possible violation attempts, by means of well-defined policies. A set of primitives, able to specify and enforce a large variety of attribute-based policies have been identified. More in detail, they regulated the following tasks: node access control, node data transmission, node data processing, user access control, user service request, and service provision.

It includes, for each NOS: (i) a *Policy Enforcement Point (PEP)*, which is the point that intercepts the requests of access to resources from users, and makes the decision requests to *Policy Decision Point (PDP)*, in order to obtain the access decision (i.e., approved or rejected); (ii) a *PDP*, which evaluates the access requests against the authorization policies, before taking the authorization decisions; (iii) a *Policy Administration Point (PAP)*, which contains the authorization policies established by the system administrators.

Policies have been expressed with a proper interoperable specification language, based on *JSON* syntax. It is flexible enough to represent the IoT heterogeneous analyzed context both in a general-purpose and in a customizable way. The considered access control model is the *Attribute Based Access Control (ABAC)* [26], because it guarantees the possibility of applying fine-grained access operations. Such attributes may be associated to both data sources and users, thus enabling the activation of different kinds of policies. Note that users have to complete a registration phase before interacting with NOSs; during this phase, a set of attributes is assigned to them on the basis of their role in a the specific application context, thus allowing them to access information regarding certain topics or not.

C. Security Assessment

NOSs exploit an algorithm valid for both registered and anonymous (i.e., non-registered) sources, which associates a score in the range $[0, 1]$ for the security metrics, intended as levels of *confidentiality* and *integrity* of the information transmitted to NOSs, *privacy* of the transmitting source and *authentication* (i.e., the robustness of the source authentication towards NOS). The security assessment algorithm, presented in [2], takes into account two sets of parameters:

- A set of threats/attacks a_n, which includes the attacks that may be carried out towards the sources or the data transmitted to NOSs (e.g., data violation, unauthorized access, masking, impersonation)
- A set of security countermeasures c_m, which regards the countermeasures made available by NOSs in order to face the attacks included in a_n (i.e., encryption, authentication, key pre-distribution).

The security model considered by the algorithm links the attacks of a_n with the corresponding countermeasures in c_m, following the taxonomy retrieved from [27]. In detail, the algorithm considers each countermeasure to present a degree of resistance to a violation or to an attack attempt. The relationship among attacks and countermeasures is many-to-many, because an attack can be tackled through a plurality of countermeasures and a countermeasure can face more than one attack. Each relationship is associated with a weight w_{a_i,c_i} in the range $[0 : 1]$, which represents the level of robustness of the countermeasure c_i with respect to the attack a_i, as represented in Figure 2.

The identified relationships are clustered into four groups, one for each security metric to be analyzed, as follows:

- g_{conf} for attacks-countermeasures related to data confidentiality
- g_{int} for the pairs related to data integrity

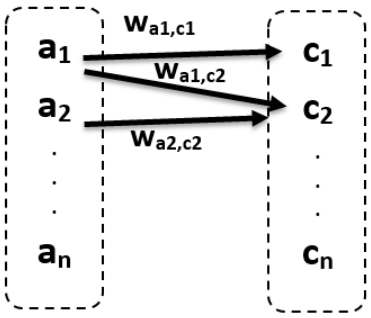

Fig. 2: Weighed relationships among attacks and countermeasures

- g_{pri} for privacy issues
- g_{auth} for the pairs concerning source authentication.

Note that such groups are not necessarily disjoint. Once the groups are defined, NOSs can update the weight corresponding to the relationships among attacks and countermeasures depending on the sources' behavior and, consequently, on the basis of the received data. NOSs recognize the possible malicious activities which can occur using a monitoring system (i.e., a sort of intrusion detector).

Weights can vary over time in a dynamic way; such variations depend on context changes such as a source updating the its key or the adopted encryption scheme. Such a process of automatic adjustment is performed by means of a well-known learning approach, namely difference temporal learning [28]. In this way, the weights may decrease over time with the observations of system or data violations, but they may also increase if a certain countermeasure turns out to be more resilient or if an attack is no longer performed. For the formula, please refer to [2].

Note that, by means of this algorithm, NOSs are not able to directly counteract malicious devices, but to recognize that the data provided by a source is corrupted or presents a poor level of confidentiality/privacy and discard it as unsuitable.

D. Secure publish&subscribe protocol

NOSs functionalities have been further extended with *AUPS (AUthenticated Publish&Subscribe system)* [8], which represents a new secure MQTT mechanism integrated with the policy enforcement framework, presented in Section IV-B.

More in detail, the MQTT broker has to interact with the underlying PEP on NOSs in order to accept or deny subscription requests on the basis of the user credentials and owned attributes.

A new component, named *Key Topics Manager (KTM)*, has been introduced, which is in charge of managing temporary keys for topics access control (i.e., the keys are used for encrypting the data before sending them to the broker - and then to the users - for the notifications). Note that the keys are not fixed, but they come with an expiration timestamp, in order to improve the system's resilience towards malicious attacks (e.g., man in the middle attacks, replay attacks, password discovery).

V. Risk Analysis

Once NOSs architectural components and main security functionalities have been presented, the aspects related to the control of violations and data assessment have been clarified. For these purposes, an enforcement system and an algorithm for data/sources evaluation have been introduced. In this way, the users who exploit the IoT services provided by NOSs can be aware of the degree of trustworthiness of the received information, but they are completely not aware about the reliability of the IoT platform. The same is for the software developers and/or the NOSs' administrators, who design and implement NOSs functionalities. However, it is fundamental to know which are the weaknesses of a running system in order to improve the adopted security modules and guarantee a transparent experience to users, mitigating the overall risk. To this end, a risk assessment evaluation, able to reveal the vulnerabilities of a complex system as the one presented in this paper, is proposed. Note that a comprehensive list of threats and vulnerabilities can be found in ISO 27001/ISO 22301[9], CERT taxonomy[10], and STRIDE threat model[11]. In this work, we extracted some threats/vulnerabilities from such lists to describe and analyze our presented solution, as detailed in Sections V-A and V-B.

According to the risk analysis method presented in Section III, we have to fulfill the following steps:

1) Define the model's components and their interaction by means of direct links
2) Identify the possible vulnerabilities and depict an attack tree for a specific risk
3) Identify the possible dependencies among the vulnerabilities
4) Assign the exploitability values to each identified vulnerability.

