BIG IoT – Bridging the Interoperability Gap of the Internet of Things

Deliverable 3.3.a

Security and Privacy Design for Smart Objects – first release

Version 1.0

Date: 15.12.2016

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688038.

List of Authors

Organisation	Authors	Main organisations' contributions
BOSCH	Jan Zibuschka, Stefan Schmid	T3.3. lead, coordination of discussions, final review with contributions to all sections, section 6
UPC	Juan Hernández-Serrano, Jose L. Muñoz, Oscar Esparza, Olga León	Sections 4, 5, 7, 8
AAU	Lars Mikkelsen	Section 7
SIEMENS	Arne Bröring	Sections 3, 8
ATOS	Wolfgang Schwarzott	Sections 4, 6

Responsible Person and Affiliation	Jan Zibuschka (Robert Bosch GmbH)
Due Date / Delivery Date	M12
State	Final
Version	1.0
Confidentiality	Public

European Commission | Horizon 2020 European Union funding for Research & Innovation

Version	Description of Changes	Date of Resolution
0.1	Initial Version	30.11.2016
0.2	Feedback from reviewers	12.12.2016
0.3	Consolidated feedback	13.12.2016
0.4	Additional feedback from reviewers	14.12.2016
1.0	Final version	15.12.2016

Table of Contents

Scope of this Document

This is the first version of D3.3 – Security and Privacy Design for Smart Objects. The document presents the work performed in T3.3 of the BIG IoT project during the first project year (up to M12). The content presented here includes the definition of security and privacy requirements and best practices, discussion of how to map them to the work within BIG IoT, the implementation support performed by T3.3 regarding BIG IoT API and marketplace, and a mapping of security mechanisms to a BIG IoT use case. For a better understanding of the contents, it is recommended readers know the BIG IoT architecture deliverable D2.4(.a), as concepts from this deliverable will not be fully reiterated here. The document will be updated and expanded by D3.3.b (M21).

1. Introduction

In the past years, the Internet of Things (IoT) has largely expanded and the number of IoT devices is evermore increasing. Today, IoT use cases span over a wide variety of application domains, ranging from smart homes over e-health systems to industrial environments. Things used in such applications are made available through IoT platforms. These platforms can be located on the device, fog, or cloud level.

A multitude of such platforms exists today. In order to enable cross-platform and even cross-domain application development, different initiatives are determined to form IoT ecosystems. An example for this is BIG IoT5 [6]. The BIG IoT project comprises overall 8 IoT platforms and is ready to grow beyond them. To ignite such an IoT ecosystem, BIG IoT focuses on establishing interoperability across platforms.

BIG IoT has two main objectives. The first one is defining a shared interface, i.e., the so-called BIG IoT API comprising common functionalities such as discovery, access, and event handling. This API needs to be supported by all participating platforms, often in addition to their existing proprietary interface, as illustrated in Fig. 1. The second objective is establishing a centralized marketplace where platforms as well as value-adding services can be registered, searched, and subscribed for by applications. In the BIG IoT project, these technologies are deployed in multiple pilot scenarios and involving various IoT platforms, services, and applications from the Smart Cities domain. We will provide an example of these scenarios in Sec. 5.

Besides the evident benefits that can be achieved by such IoT ecosystems, there are crucial challenges to deal with. In particular, new security threats must be addressed to allow the continued growth of such ecosystems. Frequently, sensitive data are stored, sent, or received by IoT platforms. Thus, security mechanisms are needed to protect these data from unauthorized access. Consider a patient who is wearing a glucose sensor that transmits its results to the IoT platform of a medical

centre. Security vulnerabilities may allow other entities to misuse this information or even put at risk the physical safety of the patient if these data are forged.

