Apparatus: Reasoning About Security Requirements in the Internet of Things

Orestis Mavropoulos[1], Haralambos Mouratidis[1], Andrew Fish[1], Emmanouil Panaousis[1], and Christos Kalloniatis[1,2]

[1] School of Computing, Engineering and Mathematics
University of Brighton, Brighton, UK
{o.mavropoulos,h.mouratidis,andrew.fish,e.panaousis}@brighton.ac.uk
[2] Department of Cultural Technology and Communication
University of the Aegean, Lesvos, Greece
chkallon@aegean.gr

Abstract. Internet of Things (IoT) can be seen as the main driver towards an era of ubiquitous computing. Taking into account the scale of IoT, the number of security issues that emerge are unprecedented, therefore the need for proposing new methodologies for elaborating about security in IoT systems is undoubtedly crucial and this is recognised by both academia and the industry alike. In this work we present Apparatus, a conceptual model for reasoning about security in IoT systems through the lens of Security Requirements Engineering. Apparatus is architecture-oriented and describes an IoT system as a cluster of nodes that share network connections. The information of the system is documented in a textual manner, using Javascript Notation Object (JSON) format, in order to elicit security requirements. To demonstrate its usage the security requirements of a temperature monitor system are identified and a first application of Apparatus is exhibited.

Keywords: Internet of Things, Security Requirements Engineering, IoT Conceptual Model, Information Security [3]

1 Introduction

One of the areas that attract attention from the research and industry worlds is Internet of Things (IoT). Weiser in 1991 [1], provides one of the most accurate yet simple vision of IoT by stating that the most profound technologies merge with the environment. Technology will be so evident that we will start perceiving it as a natural part of life. Indeed IoT along with cloud computing can turn the last statement into reality.

Despite IoT popularity a number of security challenges faced in these environments have already been revealed [2–5]. A prominent security concern, found in the surveys, are Denial of Service (DoS) attacks in embedded devices [6]. Such devices lack the necessary resources to withstand repeated requests from malicious

[3] The original publication is available at www.link.springer.com.

attackers. Another commonly identified issue are Man-In-The-Middle (MITM) attacks [7], that take advantage of either weak encryption algorithms of embedded devices or weak authentication mechanisms among the systems [8]. Security specialists argue that the most effective way of ensuring security in systems is to incorporate security focused practices in the development cycle. As a practice, it ensures that the product will meet specific security standards, which in turn will ensure its robustness when it is actively deployed in real life scenarios. The practice of including secure practices in the development cycle is also advocated by the field of requirements engineering. Requirements engineering is applied along with the stakeholders early in the development cycle to identify security requirements [9]. To this end, our main concern in this paper is to propose a novel conceptual model that can be used by software designers in order to extract respective security requirements based on the system's architectural information, in addition to stakeholder needs.

To tackle the issue of requirements elicitation in IoT systems we propose Apparatus. The speculation is, that if an IoT system is analysed in an abstract manner, the technical specifications of each scenario should not be relevant in security analysis. Moreover their core security requirements should be universal to any IoT system. Apparatus is a conceptual model to enable reasoning about security in IoT using information from the architecture of the system. From that information, security requirements can be elicited. Security analysis in the architectural level offers both advantages and limitations. The architecture of a system offers information valuable for security analysis, such as types of network connections between node or their role in the network. On the other hand, certain aspects of the system are not expressed such as users or malicious attackers. To better identify the limitations, the concept of a "microworld" is introduced, where the security analysis is being conducted in a managed environment. To mitigate the limitations identified in the "microworld", Apparatus will be integrated to other security requirements engineering methods. The reasoning behind adopting Apparatus as a core model is that it can be integrated to other security frameworks. Therefore there is no need to introduce a new framework to the already many existing security requirements frameworks (e.g., [10–14]) that have been in active development for a number of years and offer a comprehensive and robust security analysis. Their expertise will be better utilized if Apparatus acts as a "bridge" to other security requirements methods for IoT rather than it being developed as separate entity.