As regards the first step, the considered scenario is composed by: (i) the sources that transmit data to NOSs by means of HTTP connections; (ii) the NOSs that receive, process and then share the data with the authorized users; (iii) the users who exploit the services by means of MQTT notifications mediated by a broker. A schema is shown in Figure 3.

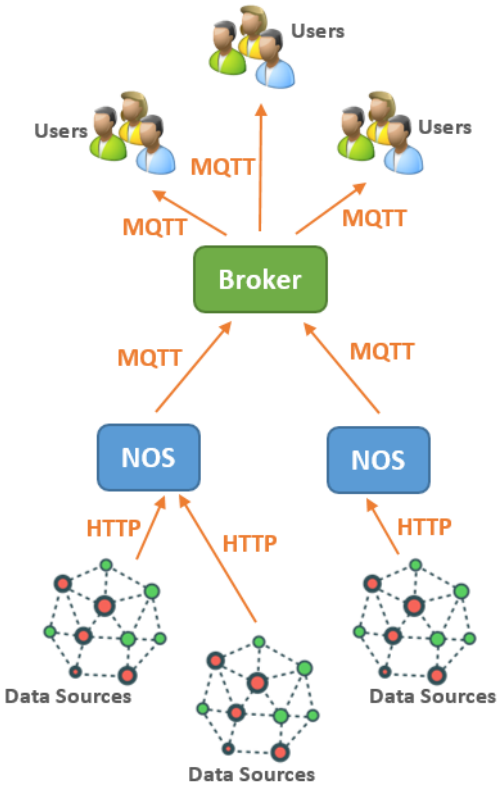

Fig. 3: Risk assessment scenario

Three main threats may happen within the presented scenario and will be analyzed in the following sections: (i) a non-authorized user intercepts information of a certain topic for which he/she has no right of access; (ii) an attack to the network resources is carried out towards NOSs middleware (e.g., denial of service) and NOSs do not put in act any countermeasures; (iii) a denial of service (DoS) attack is carried out towards NOSs middleware and NOSs react by means of REATO [29], which is a technique, developed by the authors, for recognizing and counteracting DoS attacks, as detailed in the following.

It is worth to remark that a precondition of such an analysis is that NOSs are installed on a secure and stable platform. Therefore, users are supposed to trust that NOS middleware's functionalities behave in the expected manner, as detailed in the previous sections. A similar situation happens with the cloud, where users who buy a cloud service suppose that the system runs on a secure platform, besides respecting the agreed data management's terms and conditions. Summarizing, there is no way for directly pursuing an attack to NOSs'

[9]https://advisera.com/27001academy/knowledgebase/threats-vulnerabilities/

[10]https://www.enisa.europa.eu/topics/csirt-cert-services/community-projects/existing-taxonomies

[11]https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

functionalities, but a physical attack to the platform on which the NOSs run (i.e., a Raspberry Pi, as explained in the following) may partially compromise the IoT system (e.g., if one or more Raspberries Pi are attacked). In this paper, we do not consider such a kind of situation, because it strictly depends on the resilience of the runtime platform adopted (e.g., possibility of accessing to the Raspberry's shell); while the risk analysis carried out hereby performs an assessment of the IoT middleware from a functional point of view. Hence, such an analysis is independent from the chosen platform. In the future, further investigation could be performed in order to compare the risk associated to NOSs installed in different environments (e.g., a distributed network composed of Raspberries, a cloud platform, and so on).

A. First Case Study

We start from the first case study with the identification of the possible vulnerabilities and the definition of the attack tree in Listing 1. An attacker intercepts the data, exchanged by means of MQTT protocol among the broker and the users (or NOSs and the broker), if he/she/it is able to listen the communication and understand its content. In fact, the data transmitted by NOSs are encrypted with proper keys established on the basis of the topic assigned to the data themselves. Therefore, only the users who own the correct decryption keys for that topic should be able to discover the data content.

The attack tree points out these three sub goals and their internal relationships, expressed by "AND" and "OR" implications: (1) the attacker gets a copy of the communication; (2) the attacker decodes the encrypted content; (3) the attacker succeeds in authenticating himself/herself/itself as a legitimate user. In order to intercept a data or a set of data under a specific topic, an attacker should identify possible paths of notification from the broker to the users or, alternatively, tries to connect to the administrator channel from one NOS to the broker. In these ways, the intruder can either control the desired subscribed users or poison the notification path towards the malicious device in order to control the traffic in that direction. Then, once discovered such a path, the attacker should decode the information content by understanding the encryption algorithm and/or discovering the encryption key of the topic of interest. If the attacker achieves the credentials of a legitimate user, then he/she/it can obtain the access to the desired information and, also, subscribe to new topics. Such considerations are pointed out in Listing 1.

```
Goal: Unauthorized access to data
1. To intercept the data during its
    transmission
  1.1 To identify a possible path of
    notification of a certain topic
      1.1.1 To choose and control one or more
    subscribed users to that topic (V1)
     OR 1.1.2 To connect to the administration
     channel among a NOS and the broker (V2)
 OR 1.2 To divert the notifications towards a
    malicious device
     1.2.1 To identify a possible path of
    notification of a certain topic (as 1.1)
     OR 1.2.2 To poison the notification path
    between a user and the malicious device
            1.2.2.1 To identify a specific
    topic communication in the traffic (V3)
OR 2. To decode the communication content
   2.1 To understand the encryption algorithm
      2.1.1 To guess the encryption algorithm (
    V4)
     OR 2.1.2 To read the encryption algorithm
     in a controlled channel (V5)
  AND 2.2 To determine the encryption key for
    that topic
      2.2.1 To guess short or weak key (V6)
     OR 2.2.2 To sniff the key where it is
    transmitted by NOSs to the user (V7)
OR 3. To impersonate a legitimate user
    3.1 To determine the credentials of a
    legitimate user
      3.1.1 To guess short or weak key (as
    2.2.1)
     OR 3.1.2 To sniff the key where it is
    agreed or updated by NOS and user (as
    2.2.2)
     AND 3.1.3 Access to the desired topics (
    V8)
```

Listing 1: Attack tree - case study 1

Table III summarizes the vulnerabilities identified in Listing 1.