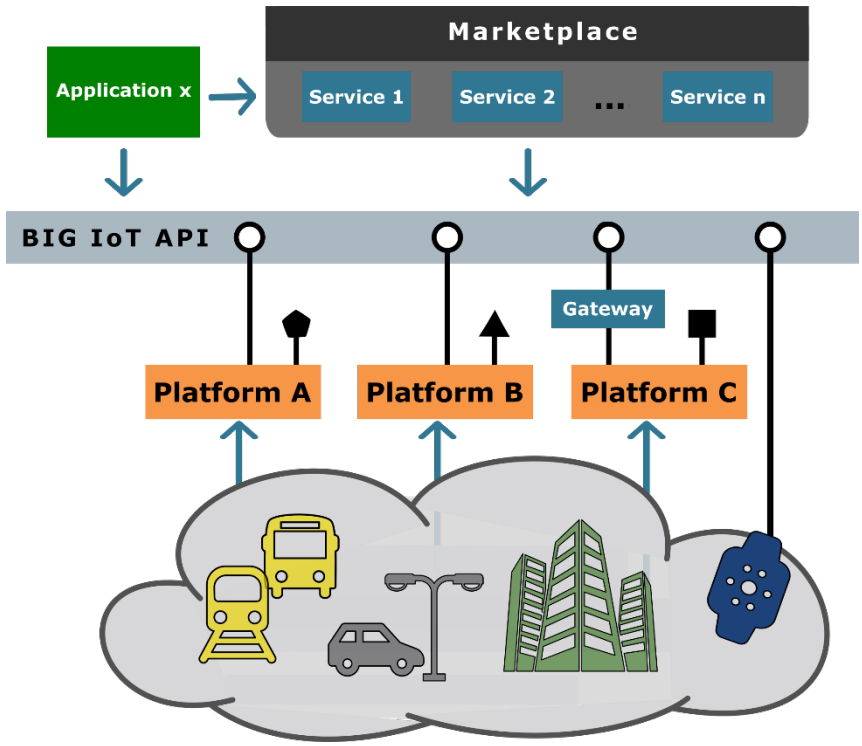

Fig. 1. The BIG IoT approach for building an ecosystem of IoT platforms[1]

Dealing with IoT security risks is challenging and can be more complex than in conventional networks, particularly for companies entering IoT ecosystems without any experience in the security field. Moreover, as new security vulnerabilities may be discovered over time, there is a need for updating IoT platforms on a regular basis. This might be hard to achieve in some cases either due to the simplicity of some device-level IoT platforms, or due to the lack of awareness of users or platform admins that forget or just skip updates. Finally, it may happen that some IoT platform manu-

[1] Icons by Freepik from http://www.flaticon.com.

facturers decide not to provide ongoing support nor security updates in order to reduce costs.

Last, but not least, privacy must be a mandatory concern. A privacy analysis should find an appropriate answer to the question: do the collected data allow drawing conclusions on individual human beings or onto small, specific groups of human beings? Note that such conclusions may be drawn by an unauthorized eavesdropper, and then this discussion is overlapping with confidentiality. The purpose of this article is to outline current discussion, analysis, and specific actions with regard to security and privacy in IoT ecosystems, and particularly to the BIG IoT realization of such an ecosystem. Requirements and best practices presented here will help to secure all the assets of the BIG IoT ecosystem and to prevent abuse of sensitive data.

2. Securing an IoT Ecosystem

The BIG IoT marketplace, the common API, and all the platforms/services/applications in the ecosystem must comply with a set of security requirements. After an analysis of the BIG IoT needs, seven security requirements were identified[2], which are presented in Sec. 2.1. Moreover, in order to face these risks, some solutions were already discussed (see Sec. 2.2).

2.1. Requirements

1. End-to-end security. IoT communications typically spread over several nodes and technologies. In particular, BIG IoT is not another IoT platform, it is a framework for a heterogeneous set of platforms, services, and applications. A possible solution to provide security would be to leave the mechanisms already in use for each platform, and then to define adaptation policies of these mechanisms in the boundary points of

[2] See also BIG IoT D2.3(.a)

platforms. The definition of these "low-level" relationships would highly increase complexity, as each individual security protocol (suite) provided by a component would have to be mapped to each protocol (suite) offered by each component it communicates with, which may fail, and hence should be avoided. The solution adopted in BIG IoT is to provide security at the API level, because it is common for all platforms. So, there is no need to adapt protection mechanisms between platforms, as the API is end-to-end by nature and assures that security remains independent of low level platform components.

2. "Batteries included but swappable". BIG IoT has to be designed to be capable of ageing in place while still addressing evolving risks [20]. There may appear new attacks, crypto systems, counter measures, techniques, and topologies, but the IoT system must be capable of dealing with these emerging concerns long after the system was deployed. Consequently, BIG IoT must ship a default but swappable security implementation, not hard-coded to specific security protocols/systems. Therefore, the aforementioned API-level security mechanisms need to be modelled and made exchangeable. An initial implementation of course still needs to be included as mandated by Req. 1.
3. Flexible authentication/authorization. The authentication and authorization systems used in the BIG IoT ecosystem must ease the management of identities and permissions. Features like single sign on and authentication without intervention of the BIG IoT authentication manager are key. Therefore decentralized, federated or delegated authentication must be supported.
4. Ownership transfer. BIG IoT should support safe transfer of ownership, even if a component is sold or transferred to a competitor; something that often happens during the lifespan of IoT nodes/components.
5. Accounting and charging. The BIG IoT must implement a secure accounting of resources consumption. This accounting must generate

enough charging data, typically in the form of a Charging Data Record (CDR), so that the desired charging policies can be enforced. As a result of a charging policy, a billing system may be necessary to generate invoices for service consumers. All these systems must be flexible enough to implement different business models and monetization strategies of services that can be implemented in the BIG IoT ecosystem. The BIG IoT marketplace must support offline and online charging and billing.

