Analysis of Assets for Threat Risk Model in Avatar-Oriented IoT Architecture

Ievgeniia Kuzminykh(✉) and Anders Carlsson

Blekinge Institute of Technology, Campus Grasvik, 371 41 Karlskrona, Sweden
{ievgeniia.kuzminykh,anders.carlsson}@bth.se

Abstract. This paper represents new functional architecture for the Internet of Things systems that use an avatar concept in displaying interaction between components of the architecture. Object-oriented representation of "thing" in the avatar concept allows simplify building and deployment of IoT systems over the web network and bind "things" to such application protocols as HTTP, CoAP, and WebSockets mechanism. The assets and stakeholders for ensuring security in IoT were specified. These assets are needed to isolate the risks associated with each of assets of IoT system. Example of Thing Instance's description and its functionality using JSON format is shown also in the paper.

Keywords: IoT · Avatar · Thing instance
Threat assessment · Security risk assessment

1 Introduction

The complexity of ensuring IoT security is that the system is heterogeneous, consists of many assets on each of the architecture layer. The experts from research organizations in the IT field, as well as equipment manufacturers, agree that providing IoT security on the vertical way is a complex task since security aspects will vary depending on the use case and scenario, the application domain and platform used [1].

Although, the threats in the IoT can be similar to those in the traditional IT network, the overall impact could be significantly different. That is why many experts in IoT security organizations focus on threat analysis [2–7] and risk assessments to estimate the impact if a security incident or a breach occurs.

Depending on the application domain of the IoT, a corresponding risk assessment is necessary to implement:

- highlight the specific threats inherent in this or that application and the assets on which it can affect;
- identify possible attack scenarios and distribute them in the context of a specific IoT service;
- determine what threats are critical and how they can be mitigated.

Supported by Swedish Institute Scholarship Programme.

O. Galinina et al. (Eds.): NEW2AN 2018/ruSMART 2018, LNCS 11118, pp. 52–63, 2018.
https://doi.org/10.1007/978-3-030-01168-0_6

Security threats for each of the application domain (such as Smart Cars, Smart Airports, Smart Hospitals, Smart Homes, Intelligent Public Transport, ICS/SCADA, etc.) are unique but there are also universal threats and attacks which most often appear in the applications that are based on the Internet.

In order to provide the general security requirements for the IoT system using threat risk modeling, the first thing to do is to identify the main security stakeholders, security assets, possible attacks, and, finally, threats for the IoT system. Using this general IoT threat model as a basis you can create a specific set of security objectives for a specific use case, IoT application domain.

In this work, we will try to highlight such assets that is necessary for further analysis of the treat risk model for the Internet of Things. We will also specify the stakeholders who are the connecting link between IoT devices, services and customers, as well as link between transfer and displaying the client commands onto smart things.

For describing the model of component interaction in IoT system we will use the avatar-oriented approach, since it allows us to merge objects into a system of objects, system of objects has more functionality than standalone object since the IoT application has complicate interface. If we assume Service as a key component of the IoT system then it displays only a single entity with a relatively simple interface that abstracts a significant amount of activity. Service is an atomic unit of functionality. Like a well-constructed object in object-oriented programming, from the Service Oriented Architecture (SOA), discipline begins, "the services are collections of capabilities". But the IoT Service has a more complex structure than a single entity. The application can use several services to display all information to the end user, can aggregate data from several devices. In simple words, opening the application on the smartphone, the customer wants to have access to the state of his home and all the devices in it, as well as to his car activities, monitor his health and nutrition, find free parking places and check traffic conditions in the city. At the same time, the customer, clearly, does not want to open the state of each physical sensor in his house but wants simply to see the general state, for example, "house is all right", and, also, user wants his position to be determined automatically for searching parking and route selection. Naturally, to perform such a functional, the service uses other services such as GPS and smart home control center in our cases, and after displays the data via the web interface to the user.