TABLE III: Vulnerabilities - case study 1

Vulnerability	Description
V_1	Information disclosure by one or more users
V_2	Information disclosure by broker/NOS connection
V_3	Topic disclosure by traffic analysis
V_4	The encryption algorithm is weak
V_5	The encryption key is weak
V_6	The user has a weak authentication mechanism
V_7	A link connected to the identified user can be sniffed
V_8	User identity stolen

The next step concerns the identification of the dependencies among the vulnerabilities just presented. The dependency graph is shown in Figure 4. It reveals that adopting a weak authentication mechanism in the communications among NOSs, broker and users (i.e., vulnerability V_6) has impact on the disclosure of various information about topics and users, represented by the other vulnerabilities. Moreover, vulnerability V_7 influences the sniffing of information regarding the topics

(i.e., vulnerabilities V_1, V_2, V_3) and the users (i.e., vulnerability V_8).

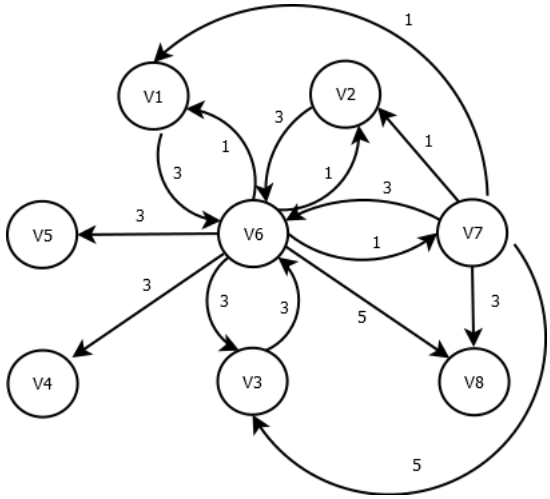

Fig. 4: Dependency graph - case study 1

The final step is the assignment of the exploitability values E_i to each identified vulnerability. Table IV shows the values calculated for E_0 and E_2, so until the time when the system converges, following the formula presented in Section III, considering both vulnerabilities and their dependencies. We established that the conversion between the qualitative and the quantitative evaluations is that of Table I (the reported values are the same of [15]). Here, the used scale decreases from high values if it is easy to exploit the vulnerability to low values if it is difficult to exploit the vulnerability.

Note that the values chosen for E_0 have been established considering the functionalities of NOSs. In particular, we stated that the exploitability of V_4 is "difficult", since NOSs adopt, by default, no weak encryption algorithms, but it could depend, in some cases, on the presence of constrained devices. Also the exploitability of V_5 is set to "difficult" since the KTM ensures to always dispose of updated and robust keys (see Section IV-D). The exploitabilities of V_6 and V_8 are influenced by both V_4 and V_5, so they are set to "very difficult" and "difficult". Note that, even if an attacker steals the decryption keys of a user, they will change in a short period thanks to the KTM's task. Whereas, the exploitability of V_7 is set to "difficult" due to the mechanism put in action by AUPS (see Section IV-D) for preventing unauthorized information disclosures; such a behavior leads to set the values of exploitability for V_1, V_2 and V_3 to "very difficult". Instead, the initial weights set up for the dependencies are directly shown in Figure 4.

The results show that NOSs' system is resilient towards the unauthorized access to data. All the vulnerabilities present a value for the exploitability equal or lower than 5. The main risk regards the user side, due to the fact that NOSs have no control over the user devices and on how users manage their confidential credentials (e.g., users may connect to unsecured networks or leave the keys or passwords in unsecured places). Such an uncertainty is balanced by the robust topic encryption mechanisms put in place by NOSs by means of AUPS and of the enforcement framework. Such outcomes may dynamically vary in case one or more NOSs components are enabled/disabled for a certain time, due to maintenance or malfunction. In such kinds of situation, the levels of exploitability are updated, since the active/inactive security functionalities have a great impact on the trustworthiness of the IoT platform, as detailed in Section V-D.

TABLE IV: Exploitability values - case study 1

	E_0	E_1	E_2
V_1	1	3	3
V_2	1	3	3
V_3	1	3	5
V_4	3	3	3
V_5	3	3	3
V_6	1	3	5
V_7	3	5	5
V_8	3	3	5

B. Second Case Study

The second case study regards possible attacks to the network resources, which may include: (i) the sending of invalid data by malicious sources, that causes an abnormal behavior of one or more services offered by NOSs; (ii) the flooding of non-useful traffic towards one or more NOSs in order to overload the network; (iii) this may also cause the blocking of the traffic, which results in a loss of access to network resources by authorized users.

Listing 2 highlights the identified vulnerabilities within the attack tree by means of four sub goals. An attacker may impersonate a non-registered data source, which is allowed to send data to NOSs. This is a design choice of NOSs middleware, that also accepts data from unknown sources (in order to be compliant with the dynamic nature of IoT scenarios); the expected outcome is that the data provided by such a kind of sources are characterized by low scores in terms of security and data quality (see Section IV-C).

However, both such "illegitimate" sources and "legitimate" ones may flood the network with invalid data sent to one or more NOSs, thus preventing valid data to reach NOSs (e.g., they can be lost during transmission due to

network congestion) and, at the same time, providing bad information to the services offered by NOSs. Moreover, a block of the traffic may also happen. In this case, we refer to a Denial of Service (DoS) attack, that can also bring to a Distributed DoS (DDoS) attack if more than one NOS is involved.

Finally, a legitimate (and registered) data sources may be impersonated, by sniffing or guessing its credentials, thus being allowed to pursue the aforementioned attacks.

```
Goal: Attack to network resources
1. To send invalid data
  1.1 To connect to a NOS
      1.1.1 To impersonate a non-registered
    data source (V1)
     OR 1.1.2 To access the system as a
    legitimate data source (see 4.1)
     OR 1.1.3 To access and modify the
    database
            1.1.3.1 SQL injection from the NOS
     web dashboard (V7)
           OR 1.1.3.2 Log into the database (
    V8)
           OR 1.1.3.2 Exploit a buffer
    overflow (V9)
OR 2. To flood the network
   2.1 Denial of Service (DoS)
      2.1.1 To send a huge amount of invalid
    data to one NOS (V2)
     AND 2.1.2 To prevent legitimate data to
    reach NOSs by filtering the traffic (V3)
   OR 2.2 Distributed Denial of Service (DDoS)
    (see 2.1)
AND 3. To block the traffic
   3.1 Denial of Service (DoS) (see 2.1)
   OR 3.2 Distributed Denial of Service (DDoS)
    (see 2.2)
OR 4. To impersonate a legitimate and
    registered data source
   4.1 To determine the credentials of a
    legitimate data source
      4.1.1 To guess short or weak key (V4)
      OR 4.1.2 To sniff the key where it is
    agreed or updated by NOS and the source (V5
    )
      AND 4.1.3 Access to the data acquired by
     the impersonated source (V6)
```

Listing 2: Attack tree - case study 2

Table V summarizes the vulnerabilities identified in Listing 2.