The paper is structured as follows: Section 2 describes the related work in the fields of IoT and Security Requirements Engineering. Section 3 presents the conceptual model of Apparatus. Section 4 shows a security analysis of an IoT system using the Apparatus security reasoning. Section 5 discusses future extensions of Apparatus along with its limitations and also concludes this paper.

2 Related work

Security requirements engineering has been used in a variety of systems and fields in order to analyse security during the development cycle. Although there are many surveys on identifying security issues in IoT [2–5], few attempts have been made to address them from a security requirements engineering point of view.

An attempt to provide a framework for security and privacy in IoT systems using requirements engineering was made by Alqassem [15]. He identifies the complexity of analysing security in IoT systems and he states that the key components in IoT are only two: RFID systems and sensor networks. To reason about security in IoT they propose the use of the $i*$ framework in order to undertake security analysis in future case studies. The paper does not consider other technologies and topologies that are being used in IoT. IoT is not restricted only to RFID systems, but uses any communication technology, such as Wi-Fi, NFC or Bluetooth. Moreover the architectural topologies are not restricted to networks solely comprised by sensors, but include any type of device.

IoT as a whole is composed by a multitude of devices. A large number of those devices are embedded devices. An informative paper from Gürgens describes a vision in applying security engineering to embedded systems [16]. He identifies a number of security challenges of embedded systems, that should be addressed in order to have a secure system. He reasons that specific security requirements tools should be designed, tailored to the needs of embedded systems. Another framework aiming to provide security in embedded IoT systems is proposed by Babar [17]. In his paper he classifies the types of attacks aimed at IoT systems. He proposes a basic three step security framework to elicit requirements in embedded systems. To accomplish that he identifies the building blocks of embedded systems in IoT. Tian designs a security framework specific to wireless sensor networks [18]. The framework proposes a system architecture, that is broken down into eight modules, with each module having specific functionality to mitigate security issues. To summarize, the presented works do not view IoT in a comprehensive manner, since they only aim to mitigate security issues in specific areas. Therefore, they cannot be used to offer a universal security analysis to any IoT scenario, but only aim to address specific instances of IoT systems.

Díaz identifies a number of issues and open challenges with the integration of IoT and Cloud computing [19]. Many functionalities of Internet of Things are only possible through the cloud infrastructure. Some scenarios may require the use of sensors as a service, while others may use cloud services for processing functionality. He states that IoT will function as the middle-ware that will transmit all its data to the cloud for processing. The paper shows that the current trend for IoT application development is based on Cloud computing.

Although outside the scope of requirements engineering, a framework for modeling and assessing security in IoT system is proposed by Ge [13]. The framework uses a graphical security model that evaluates the level of security using security metrics. The framework assesses security of IoT systems in a compre-

hensive manner and is not limited to specific IoT scenarios, such as embedded systems or RFID systems. A similar universal comprehensive approach in security of IoT is used in Apparatus, in order to reason about security requirements in any IoT system, but from a requirements engineering point of view.

3 Presentation of Apparatus reasoning

Apparatus is a conceptual model for reasoning about security in IoT systems. In the context of Apparatus, security requirements are defined as *a restriction related to security issues, such as privacy, integrity and availability, which can influence the analysis and design of a multiagent system under development by restricting some alternative design solutions, by conflicting with some of the requirements of the system, or by refining some of the system' s objectives*, an approach used by Secure Tropos concept of *security constrain* [12]. A similar definition of security requirements is given by Haley [14]. He defines them as *constraints on the system's functional requirements, rather than being themselves functional requirements.* The current version of Apparatus focuses on the proposed conceptual model, as shown in Fig. 1, through which designers can be assisted on capturing the necessary knowledge from the IoT system's architecture perspective in order to identify and extract security requirements. In order to achieve that, the information of the IoT system has to be presented in a specific format that will make the extraction of security requirements straightforward. The model does not aim to provide a detailed security analysis in any given IoT scenario. It provides the necessary information, for an IoT system to be secure in a "microworld", where users, malicious attackers and other communication with the world outside of the system are not part of the requirements elicitation. In order to provide a more detailed security analysis, Apparatus will be integrated with other security requirements methods, such as Secure Tropos [11] and SQUARE [10]. Security analysis in IoT systems can be performed in a two step approach. Apparatus will be used as the first step in the security analysis, where its security requirements are derived from the IoT architecture. The second step will be the usage of one of the aforementioned methods so as to facilitate a more complete security analysis of the system. The integration of Apparatus to other methods will be part of future work and is out of the scope of this paper. The proposed conceptual model of Apparatus is presented in Fig. 1 using an Entity-Relationship Diagram with crow's foot notation.