To manipulate the data objects the avatar representation approach is most appropriate, then you can easily connect or disconnect microservices, data from the things, or change the visual representation of data.

Avatars are designed to:

- expose objects as resources on the Web: the avatars can be invoked using semantic enabled service-oriented protocols;
- compose collaborative functionalities: they interact with the avatars of other objects to negotiate and fulfill requests requiring complex functionalities, thus enabling inter-object collaboration;

- manage context adaptation: they can adapt objects behavior according to their surrounding environmental changes;
- cope with pervasive setups: they allow network communication disruptions and support optimized communications with remote objects;
- deploy code on the objects: they either deploy application code modules onto the objects or execute them in a cloud infrastructure if objects do not have enough resources to do this [8].

2 Place of Assets in Threat Risk Model

First of all, to ensure security it is necessary to follow policies that are generally aimed at making the system more reliable and resistant to attacks. They should be adequate for a particular service or application platform and should contain well-documented information. When designing the IoT system it is necessary to take into account the features and context of the case of use itself, to determine the interfaces, communications and the instances that will be used during deployment. The IoT security system in the home environment will be different from the IoT in the critical infrastructure. Thus, the risk depends on the context, and regarding to this, security measures should be applied with this in mind.

To identify significant risks using a defense-in-depth approach, first of all, it is necessary to isolate the risks associated with each of assets of the IoT system. This should be done at the early stages of the life cycle of the program, at the design and testing stage. The Fig. 1 shows the interaction of the risk model components.

To compile a risk model we need to know:

- asset, A;
- application domain;
- list of potential threats to Asset, T;
- list of potential vulnerabilities, V;
- list of countermeasures and recommendations for risk mitigation.

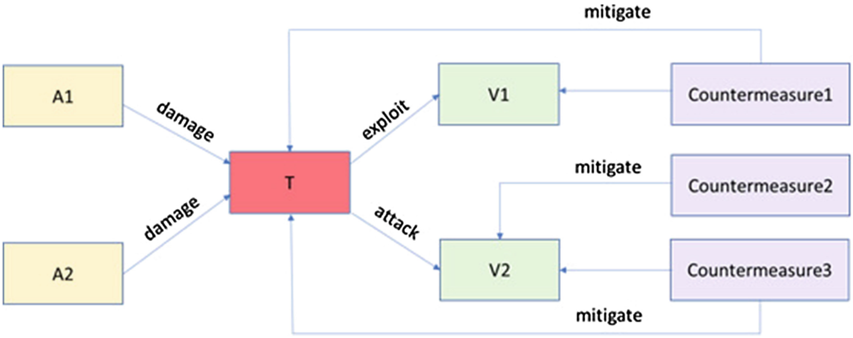

Fig. 1. Risk model components.

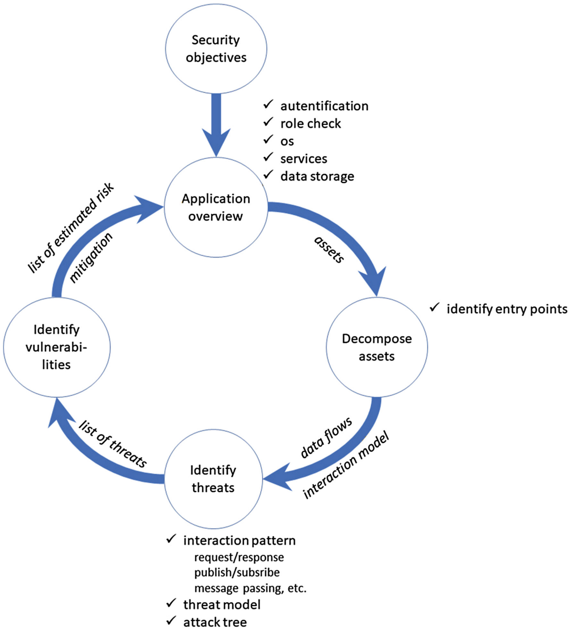

Fig. 2. Algorithm for threat assessment.