The dependency graph of the presented vulnerabilities is shown in Figure 5. It reveals that impersonating a legitimate data source (i.e., vulnerabilities V_1, V_5 and V_6) has impact on the provision of invalid data to NOSs and, as a consequence, to the users (i.e., vulnerability V_2). Such a vulnerability V_2 has further impact on the loss of valid data (i.e., vulnerability V_3). Moreover, vulnerabilities V_4 and V_5 influences the source identity stolen (i.e., vulnerability V_6). Vulnerability V_3 is also associated to possible occurrences of buffer overflows (i.e., vulnerability V_9), eventually due to attacks to the database (i.e., vulnerabilities V_7 and V_8).

TABLE V: Vulnerabilities - case study 2

Vulnerability	Description
V_1	Impersonate a legitimate source
V_2	Invalid data provided to users
V_3	Valid data discarded
V_4	The encryption key is weak
V_5	A link connected to the identified source can be sniffed
V_6	Source identity stolen
V_7	SQL injection
V_8	Invalid access to the database
V_9	Buffer overflow

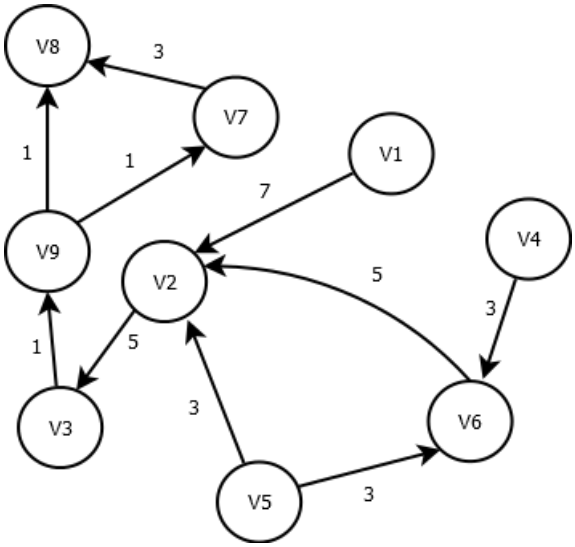

Fig. 5: Dependency graph - case study 2

Now, the exploitability values E_i associated to each identified vulnerability are shown in Table VI; while the initial weights set up for the dependencies are shown in Figure 5. The conversion between the qualitative and the quantitative evaluations is the same of the first case study. As for the previous case study, the values chosen for E_0 have been calculated considering the behavior of NOSs with respect to the data sources (i.e., the management of both registered and non-registered sources).

An important issue arises with the exploitability of V_1 and V_2, which are set to "easy", since non-registered sources may cause serious damages to NOSs system; with respect to such a behavior, no countermeasure is actually put in place by NOSs to prevent the block of the traffic as well as the provisioning of invalid data. The algorithms presented in Section IV-C only perform an evaluation of the received information, but malicious actions are no directly counterattacked by NOSs. This aspect reveals to be the main drawback of

NOSs middleware in its form, until now presented; in fact, until the work described in [8], NOS is not able to prevent or block lively attacks, such as DoS or DDoS, to the sources, and protect valid data from being lost or discarded from the network (i.e., exploitability of V_3 is set to "on average").

As regards registered sources, their identity may be stolen in case of weak encryption key or by sniffing their traffic; in such a situation, since different keys are used for different sources, exploitability of V_4 and V_5 is set to "difficult"; instead, the resulting exploitability of V_6 is set to "on average" because source identity stolen may lead to send invalid data to NOSs from the impersonating malicious entity.

With reference to attacks to the database, due to the fact that it is not exposed to external entities, but it is only internally accessed by NOS, then it is more resilient towards possible attacks. In fact, data are provided to users by means of a publish and subscribe system mediated by a broker, as explained in Section IV-D.

TABLE VI: Exploitability values - case study 2

	E_0	E_1	E_2
V_1	7	7	7
V_2	5	5	7
V_3	3	5	5
V_4	3	3	3
V_5	3	3	3
V_6	3	5	5
V_7	1	3	3
V_8	1	1	3
V_9	1	1	1

Hence, the results show that NOSs' system is not enough resilient towards the attack to network resources. The main risk regards the non-registered sources, because NOSs either have no control over their behavior or provide no proper mechanisms for counteracting traffic congestion and data counterfeiting.

C. Third Case Study

The third case study includes the same features of the second one, presented in Section V-B. However, in such a context, a new module, named REATO, is introduced into NOSs platform [29]. It represents a methodology for preventing, recognizing, and even blocking DoS attack's attempts, in order to protect the legitimate connected data sources as well as the platform itself. For further details about the functionalities and the behavior of REATO, please refer to [29].

The vulnerabilities for this case study are those listed in Listing 2 and in Table V. The dependency graph is not the same as for the second case study, because the initial weights set up for the dependencies are influenced by the presence of REATO. The dependency graph is reported in Figure 6).