6. Continuous security. The BIG IoT system should be ready to respond to hostile participants, compromised nodes, and any other adverse event. Therefore, it is necessary to implement mechanisms and/or tools to reissue credentials, exclude participants, distribute security patches, updates, swap algorithms, or protocols, etc.
7. Secure development. Security must be a key part during the design phase of every BIG IoT software, but a secure design would be useless if development errors open unexpected attacks and/or vulnerabilities. Using a Secure Software Development Life Cycle (S-SDLC) and secure Source Code Analysis (SCA) would help developers to build more secure software and address security compliance requirements.

2.2. Addressing the Security Requirements in BIG IoT

Even though many strategies or decisions are still to be taken, some actions have already been adopted in order to address the above requirements. Requirement 1 is directly meet as the BIG IoT API is an HTTP(s) based API, and so it is end-to-end by design. Moreover, in order to comply with Requirement 2, the API should be flexible enough to handle any protocol and/or content. BIG IoT handles this by defining a very generic API; semantic annotations of the syntactic descriptions of each registered service and platform are then used to clarify the details on how to establish communication with these components.

Requirement 3 states that there is a need of providing flexible authentication in the IoT ecosystem. I.e., BIG IoT must implement an authentication and authorization system to be shared by participating platforms, services, applications, and end-users. Moreover, BIG IoT has to be able to work even when the authentication managers are not available. To solve this, BIG IoT uses an approach that is similar to the ones used by other widely-known IoT initiatives (e.g., [2]): signed manifests or tokens. A client presents a signed manifest to a server to demonstrate that it is able to perform a given action on a given asset.

When the server receives the signed manifest, it can trust the contents because the manifest is signed by a common centre of trust. Many state-of-the-art technologies have already dealt with the fact of using such signed manifests. Most solutions for the Web use JSON, CBOR, or XML encodings and rely on JSON Web Encryption (JWE) [10], JSON Web Signature (JWS) [9], XML Encryption (XML-Enc) [8], or XML Signature (XML-Sig) [4]. Obviously, one can decide to design a custom solution from scratch, which may seem at-a-glance a better suited solution. However, experience tells us that security protocols are subtle and often tricky. Consequently, the BIG IoT position is to adopt existing, already tested, security technologies. The specific set of solutions is still to be decided though. Given that the BIG IoT API relies on HTTP REST, potential candidates are SAML [1], OAuth [21], OAuth 2.0 [22], or OpenID Connect [7] (on top of OAuth 2.0), supporting delegated authorization and authentication/identification.

Requirement 4 must also be considered in the choice of the previous technology. The authentication/authorization system has to be defined with focus on easy management of identities and permissions, easing actions that are quite common in the IoT. This includes safe transfer of resources' ownership and quick response to dynamic topologies with frequent admissions and withdrawals.

Requirement 5 states that an appropriate accounting is key to develop charging/billing systems, both offline or online. An offline charging system just stores a CDR containing the relevant accounting and charging information (starting and end-

ing time, data used, bandwidth, etc). Then, the user is charged after resources have been used. In general, users being charged offline provide a bank account to pay the corresponding bill. On the contrary, when using online charging, the user typically buys a prepaid amount of credit. In this case, the charging system has to monitor online the resources consumption and then, needs to stop (or constrain) the service when the credit limit is reached. In both approaches (offline and online), it should be desirable to have non-repudiation proofs for both, the users and the marketplace to be able to verify consumptions, bills, etc. and to solve possible inconsistencies.

Requirement 6 forces the marketplace to host a secure repository where to securely download software and software updates/patches. This is a challenge that has often been addressed in the past and present. Experience here says that, apart from security, success depends on the ease of use for both end users and developers. The app stores of Apple, Google and Amazon are good examples, but BIG IoT is aiming for a more open approach for this component.

Requirement 7 makes mandatory the use of S-SDLC. To accomplish this, BIG IoT developers have to make use of the best practices for secure software development set up by the Open Web Applications Security Project (OWASP) [13]. First, the organization itself has to fulfil security related activities and software security practices, which are described in the OWASP Software Assurance Maturity Model (SAMM) [19] framework. Second, the applications have to meet requirements based on the OWASP Application Security Verification Standard (ASVS) [14]. Third, the application source code has to be analysed according the OWASP secure source code analysis (SCA) guidelines [15]. And finally the application will be tested for vulnerabilities and design flaws according the OWASP testing guidelines [16].

The OWASP SAMM framework builds the foundation of a secure development environment and organization. The BIG IoT development organization shall follow the twelve security practices and carry out the activities listed there at least to maturity level 2 but the ultimate goal should be to incorporate also the level 3 activities.