The list of threats will be based on a list of potential attacks, and the threat will be considered as the possibility of implementing an attack. The risk is estimated as the probability of the exploitability on Impact (Damage Potential). Regardless of which risk model will be chosen, for example, STRIDE, DREAD or Severity system, the threat algorithm for the risk model will look like this, as shown on Fig. 2.

After all operations to assess the risks of threats we obtain the estimated risk for each of the applications in this application domain and a list of recommendations for reducing these risks.

3 Architecture of IoT

Historically, each IoT solution was based on an application that required the interaction of things with each other and with the user, and therefore the system was developed for each application separately. Hence, there are such variety of architectures offered by different organizations in IoT field as NIST [9], ITU-T IOT [10], AIOTI, W3C WoT [11], and also architectures from manufactures as Microsoft Azure [12], Cisco [13], AWS, Google [14]. They suggest different layers in architecture, include different components, give different terminology, it is not difficult to get confused in this variety.

This paper will use terminology that is closer to the abstract representation of IoT but not to the physical representation. Such a decision was made because in the study we focus on the functionality of the IoT system components, and even on the combinations of these components. We can say that we consider the system as the interaction of functionals, the interaction between layers but not the elements on each level.

3.1 Classical IoT Architecture

A classical IoT architecture is presented on Fig. 3. It includes physical components on each layer such as Device, Gateway in comparison with the functional architecture. The classical architecture displays a transfer of data from the end device to storage and data handler from where the interaction with user begins. This is the time axis of IoT system. But apart from the time characteristics of the assets and the physical interaction of components, there are certain actions and events that occur in the system, as well as the reaction to these events. Such representation focused on actions and events is closer to security since it contains assets that are more convenient for manipulating the security language such as an action (property), an event, a reaction, which can be interpreted in the language of risk theory as: for action is damage, exploit, risk, threat, vulnerability; for event is attack; for reaction is logging, countermeasure.

The classical IoT architecture is more focused on the manufacturer of the device and cloud, on the physical structure of the service. For example, places for ensuring security are cloud, servers, routers, gateways, devices. But at the same time the transient processes are not considered when transferring an IoT object with its properties from the IoT device to the final user and vice versa. Data integrity can be lost when changing from one form of IoT objects to another.

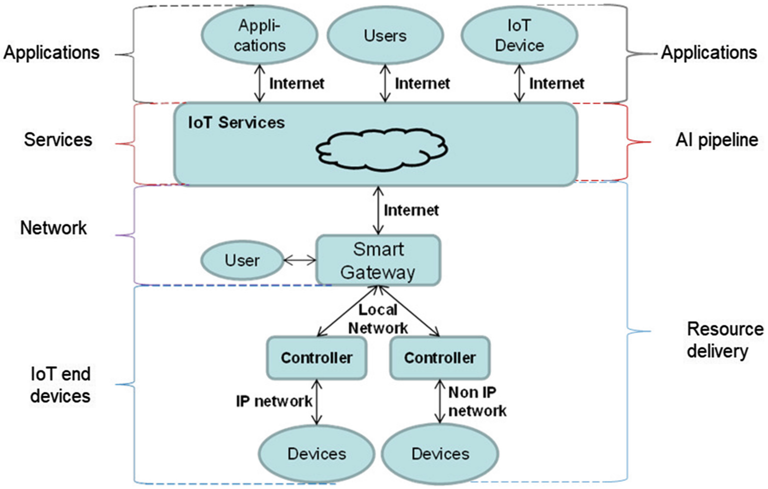

Fig. 3. Classical IoT architecture.

3.2 Avatar-Oriented IoT Architecture

The presentation of avatar-oriented model of interaction of the IoT components is shown on Fig. 4 and does not in any way eliminates the use of classical architecture. In addition, we take the assets which must be protected from potential attacks and threats from the classical model. It includes already not physical components but abstract components such as Thing Instance, Resource design, Service, Avatar.