Apparatus is an architecture-oriented model, where an IoT system is described as a cluster of IoT nodes connected to each other using network connections. Each IoT node has a set of properties that describe its functionality in the system. The properties include information such as the type of the device and its role within the system. There are two main concepts in Apparatus: the *IoT node* and the *network connection*.

Each of the main concepts has a set of properties that further describes the system. The properties of the *IoT node* are: (1) **identification**: gives an

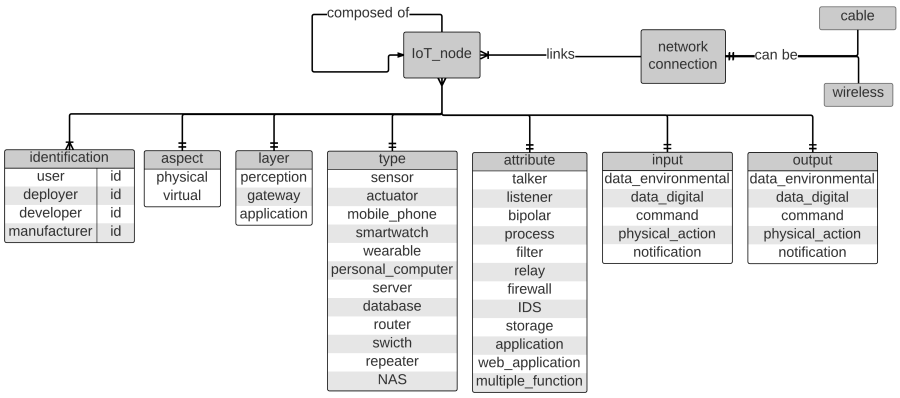

Fig. 1. Conceptual model of Apparatus in Entity-Relationship Diagram

id to each of the stakeholders of the node. The stakeholders are four, *user, deployer, developer, manufacturer*; (2) **aspect**: declares whether the IoT node is a single node, or composed of sub nodes. (3) **layer**: the layer of the IoT architecture to which the node belongs; (4) **type**: what kind of device the node is; (5) **attribute**: the type of role or operation that the node performs for the network; (6) **input**: what is required in order for the node to perform its role or operation; (7) **output**: the result of the node's operation or role. The IoT nodes are connected to each other using *network connections*. The type of connections can be two: (1) **Wireless**: signifying a connection using a wireless protocol and (2) **Cable**: signifying a connection using a wired medium.

Each node property can only hold a single value. If an IoT node that has more than one functionality, needs to be defined, it can be broken down into the same number of sub nodes as the functionalities we want to express. For example a laptop that acts both as processing server and as the system's database, will be composed of two sub nodes. One sub node that describes the server functionality and one sub node that describes the database functionality. These two sub nodes in return make the laptop IoT node. Identical nodes in an IoT system, such sensors and actuators can be grouped together and presented as a single node.

The properties of the IoT node are explained in detail: **Identification:** There are four types of stakeholders in an IoT network, *user* (user of the node), *deployer* (installer of the node), *developer* (producer of the software of the node), *manufacturer* (hardware producer of the node). In Apparatus each of the stakeholders that is involved in an IoT system is assigned a unique ID. That ID is used to identify a stakeholder's role in an IoT node and takes an integer value.

Aspect: The property *aspect* can take one of two values. The node can either be a physical node, that declares the IoT node as a single node in the system, or as a virtual node, meaning that the node belongs to a set of virtual nodes that in turn compose a single physical node.