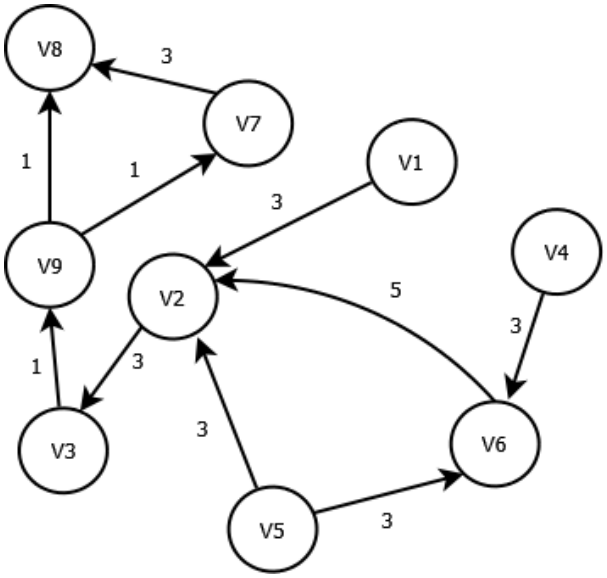

Fig. 6: Dependency graph - case study 3

What also vary are the exploitability values E_i, shown in Table VII. In such a scenario, the exploitability of V_1 and V_2 cannot be set to "easy", because, with REATO, non-registered sources are no longer a threat for NOSs system. Hence, the exploitability of V_1 and V_2 is set to "difficult", since NOSs are now able to prevent the block of the traffic. In this way, the discarding of valid data is partially prevented and mitigated (i.e., exploitability of V_3 is set to "difficult"). Instead, exploitability values of V_4-V_9 follow the same considerations presented in the second case study.

TABLE VII: Exploitability values - case study 3

	E_0	E_1	E_2
V_1	3	3	3
V_2	3	5	5
V_3	3	3	3
V_4	3	3	3
V_5	3	3	3
V_6	3	5	5
V_7	1	3	3
V_8	1	1	3
V_9	1	1	1

Hence, the results show that NOSs' system with

REATO is more resilient towards the attack to network resources. In fact, the exploitability values obtained for the third case study are lower than the ones obtained for the second case study, thus demonstrating a higher degree of trustworthiness for the analyzed IoT platform. Another important consequence is that the traffic block prevention and the depletion of NOSs' computational resources has a relevant impact on data availability. In fact, if the IoT system could be affected by DoS attacks, then we expect that the end-users will no longer be able to use the IoT services in real time.

D. Considerations

As emerged from the risk analysis presented above, on the one hand, NOSs middleware reveals to be robust towards improper access to data once they are received, processed, and shared by NOSs; this represents an important feature since we can state that NOSs are able to guarantee a high-level of security and privacy of information. On the other hand, serious issues arise with the data provision to NOSs, since non-registered sources or malicious internal ones may send invalid data up to blocking the network traffic. The various kinds of DoS attacks that may be carried out towards the platform as well as possible physical attacks to the IoT devices, if not counteracted, leave the IoT system affected by important threats. The presence of REATO partially cope with such threats by preventing and mitigating the actions of DoS attack's attempts. Moreover, a further solution should probably include an intrusion detection system, able to quickly recognize abnormal behaviors on the basis of well-defined features [30]. In fact, a similar approach could be adopted for recognizing possible misbehavior of the platform itself, thus allowing to assess the risk of the IoT system from a physical point of view.

It is worth to remark that a lot of work has already been done to secure NOSs and the managed data, as demonstrated by the module in charge of data security assessment (see Section IV-C), by the enforcement framework (see Section IV-B), by AUPS (see Section IV-D), and by REATO [29]. Such modules are considered to be trusty.

If we suppose to disable the enforcement framework along with AUPS functionalities, we obtain very different results from the risk analysis, as shown in Table VIII. With respect to Table IV, the exploitability values of many vulnerabilities are set to "on average" because, without the refreshing of the encryption keys performed by the KTM and the enforcement of access control on the published topics, the system becomes more easily exposed to information disclosure and identity stolen.

The graph of the dependencies is the same of the first case study, but the exploitability values have to be updated, as shown in Figure 7. Thus, the exploitability

TABLE VIII: Exploitability values - case study 1 modified

	E_0	E_1	E_2
V_1	5	5	5
V_2	5	5	5
V_3	5	5	7
V_4	3	3	3
V_5	5	7	7
V_6	5	7	7
V_7	5	5	5
V_8	5	5	7

calculated for E_2 presents, with respect to the original case study, higher values. Such an outcome means that AUPS and the enforcement framework are essential for preserving a high level of trustworthiness of NOSs middleware from a functional point of view.

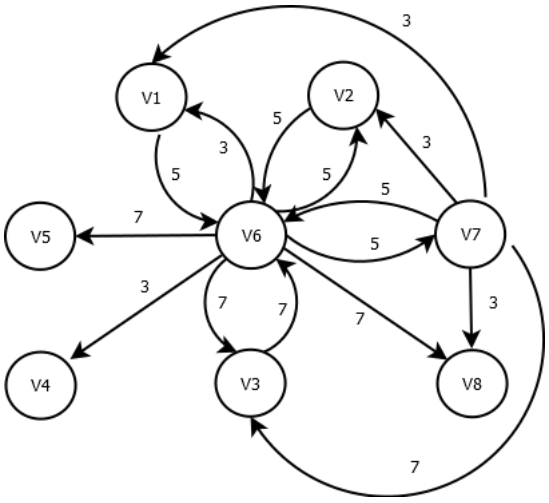

Fig. 7: Dependency graph - case study 1 modified

VI. Dynamic Assessment and Validation

The risk analysis discussed in Section V has helped to understand the potentialities and, at the same time, the actual weaknesses of NOSs middleware. However, such a kind of evaluation, in the present form, is static, in the sense that it does not vary over the time unless architectural of functionalities' changes are made to NOSs' components. For example, one or more NOSs belonging to the IoT network may be configured with some disabled modules (e.g., the enforcement framework, AUPS, the data security analysis). The advantage of disabling some functionalities is a decrease of delays and of computational overhead, if the application scenario allows it.

For such a reason it would be interesting to extend the static risk analysis just described with its integration with the outcomes provided by the algorithm for the security assessment, presented in Section IV-C. Such a mechanism, in fact, allows NOSs and, thus, users to be aware of the levels of security associated to the received data, in terms of confidentiality, integrity, privacy and authentication. Such information also allows to infer about the trustworthiness of the corresponding source; in fact, if the data provided by a certain producers are marked for a long time as unreliable, we may suppose that the originating source is also unreliable or under a malicious attack.

Bearing in mind such considerations, we have that the IoT system is composed by a network of NOSs, which, during the running time, handle and communicate with different sources located in the IoT environment. Such sources usually join and leave the IoT network and may also pass from one NOS to another.