In the classical model the lower level was a set of sensor devices that collect information and then forward it to the processing and control centers and to the storage with future visualization. In the functional model, all these functions of collecting, forwarding, processing are combined into one layer named *the resource delivery layer* the purpose of which is to deliver information to the service provider. Applications and services do not need primitive data from sensors, they need data of a higher level. At the level of the resource delivery a "thing instance" is created. *Thing instance* is an object representation of the merged data that contains the above-mentioned *data, metadata, interaction model* attributes, requirements for *communication* and *security.*

The resource delivery layer consists of the devices and the network instrumented so they can be addressed individually. The AI pipeline layer consists of platform that helps the resources to interconnect and of intelligence.

From the architecture on Fig. 4 it is not yet obvious which areas of IoT systems to protect. Also, it is not clear where the above-mentioned functionals of the model such as action (property), event, reaction. To move to language of the description of the system from the security point of view it is necessary to determine the so-called assets [8,15–19] which we will use in the threat risk modeling. Over IoT system the data and meta data circulate, last one describes the type of data and the interaction models inherent for particular application platform or service [20]. The requirements for communication, security and privacy must be implemented for effective interoperability of platforms. The security means that the system must support its functionality even during an attack. The privacy means that the system should protect the confidentiality of personal identifiable information.

Data – information that thing provides to user.

Metadata – supporting information about Thing instance. It includes Protocols and ports, Data formats & encodings, Multiplexing and buffering of data, Efficient use of protocols, Devices specifications.

Interaction model – link from a Thing to the interaction patterns it provides.

Security – links a given Thing to the security information that indicates the access metadata information for securely transmitting information via all the resources of the Thing.

Link – provides Web links to arbitrary resources that relate to the specified Thing instance.

The *interaction model* should support multiple interaction patterns and messaging methods. By default, interaction patterns contain of such assets such as