Layers: APPARATUS uses a three-layer architecture that consists of the Application Layer, Network Layer and the Perception Layer [20, 21]. Other architectures provide more levels of abstractions. For example a SOA based approach identifies five layers, *application, service composition, service management, object abstraction, objects* [2]. Another approach identifies more layers, that are *application, middleware, coordination, backbone network, existed alone network, access layer, edge technology* [22]. The proposed architectures for Internet of Things have yet to fuse into a reference model [23], for that reason we chose the three-layer approach. It provides the necessary properties for reasoning about security, while allowing to be extended if more levels of abstraction are introduced into the reference model of IoT.

Type: Defines the device the IoT node represents. For example the type of an IoT node could be a sensor, an actuator or a mobile phone as shown in Fig. 1.

Attribute: Describes the functionality of a node in the system. Examples of a node's functionality are shown in Fig. 1. Some attributes are better suited to specific types of nodes. For example a *talker* node is a type of sensor, that only sends information to the network and does not perform any actions. A *listener* node is a type of actuator, that only performs actions and does not send any information from the environment. A *bipolar* is a node that is both a talker and a listener. When an IoT node is composed of virtual nodes, the attribute of their physical node is *multiple functions*.

Input & Output: Have the same set of values. As mentioned before the *input* signifies the required data for the node in order to produce the desired *output*. The value could be a loose term such as data, or a notification as shown presented in Fig. 1.

Network connection: The construct *network connection* represents the type of the network link between nodes. The link can either be wireless, such as via Wi-Fi, Bluetooth and RFID, or using a cable, such as an Ethernet and a USB. From a high level security point of view, there is little difference if a technology such as Wi-Fi over Bluetooth is used. Since they both are wireless mediums, they share the same security requirements. The same applies to cable mediums. From a low level perspective what technology is used, has a pivotal role, since each one has specific vulnerabilities. That level of reasoning is not present in the current version of APPARATUS.

APPARATUS currently represents information in a textual format, since it will be used by machines for automated security analysis. An IoT system that has been described using APPARATUS, is formatted as a Javascript Notation Object (JSON) file. JSON format is an Open Standard used to transmit data objects consisting of attribute–value pairs. The "attribute" is immutable and corresponds to the names of the IoT node's properties. The "value" is mutable and corresponds to the values of the IoT node's properties. APPARATUS uses that formatting style in order to correlate the information of the IoT system with the necessary security requirements. Using JSON format the process of requirements elicitation can be automated, thus making the analysis of large IoT networks more efficient. That type of formatting is useful for establishing

```
{
  "IoT_system": [{
    "IoT_node_01": [{
      "identification": {
        "user": "",
        "deployer": "",
        "developer": "",
        "manufacturer": ""
      },
      "aspect": "",
      "type": "",
      "attribute": "",
      "input": "",
      "output": ""
    }, {
      "connection": {
        "connects_to": "node_name",
        "connection_type": ""
      }
    }]
  }]
}
```


Fig. 2. Skeleton JSON template

a more formal rule-based approach for correlating security requirements with properties of the IoT system. Using the textual notation of a JSON file, a visual notation will be incorporated into the next versions of Apparatus in order to offer a human friendly approach.The skeleton template in JSON format is presented in Fig. 2.

4 Example of security reasoning

In this section an application of the Apparatus reasoning is presented using an IoT system implementation of a temperature monitoring application, that is hosted in a private in house network. The components of the network are the following: a (1) temperature sensor, a (2) router, a (3) server, that functions as a database and a data formatter for the database, and a (4) laptop. The temperature sensor gathers environmental data and sends them to the server through the router. The server formats the data and stores them in its database. The laptop then requests the information from the database. In Fig. 3 the layout of the system is shown.

Each of the devices, constitutes an IoT node in Apparatus. For the sake of brevity the stakeholders of the system are two. The same "person" has installed and makes use of the devices. He acts as the *user* as well as the *deployer* and has been assigned the id: 01. Another "person" has produced all the devices along with their software, so he acts as both the *manufacturer* and the *developer* of the system with the id: 02. In Fig. 4, the temperature monitoring system is expressed in JSON format using Apparatus reasoning.