In this scenario, a dynamic risk assessment should take into account further vulnerabilities, with respect to those discussed in Section V, which are:

- The number of connected sources with confidentiality score lower/greater than a determined threshold t_1
- The number of connected sources with integrity score lower/greater than a determined threshold t_2
- The number of connected sources with privacy score lower/greater than a determined threshold t_3
- The number of connected sources with authentication score lower/greater than a determined threshold t_4.

We remark that different thresholds (t_1, t_2, t_3, t_4) may be chosen for the four cases. Note that such parameters, named V_{sec}, varies over the time, depending on the amount of active sources and on the basis of their behavior (i.e., the levels of security calculated for their data by the algorithm in [2]). Since the security scores $score_{sec}$ are expressed (in the original version, presented in [2]) as decimal values in the range [0:1], then their exploitability values can be easily mapped to those established for the quantitative evaluation of vulnerabilities, presented in Table I, as follows:

$$\begin{gathered} \forall V_{sec} \in V : \\ E_{sec} = 10 - |score_{sec} * 10| \end{gathered} \tag{3}$$

Eligible values are those in the range [0:10] and the corresponding vulnerabilities should be included in the dependency graph of each considered risk scenario. In this way, not only "static" features (i.e., the enabling of certain security modules), but also dynamic parameters can be considered in the risk assesment during the system running.

A. Experiments

NOS platform has been developed, as introduced in Section IV, on a real prototype running on a Raspberry Pi. Data are obtained in real time from six sensors at a meteorological station installed in Trentino (Italy). In order to simulate the behavior of a network of NOSs, three instances are run and sources are supposed to be characterized by different security scores, as those calculated in [2]. The connections simulated among NOSs and sources are depicted in Figure 9.

Then, supposing that source 5 is the one that behaves in the worst way with respect to the considered security scores (i.e., sent data are recognized as violated, maybe intercepted by a non-authorized entity because the encryption algorithm/key is weak, as happens in case study 1 in Section V) and source 6 is a non-registered one, the exploitability values associated to the corresponding vulnerabilities will get higher and, thus, the risk associated to the use of the data managed by the connected NOS 3 will be greater than those evaluated for others NOSs, as shown in Table IX. Instead, sources connected to NOS 2 adopt stronger encryption algorithm/key; as a consequence, its exploitability values are lower than those of NOSs 1 and 3 (i.e., it is more difficult to exploit the corresponding vulnerabilities to carry out an attack).

If users or other data sources are aware of such information, they could choose to receive services or connect to more reliable NOSs at any time (e.g., NOSs 1 and 2 in this example). Hence, they are not only able to filter on the data which are received, but also to select the components of the IoT platform to interact with. Nevertheless, users/sources may also set, by means of a specific function provided by the IoT platform, a certain threshold t_{max} on the final exploitability calculated by the risk analysis and decide to interact only with NOSs owning a pre-defined degree of trustworthiness. Such a capability is particularly fundamental in case one or more NOSs are compromised by a network attack or by a failure; in these situations, they can be promptly isolated from the IoT system if the associated risk falls below a certain limit, defined by a proper threshold t_{min}. Figure 8 shows the corresponding dashboard provided to data consumers, which allows them to select the minimum and the maximum threshold values and, thus, see what is the selected NOS, along with its corresponding risk values, to interact with at any time.

Finally, we provide a sketch about the impact on NOS middleware performance of the implementation of the risk analysis mechanism just presented. Hence, we decided to run the system for a period of 1 hour; in such a period of time, NOS fetches the data at two different rate: 10 packets per second and 20 packets per second. This frequency obviously affects the delays as well as the computational effort. We consider the six sources