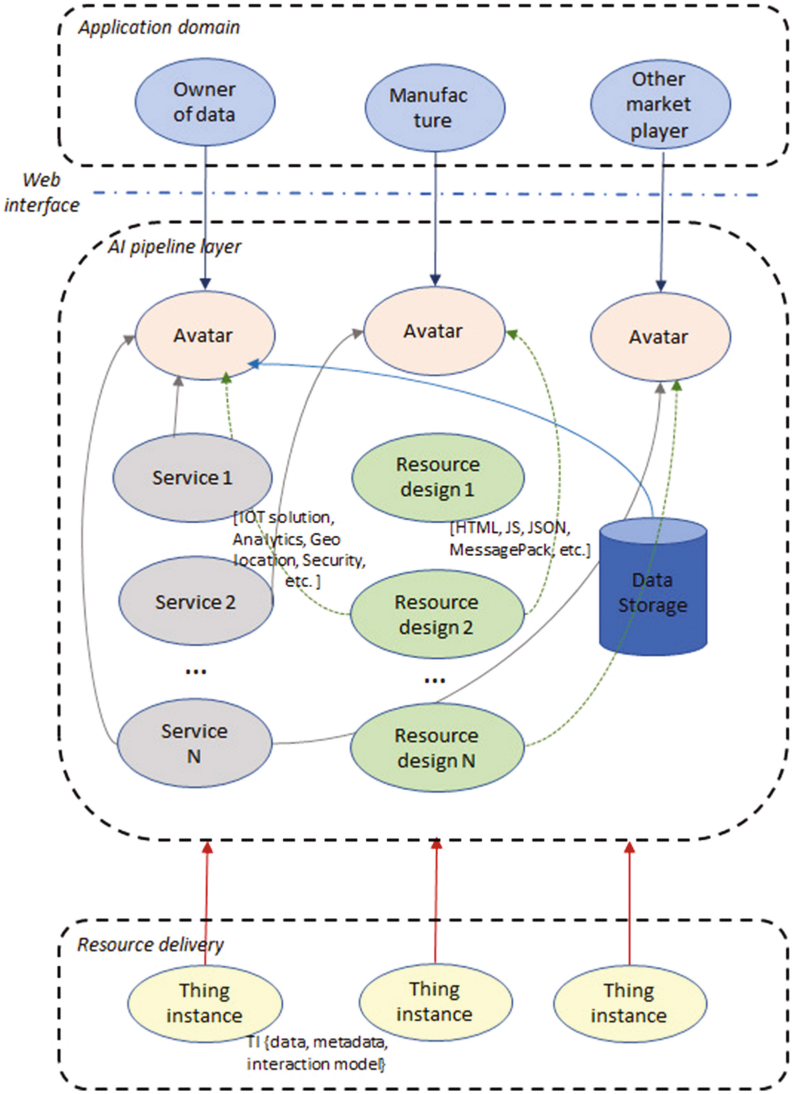

Fig. 4. Avatar-oriented architecture of IoT.

Property, Action, and *Event.* These assets were found to be able to cover the object model representation of any IoT Platforms.

Properties are abstract data points that can be read and often written, it displays the status of the object and stores a value, for example, boolean.

Actions are abstract invokable processes that may run for a certain time, they display object status changes, and are often a function.

Events are abstract interactions where the remote endpoint changes data asynchronously, most often the parameters of the function, threshold values and limited functions.

Simple example, for better understanding of “thing” representation via Thing instance, is shown on Listing 1. Listing 2 describes system of a smart room in a hotel where only a registered guest has access represented via JSON code and using avatar as object.

Door

```
{
  "name": "door",
  "description":
  "A door that should
  be opened when valid
  key is presented",

 events: {
   bell: null,
   key: {
     valid: boolean
   }
 },
 properties: {
   is_open: boolean
 },
 actions: {
   unlock: null
 }
}
```

Light switch

```
{
  "name": "light",
  "description": "Light
    is switched on when
    valid key is presented",
  properties: {
    on: {
      type: boolean,
      writable: true
    }
  }
}
```

LCD Display

```
{
  "name": "screen",
  "description": "A simple
    display that show
    notification to client",
  "events": {
    "write",
    "clear",
    "blink",
    "color",
    "brightness"
  },
  "properties": {
    "brightness": {
      "type": "integer",},
    "content": {
      "type": "string",},
    },
  actions: {
    is_displayed: content
  }
}
```

Listing 1. Simple example of Thing Instance representation.

Interactions between Things can be as simple as one Thing accessing another Thing’s data to get or change representation of data such as metadata, status or mode. A Thing may also be interested in getting asynchronously notified of future changes in another Thing, or may want to initiate a process served in another Thing that may take some time to complete and monitor the progress (for example, in different IoT use cases where they need access to GPS location or weather server to provide their own functionality). Interactions between Things may involve exchanges of data between them. This data can be either given as input by the Thing User, returned as output by the Thing Provider or both who are the main stakeholders of Thing.

Each Thing instance can have one or more virtual representations of physical or abstract entities which are called avatars. The Things can also have a story, for example, a car has a story about previous owners. The Avatars have attributes such as a history, patterns of interaction, description, services, identifiers, access control policy, data processing policy, security policy. The Avatars have URIs and are accessible via the web interface. They allow us to simplify the collection of services and applications that can use information from different sources.

```
{
  {
    context: {
    link = http://hotel.de/room12 },
    dependencies: {
    door: door12,
    light: switch12,
    screen: lcd12
  }
// invoked when service starts
function start () {
  door.observe(key, unlock); }
  function unlock(key) {
  if (key.valid) {
    door.unlock();
    light.on = true;
    screen.display (Welcome!);
  } screen.display (Door is locked!)
}
```

Listing 2. Simple example of avatar representation of smart room.