Security analysis: The security analysis is being conducted using the information presented in Fig. 4. During this part of the analysis the IoT system in part of a "microworld". As explained before, the "microworld" acts as partial view of the actual security dangers of the real life world.

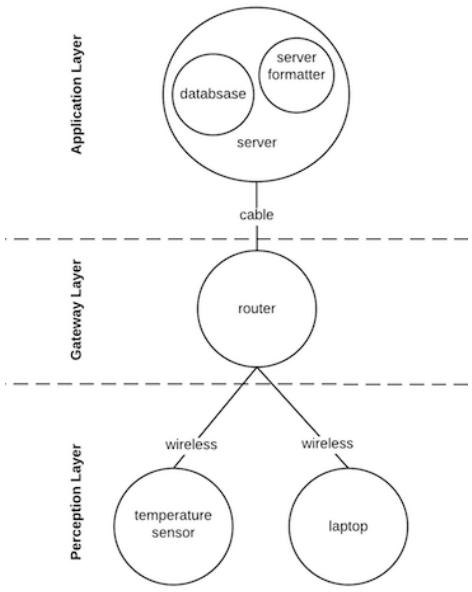

Fig. 3. Temperature monitoring application layout

The IoT nodes *sensor* and *laptop* are parts of the perception layer, as seen in lines 11 and 112 in Fig. 4. Devices in the perception layer allow physical access to their users. A security requirement resulting from that information, is that the nodes *sensor* and *laptop* need to be physically secure. Further analysis shows that both nodes communicate using wireless mediums (lines 19 and 119 in Fig. 4). Wireless mediums have limited range. Moreover all the nodes share the same user and developer. It can be deduced that all the nodes should be in close proximity to each other. Meaning that if an attacker has physical access to one, he probably has access to the other nodes of the system. The security requirement should be updated to include that all the nodes of the system need to be physically secure.

The IoT system uses wireless mediums (lines 19, 119 in Fig. 4). A requirement of wireless transmission is that access should only be allowed to authorised users and devices. All the nodes share the same manufacturer and developer (lines 7, 8, 28, 29, 48, 49, 68, 69, 88, 89, 108, 109 in Fig. 4). It can be assumed that some security mechanisms to prevent unauthorised access are in place. Those mechanisms should be taken into account when implementing the system.

The sensor node takes environmental data as an input (line 14 in Fig. 4). Environmental data are not controlled by the system, and a level of integrity must be ensured, in order to prevent Denial Of Service attacks or any other kind of tampering with the network. The security requirement is that environmental data cannot tamper with the system if they deviate from their expected behaviour.

```
{
  "IoT_system": [{
      "sensor": [{
        "identification": {
          "user": "01",
          "deployer": "01",
          "developer": "02",
          "manufacturer": "02"
        },
        "aspect": "physical",
        "layer": "perception",
        "type": "sensor",
        "attribute": "talker",
        "input": "data_environmental",
        "output": "data_digital"
      }, {
        "connection": {
          "connects_to": "router",
          "connection_type": "wireless"
        }
      }]
    },
    {
      "router": [{
        "identification": {
          "user": "01",
          "deployer": "01",
          "developer": "02",
          "manufacturer": "02"
        },
        "aspect": "physical",
        "layer": "gateway",
        "type": "sensor",
        "attribute": "relay",
        "input": "data_digital",
        "output": "data_digital"
      }, {
        "connection": {
          "connects_to": "server",
          "connection_type": "cable"
        }
      }]
    }, {
      "server": [{
        "identification": {
          "user": "01",
          "deployer": "01",
          "developer": "02",
          "manufacturer": "02"
        },
        "aspect": "physical",
        "layer": "application",
        "type": "server",
        "attribute": "multiple_function",
        "input": "data_digital",
        "output": "data_digital"
      }, {
        "connection": {
          "connects_to": "router",
          "connection_type": "cable"
        }
      }]
    }, {
      "database": [{
        "identification": {
          "user": "01",
          "deployer": "01",
          "developer": "02",
          "manufacturer": "02"
        },
        "aspect": "virtual",
        "layer": "application",
        "type": "database",
        "attribute": "storage",
        "input": "data_digital",
        "output": "data_digital"
      }, {
        "connection": {
          "connects_to": "router",
          "connection_type": "cable"
        }
      }]
    }, {
      "server_formatter": [{
        "identification": {
          "user": "01",
          "deployer": "01",
          "developer": "02",
          "manufacturer": "02"
        },
        "aspect": "virtual",
        "layer": "application",
        "type": "server",
        "attribute": "process",
        "input": "data_digital",
        "output": "data_digital"
      }, {
        "connection": {
          "connects_to": "router",
          "connection_type": "cable"
        }
      }]
    }, {
      "laptop": [{
        "identification": {
          "user": "01",
          "deployer": "01",
          "developer": "02",
          "manufacturer": "02"
        },
        "aspect": "physical",
        "layer": "perception",
        "type": "laptop",
        "attribute": "listener",
        "input": "data_digital",
        "output": "notification"}, {
        "connection": {
          "connects_to": "router",
          "connection_type": "wireless"
        }
      }]
    },
  ]
}
```