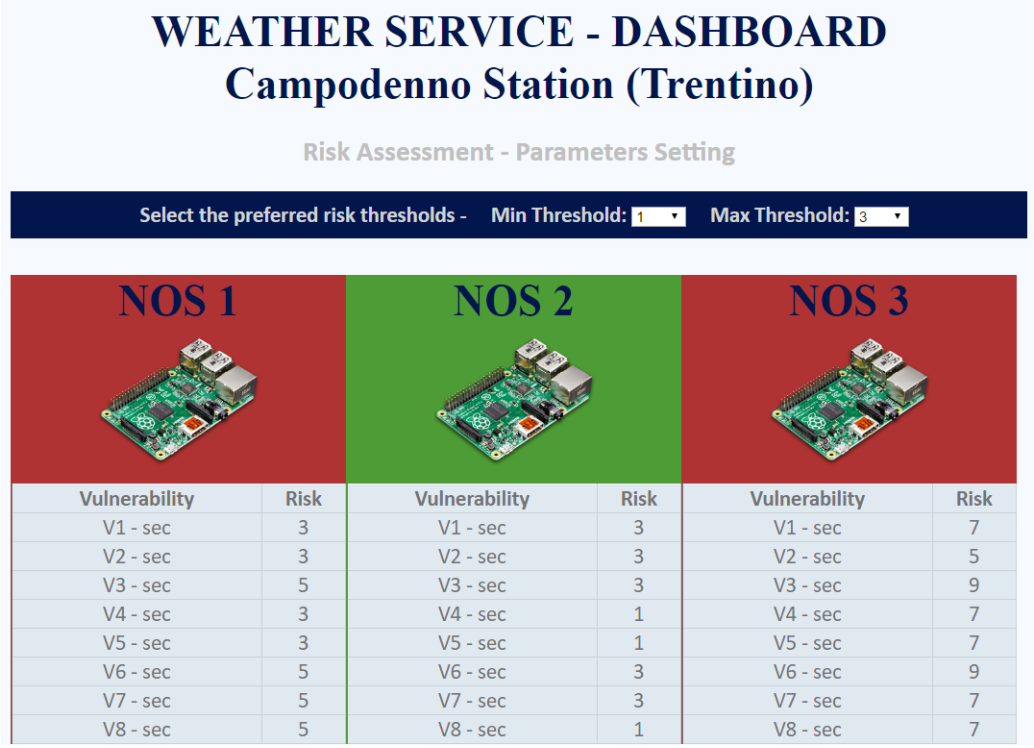

Fig. 8: Risk assessment dashboard

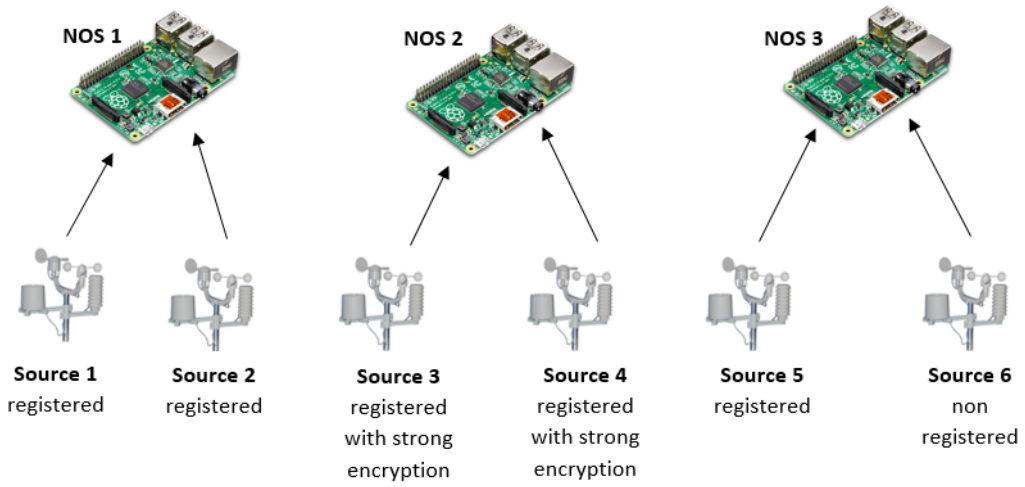

Fig. 9: Connections simulated among NOSs and souces

presented above and the presence of three users who interact with NOSs' dashboard every 5 minutes.

Concerning the delay, it is calculated in a end-to-end manner, and it is computed as the distribution of the elapsed time from the data reception to NOS until it is sent to the broker. Latency is evaluated for two different settings: with and without running the risk assessment module. Note that two kinds of delay are introduced by the risk assessment methodology: (i) on the NOS side, the periodical calculation of the risk's exploitability values; (ii) on the user' side, the filtering operation on the dashboard, which may change the NOSs authorized to send data to the specific user. The former strictly depends on the frequency of weights' update associated to the security vulnerabilities (see Section VI), since the enabling/disabling of the NOSs security modules is supposed to be less frequent. The latter depends on the behavior/needs of the user.

TABLE IX: Exploitability values - case study 1

	E_{NOS1}	E_{NOS2}	E_{NOS3}
V_{1-sec}	3	3	7
V_{2-sec}	3	3	5
V_{3-sec}	5	3	9
V_{4-sec}	3	1	7
V_{5-sec}	3	1	7
V_{6-sec}	5	3	9
V_{7-sec}	5	3	7
V_{8-sec}	5	1	7

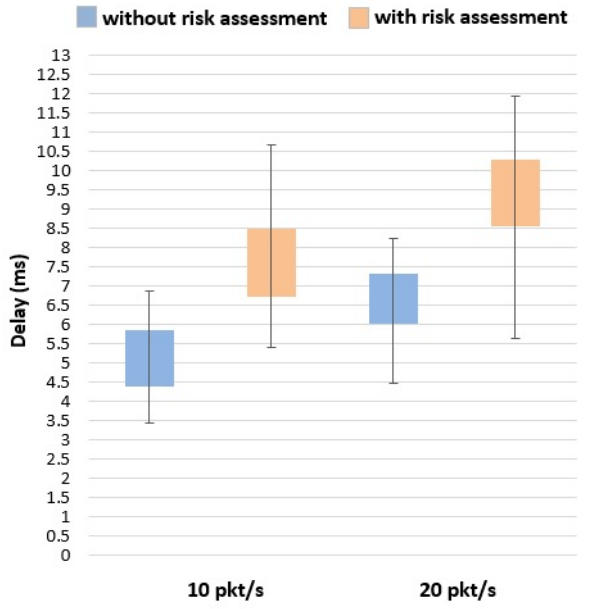

Fig. 10: Delay: whiskers-box diagram for the NOSs system with and without risk assessment, and with two different data fetch rate (10 and 20 pkt/s).

Figure 10 shows that the risk assessment module does not affect in a relevant way the delay of data processing and transmission. The same conclusions can be drawn for the CPU load, as shown in Figure 11

VII. Conclusions

In order to assess the vulnerabilities, or the degree of robustness, of a system towards possible internal and external attacks, it is important to carry out proper countermeasures and/or add/modify specific security modules. In the IoT scenario, platforms store, transmit, and share data provided by the users that exploit the offered services; such data may also be sensitive/private according to their content. As a consequence, users have to trust these systems and the way in which their information is treated, or they do not use them at all. Hence, the paper

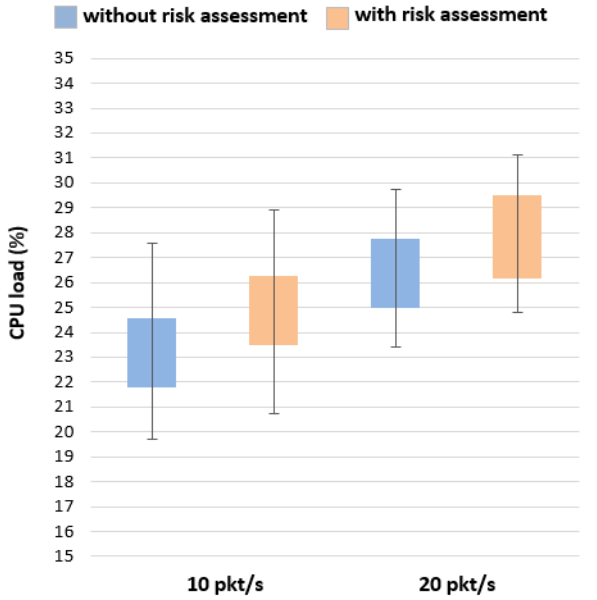

Fig. 11: CPU load: whiskers-box diagram for the NOSs system with and without risk assessment, and with two different data fetch rate (10 and 20 pkt/s).