The avatar from the software developer point of view well presented in [11]. There are six functional modules that are responsible for interaction between avatar's attributes and with external components. In our avatar-oriented architecture from Fig. 4 on the top of AI pipeline layer that object of avatar described in [11] is partially formed. Partially is because some components are already inherited from previously formed objects, for example, the functions of the communication module that is responsible for selecting the right network interface and for selecting the right network was formed when Thing instance had been created.

4 Stakeholders and Assets

At each layer of the functional model it is possible to identify specific stakeholders. At the resource delivery layer when instance object is creating, the Manufacturer of the Device plays the role of the stakeholder. Stakeholder's function is to describe the characteristics of the model, properties, supported interaction models, all this information serves to create an instance.

The next stakeholder is the Thing Provider which uses the thing instance to build various specific solutions for different IoT domains. Thing Provider might define new instance or modify provided instance using the AI pipeline layer functionality. In addition, to maintain the integrity of the instance Thing Provider's privacy function increases at the AI pipeline layer. The thing instance can have several providers. In this case, the function of isolation of the Thing Provider increases with the subsequent division of the rights and preferences of users.

Stakeholder Thing User can be either a physical user or an abstract user, for example, if the instance is used by a business provider or company. Thing User must trust to two underlying layers his data and actions of physical objects (for example, video stream data from surveillance cameras or startup of the machine when certain thresholds are triggered). Thing User can differ in the functionality of using avatars and data itself, depending on the access rights. Some can change information, and some only read it. In this case, the function of ensuring the proper authentication and authorization of the user is increased.

Having information about the stakeholders and functional layers of IoT architecture, assets for security can be allocated. We can specify such assets as:

- Thing user data,
- Thing provider data,
- Thing instance itself,
- Interface (Administrative, Device Web Interface, Cloud Interface, Mobile Application) [21].

From the end user view the interaction with IoT system occurs in this way: the user through the IoT interface (for example, the browser on the smartphone, PC, smart TV) communicates through the IoT interfaces and IoT protocols with IoT network where the data about the object is stored in the form of Thing provider data and Thing instance, and user can perform certain actions on his own Thing user data.

Web of Things framework gives very limited approach for implementation security aspects in thing instance. Just couple of line presented in non-official draft of WoT standard [16]. For example, to provide simple security for Listing 2 we can add lines presented below:

```
"security": {
  "cat":"token:jwt",
  "alg":"HS256",
  "as":"https://authority-issuing.example.org"
}
```

Here as an example, JSON Web Token (JWT) type is assigned (cat), the corresponding hashing algorithm “HS256” (alg), and issuing authority of the security token (as).

5 Conclusions

Since IoT ecosystem is heterogeneous new vulnerabilities appear related to the interaction between the microservices on AI pipeline layer. More often the end user wants to have one application for many IoT systems where he can log in with the same credentials but do not download and open applications for each of the services. This multi-service also causes risks.

The security policy in the IoT must ensure the integrity of the thing instances and delivering it only to an authorized consumers, i.e. stakeholders, whether it is a service provider or an end user. According to the assets that were defined in the paper the attacks in IoT could be:

(1) Against Thing instances modifying
- property,
- action,
- event.
(2) Against Thing user data.
(3) Against Thing provider data.
(4) Against Interfaces.
(5) Against Communication.

For now, security metadata in Thing instance is defined as optional. That is why it is big challenge for researchers and software developers to implement security methods and mechanism for IoT that is avatar-oriented. Among the tasks under development are ensuring privacy and protecting Thing and related Assets against web attacks, DoS attacks, securing software and firmware updates.

Each IoT stakeholder should focuses on how devices and their resources must be secured so that they can only be accessed by authorized users and applications. Among the mechanisms that provides sharing Thing user data, Thing provider data, Thing instance itself in secure and flexible way there are well-known PKI, Encryption, TLS, OAuth, API tokens, JWT, delegated authentication, as well as specific to Web of Things concept mechanisms such as Social Wo Tot Social Networks authentication, WebSockets, Webhooks.