Fig. 4. Temperature monitoring system JSON format

One of the server's functions is to act as database (line 73 in Fig. 4). Databases take data as an input and store it a specific format.Two security requirements are elicited from that information. Firstly, database input must be subject to sanitation, to prevent SQL injection attempts or similar attacks. Secondly, the contents of the database should only be modified by authorised users and devices of the system.

The IoT nodes compose a network of devices. Chronological records of any activity that affects operations, procedures or events of the system must be kept. That functionality is a default security requirement of any IoT systems in Apparatus.

The process of security requirements elicitation that was described above can be automated, by correlating security requirements with values in the IoT system. The identified security requirements compose a list, with each security requirement being a separate entry that correlates with IoT values shown in Fig. 1. An example of the automated security requirements elicitation specific to the temperature monitoring application is shown in Table. 1.

Table 1. Security requirements elicited from IoT system properties

Security Requirements	**IoT system properties (lines)**
Nodes should be physically protected	layer: perception (11, 112) connection: wireless (19, 119) user/deployer id:01 (all nodes) developer/manufacturer id:02 (all nodes)
System can only be used by authorised users and devices	connection: wireless (19, 119)
Environmental data should not tamper with the system	input: data_environmental (14)
Database data should only be modified by authorised users or devices	type: database (74)
Input of the database should subject to sanitation	type: database (74)
Nodes must keep chronological records of any activity that affects operations, procedures or events of the system	Default security requirement of IoT systems

5 Conclusion

This paper, starts by illustrating the importance of proposing a novel model to facilitate security analysis and reasoning for IoT. Security analysis should be use a generic approach while being able to be used along with specialized security methods when needed by the system. In that spirit, the proposed model

should offer its capabilities to any IoT scenario without losing its properties, while offering enough flexibility for further extensions. In order to investigate security issues with the use of security requirements engineering, APPARATUS was presented. APPARATUS enables reasoning about security in IoT systems from a system architecture point of view. An IoT system is expressed as a cluster of IoT nodes, with each node having a set properties, that define its role within the system. The information of the system is conveyed in a textual manner, specifically using JSON format. Using APPARATUS, the security analysis of a temperature monitoring system was undertaken and described in this paper. Despite being novel and coherent, the current version of APPARATUS has a number of limitations. An IoT system is analysed from an architectural view and therefore cannot offer a comprehensive security analysis. To illustrate the limitations, the concept of "microworld" was introduced. The "microworld" facilitates security analysis by emulating a managed environment. Another limitation is the inability to express certain environments that have unknown variables, such as the Cloud or health applications. For example, in a Cloud system the internal architecture of the Cloud platform or its security configurations are not publicly known and as such cannot be expressed using APPARATUS.

In order to mitigate those limitations, our future work with APPARATUS aims to be able to adopt elements of other security requirements engineering methods by operating in an modular manner based on the different applications that are investigated. Furthermore, APPARATUS elicitation process will be automated by correlating security requirements with information extracted from the IoT system.