has presented a risk analysis methodology applicable to IoT infrastructures, which takes into account both static and dynamic features/components of the system itself. More in detail, we considered measurement as the process by which numbers are assigned to attributes of entities (in our case, to the exploitability of a vulnerability). Hence, what matters is the structure of the metric rather than its *absolute* value. Such a kind of evaluation is fundamental for assessing the reliability of the entities that compose an heterogeneous IoT system. Potential attacks and vulnerabilities has been pointed out considering, as an illustrative example, an existing IoT middleware, thus putting in light its potentialities and weaknesses. Finally, the feasibility of the proposed solution has been validated by means of a real prototype implementation of such a middleware. Note that it also provides a dashboard accessible to interested users that illustrates the outcomes of the risk analysis during the time. As a future work, we would apply our approach within other existing IoT platforms, in order to test effectiveness of our methodology in different contexts/environments. Moreover, it would be very interesting to make a comparison of the IoT systems with high acceptance (i.e., openHab[12], Thread[13], AllJoyn[14],

[12]https://www.openhab.org
[13]https://threadgroup.org
[14]https://allseenalliance.org/framework

Amazon Web Service (AWS) IoT[15], IBM Watson IoT[16], FiWare[17]) with respect to our proposed risk assessment methodology (now only deployed in NOS middleware), in order to reveal the differences among them in managing security features and in guaranteeing proper level of trustworthiness to the users.

REFERENCES

[1] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of Things: Vision, applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, 2012.

[2] S.Sicari, A. Rizzardi, D. Miorandi, C. Cappiello, and A. Coen-Porisini, "A secure and quality-aware prototypical architecture for the Internet of Things," *Information Systems*, vol. 58, pp. 43–55, 2016.

[3] A. Rizzardi, D. Miorandi, S. Sicari, C. Cappiello, and A. Coen-Porisini, "Networked smart objects: Moving data processing closer to the source," in *2nd EAI International Conference on IoT as a Service*, Oct 2015.

[4] G. Dini and L. Lopriore, "Key propagation in wireless sensor networks," *Computers & Electrical Engineering*, vol. 41, pp. 426–433, 2015.

[5] R. D. Pietro, L. Mancini, and S. Jajodia, "Providing secrecy in key management protocols for large wireless sensors networks," *Ad Hoc Networks*, vol. 1, no. 4, pp. 455–468, 2003.

[6] S. Sicari, A. Rizzardi, D. Miorandi, and A. Coen-Porisini, "Internet of Things: Security in the keys," in *12th ACM International Symposium on QoS and Security for Wireless and Mobile Networks*, Malta, Nov 2016, pp. 129–133.

[7] S. Sicari, A. Rizzardi, D. Miorandi, C.Cappiello, and A. Coen-Porisini, "Security policy enforcement for networked smart objects," *Computer Networks*, vol. 108, pp. 133–147, 2016.

[8] A. Rizzardi, S. Sicari, D. Miorandi, and A. Coen-Porisini, "Aups: An open source authenticated publish/subscribe system for the Internet of Things," *Information Systems*, vol. 62, pp. 29–41, 2016.

[9] 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.

[10] T. Kirkham, D. Armstrong, K. Djemame, and M. Jiang, "Risk driven smart home resource management using cloud services," *Future Generation Computer Systems*, vol. 38, pp. 13–22, 2014.

[11] M. Teimourikia and M. Fugini, "Ontology development for runtime safety management methodology in smart work environments using ambient knowledge," *Future Generation Computer Systems*, vol. 68, pp. 428–441, 2017.

[12] A. Furfaro, L. Argento, A. Parise, and A. Piccolo, "Using virtual environments for the assessment of cybersecurity issues in iot scenarios," *Simulation Modelling Practice and Theory*, 2016.

[13] A. Jacobsson, M. Boldt, and B. Carlsson, "A risk analysis of a smart home automation system," *Future Generation Computer Systems*, vol. 56, pp. 719–733, 2016.

[14] T. R. Peltier, *Information security risk analysis*. CRC press, 2005.

[15] M. Benini and S. Sicari, "Assessing the risk of intercepting voip calls," *Computer Networks*, vol. 52, no. 12, pp. 2432–2446, 2008.

[16] ——, "Risk assessment in practice: A real case study," *Computer communications*, vol. 31, no. 15, pp. 3691–3699, 2008.

[17] A. Shameli-Sendi, R. Aghababaei-Barzegar, and M. Cheriet, "Taxonomy of information security risk assessment (isra)," *Computers & Security*, vol. 57, pp. 14–30, 2016.

[18] P. Shamala, R. Ahmad, and M. Yusoff, "A conceptual framework of info structure for information security risk assessment (isra)," *Journal of Information Security and Applications*, vol. 18, no. 1, pp. 45–52, 2013.

[19] J. Meszaros and A. Buchalcevova, "Introducing ossf: A framework for online service cybersecurity risk management," *Computers & Security*, vol. 65, pp. 300–313, 2017.

[20] M. Benini and S. Sicari, "A mathematical framework for risk assessment," in *New Technologies, Mobility and Security*. Springer, 2007, pp. 459–469.

[21] D. Balzarotti, M. Monga, and S. Sicari, "Assessing the risk of using vulnerable components," in *Quality of Protection*. Springer, 2006, pp. 65–77.

[22] "IBM and eurotech, "mqtt v3.1 protocol specification"," http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html, 2017.

[23] "Node.JS," http://nodejs.org/, 2017.

[24] "MongoDB," http://www.mongodb.org/, 2017.

[25] "Mosquitto, "an open source mqtt v3.1/v3.1.1 broker"," http://mosquitto.org, 2017.

[26] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *13th ACM Conference on Computer and Communications Security*, 2006, pp. 89–98.

[27] T. Roosta, S. Shieh, and S. Sastry, "Taxonomy of security attacks in sensor networks and countermeasures," in *The first IEEE international conference on system integration and reliability improvements*, vol. 25, 2006, p. 94.

[28] G. Tesauro, *Practical issues in temporal difference learning*. Springer, 1992.

[29] S. Sicari, A. Rizzardi, D. Miorandi, and A. Coen-Porisini, "Reato: Reacting to denial of service attacks in the internet of things," *Computer Networks*, vol. 137, pp. 37–48, 2018.

[30] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Gone: Dealing with node behavior," in *Consumer Electronics-Berlin (ICCE-Berlin), 2015 IEEE 5th International Conference on*, 2015, pp. 358–362.

[15] https://www.amazonaws.cn/en/iot-platform

[16] https://www.ibm.com/internet-of-things

[17] https://www.fiware.org