Once Things are connected to a public network, the most important problem to solve is how to ensure that only a specific set of users can access only a specific set of resources at a specific time and in a specific manner. In other words, if we back to smart room scenario, hotel guests (and only they) should have access to some services and devices in their room (and only there) during their stay (and only then).

For future work we are going to apply the threat risk model with the assets that were defined in the paper to the different IoT use cases, for example, to the hotel management system, health monitoring, SCADA network.

References

1. Baseline Security Recommendations for IoT. European Union Agency For Network And Information Security, ENISA (2017)
2. Ali, B., Awad, A.I.: Cyber and physical security vulnerability assessment for IoT-based smart homes. Sensors **18**(3), E817 (2018)
3. Hossain Md.M., Fotouhi, M., Hasan, R.: Towards an analysis of security issues, challenges, and open problems in the Internet of Things. In: 2015 IEEE World Congress on Services (SERVICES), pp. 21–28. IEEE (2015)

4. Macaulay T.: RIoT Control. Chapter 12 - Threats and Impacts to the IoT. Elsevier (2017)
5. Nurse, J.R.C., Creese, S., De Roure, D.: Security risk assessment in Internet of Things systems. IT Prof. **19**(5), 20–26 (2017)
6. Machine-to-Machine Communications (M2M). ETSI TR 103 167 V1.1.1 (2011)
7. Akatyev, N., James, J.I.: Evidence identification in IoT networks based on threat assessment. Future Gener. Comput. Syst. (2017). In press
8. Asset Avatars: Get a 360-Degree View of Your Assets - Whitepaper. Hitachi Vantara (2017)
9. Voas J.: Networks of 'Things'. NIST SP 800-183 (2016)
10. Reference architecture for IoT device capability exposure. Recommendation ITU-T Y.4115 (2017)
11. Mrissa, M., Mdini, L., Jamont, J.P., Le Sommer, N., Laplace, J.: An avatar architecture for the web of things. IEEE Int. Comput. **19**(2), 30–38 (2015)
12. Microsoft Azure IoT Reference Architecture. Version 2.0. Microsoft Inc. (2018)
13. Nivaggioli, P.: Cisco SP IoT Architecture. Cisco. https://www.cisco.com/c/dam/m/fr_fr/events/2015/cisco_day/pdf/7-ciscoday-10june2016-sp-iot.pdf. Accessed 29 May 2018
14. Guth, J., et al.: A detailed analysis of IoT platform architectures: concepts, similarities, and differences. In: Di Martino, B., Li, K.-C., Yang, L.T., Esposito, A. (eds.) Internet of Everything. IT, pp. 81–101. Springer, Singapore (2018). https://doi.org/10.1007/978-981-10-5861-5_4
15. Web of Things (WoT) Thing Description. W3C Draft. https://w3c.github.io/wot-thing-description/#introduction. Accessed 29 May 2018
16. Web of Things (WoT) Security and Privacy Considerations. W3C Draft. https://rawgit.com/w3c/wot-security/master/index.html#introduction. Accessed 29 May 2018
17. Zambonelli, F.: Towards a General Software Engineering Methodology for the Internet of Things. Cornell University Library, arXiv:1601 (2016)
18. ASAWoO project - Adaptive Supervision of Avatar/Object Links for the Web of Objects. https://projet.liris.cnrs.fr/asawoo/doku.php. Accessed 29 May 2018
19. Kuzminykh, I.: Avatar conception for "Thing" representation in Internet of Things. In: 14th Swedish National Computer Networking Workshop, Karlskrona, Sweden (2018)
20. BONSEYES - Artificial Intelligence Marketplace. https://www.bonseyes.com/. Accessed 29 May 2018
21. OWASP Internet of Things Project. https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project. Accessed 29 May 2018