References

1. M. Weiser, "The computer for the 21st century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.
2. L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
3. A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.
4. J. Granjal, E. Monteiro, and J. S. 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.
5. Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the internet of things: Perspectives and challenges," *Wireless Networks*, vol. 20, no. 8, pp. 2481–2501, 2014.
6. H. Suo, J. Wan, C. Zou, and J. Liu, "Security in the internet of things: A review," in *2012 International Conference on Computer Science and Electronics Engineering*, (Hangzhou), pp. 648 – 651, Institute of Electrical & Electronics Engineers (IEEE), 2012.
7. J. Du and S. Chao, "A study of information security for m2m of iot," *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*, 2010.

8. J. Liu, Y. Xiao, and C. P. Chen, "Authentication and access control in the internet of things," in *2012 32nd International Conference on Distributed Computing Systems Workshops*, (Macau), pp. 588–592, Institute of Electrical & Electronics Engineers (IEEE), 2012.
9. E. S. Coles, "Analyzing and specifying security requirements in early stages of software development life cycle," *Journal of Mobile, Embedded and Distributed Systems*, vol. 7, no. 2, pp. 87–94, 2015.
10. N. R. Mead and T. Stehney, "Security quality requirements engineering (square) methodology," *ACM SIGSOFT Software Engineering Notes*, vol. 30, no. 4, pp. 1–7, 2005.
11. P. Giorgini and H. Mouratidis, "Secure tropos: A security-oriented extension of the tropos methodology," *International Journal of Software Engineering and Knowledge Engineering*, vol. 17, no. 02, pp. 285–309, 2011.
12. H. Mouratidis and P. Giorgini, "Secure tropos: A security-oriented extension of the tropos methodology," *International Journal of Software Engineering and Knowledge Engineering*, vol. 17, pp. 285–309, 04 2007.
13. M. Ge and D. S. Kim, "A framework for modeling and assessing security of the internet of things," in *2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS)*, (Melbourne, VIC), pp. 776 – 781, Institute of Electrical & Electronics Engineers (IEEE), 2015.
14. C. Haley, R. Laney, J. Moffett, and B. Nuseibeh, "Security requirements engineering: A framework for representation and analysis," *IEEE Transactions on Software Engineering*, vol. 34, no. 1, pp. 133–153, 2008.
15. I. Alqassem, "Privacy and security requirements framework for the internet of things (iot)," *Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014*, pp. 739–741, 2014.
16. S. Gürgens, C. Rudolph, A. Maña, and S. Nadjm-Tehrani, "Security engineering for embedded systems," *Proceedings of the International Workshop on Security and Dependability for Resource Constrained Embedded Systems - S&D4RCES '10*, 2010.
17. S. Babar, A. Stango, N. Prasad, J. Sen, and R. Prasad, "Proposed embedded security framework for internet of things (iot)," *2011 2nd International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE)*, pp. 1–5, 2011.
18. B. Tian, Y. xian Yang, D. Li, Q. Li, and Y. Xin, "A security framework for wireless sensor networks," *The Journal of China Universities of Posts and Telecommunications*, vol. 17, no. 2, p. 118–122, 2010.
19. M. Díaz, C. Martín, and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing," *Journal of Network and Computer Applications*, 2016.
20. Z. Yang, Y. Yue, Y. Yang, Y. Peng, X. Wang, and W. Liu, "Study and application on the architecture and key technologies for iot," in *2011 International Conference on Multimedia Technology*, (Hangzhou), pp. 747 – 751, Institute of Electrical & Electronics Engineers (IEEE), 2011.
21. W. Miao, L. Ting-lie, L. Fei-Yang, S. Ling, and D. Hui-Ying, "Research on the architecture of internet of things," vol. 5, (Chengdu), pp. 484–5, IEEE, 2010.
22. T. Lu and W. Neng, "Future internet: The internet of things," vol. 5, (Chengdu), pp. 376–5, IEEE, 08 2010.
23. S. Krco, B. Pokric, and F. Carrez, "Designing iot architecture(s): A european perspective," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, (Seoul), pp. 79 – 84, Institute of Electrical & Electronics Engineers (IEEE), 2014.