BIG IoT engineers currently lean towards the OWASP ASVS to define the security requirements for the applications and services. This standard (in its current version) defines 19 verification requirements. All these requirements have three security verification levels, with each level increasing in depth: ASVS Level 1 "Opportunistic" is meant for all software and its compliance adequately defends against application security vulnerabilities that are easy to discover; ASVS Level 2 "Standard" is meant for applications that contain sensitive data, such as business-to-business transactions, including those that process health-care information, implement business-critical or sensitive functions, or process other sensitive assets; and ASVS Level 3 "Advanced" is meant for the most critical applications, that is, applications that perform high value transactions, contain sensitive medical data, or any application that requires the highest level of trust. Responsibilities include controls for ensuring confidentiality (e.g. encryption), integrity (e.g. transactions, input validation), availability (e.g. handling load gracefully), authentication (including between systems), non-repudiation, authorization, and auditing (logging). Each ASVS level contains a list of security requirements, and each of these requirements can also be mapped to security-specific features and capabilities that must be built into software by developers.

For BIG IoT, developers should (at least) follow the ASVS level 2 requirements, and they could complete these with level 3 requirements according to the appropriate criticality. The task of SCA for the BIG IoT software will be based on the recommendations listed in the OWASP SCA guidelines. The Second Edition of the Code Review Guide has been developed to advise software developers on the best practices in secure code review, and how it can be used within S-SDLC. The SCA for the BIG IoT software should be done for all code by means of source code analysis tools, specialized on finding security related bugs. Also, all critical software parts will be manually reviewed.

Security testing of BIG IoT applications and web services will be based on the OWASP testing guidelines. The testing shall be performed manually by skilled penetration testers, but supported by a wide variety of automated tools. In the design

phase, developers should use automated tools for as much testing as possible, executing unit and integration tests for specific and relevant fuzz and abuse cases.

3. Best Practices for Privacy in IoT Ecosystems

Igniting an IoT ecosystem involves handling big data. Often these data contain sensitive information and therefore their use could be a threat to users' privacy. The FTC published in 2015 a guide containing best practices for privacy in IoT [18] that is summarized with the following statement: while flexibility in terms of data gathering is key to innovate around new uses of data, the amount of data storage should be balanced with the interests in limiting the privacy and data security risks to consumers.

These recommendations are useful and valid in the European scope. However, they are rather generic and they should be always complemented with a specific analysis of every use case (an example is provided in Sec. 5). In the following, we provide the main ideas behind the FTC recommendations.

3.1. Data Minimisation

Data minimisation is a long-standing principle of privacy protection [12] that means that a data controller 8 should limit the collection of personal information to what is directly relevant and necessary to accomplish a specific purpose. Since users' privacy is (or it should be) key for a wide adoption of the IoT, data minimisation is key to fostering the IoT ecosystem. Indeed, data minimisation can help guard against two privacy-related risks.

First, storing huge volumes of data increases the likelihood of receiving a data breach since there is more potential harm derived from such an event.

Second, collecting and storing large amounts of data also increases the risk of using the data in a way that departs from consumers' reasonable expectations.

To minimise these risks, organizations should develop data minimisation policies and practices providing answers to questions like what types of data it is collecting, to what end, and how long it should be stored. Such an exercise is part of a privacy-by-design approach and helps ensure that a company is sensitive with data collection practices.

In the EU, the data minimisation principle derives from Article 6.1(b) and (c) of Directive 95/46/EC [12] and Article 4.1(b) and (c) of Regulation EC 45/2001 [11], which state that personal data must be “collected for specified, explicit and legitimate purposes” and it must be “adequate, relevant and not excessive in relation to the purposes for which they are collected and/or further processed”.

When a company needs to gather and store sensitive data with a business purpose, it should consider whether it can do so with a deidentified data set. Deidentified data can reduce potential consumer harm while still promoting beneficial societal uses of the information.

A key to effective deidentification (anonymization) is to ensure that the data cannot be reasonably reidentified even with external cross sources. This usually requires removing identifiers or pseudoidentifiers. Although, at first glance it seems quite affordable, recognizing non-evident identifiers is quite a challenge that often has to be faced in a manual, use-case-specific manner.

In BIG IoT, for every specific use case, an analysis of potential identifiers among the data and/or metadata stored/exchanged is being performed. Data minimisation is encouraged specifically for the BIG IoT platforms and should account for cross data not only from other BIG IoT platform/services, but also from any other external source. An example for such data minimization technologies in the context of a BIG IoT use case is given in Sec. 5.

Notice that there is a common misconception about the added costs for data minimisation. Enhancing privacy by means of data minimisation techniques does not necessarily imply added costs. Indeed, data minimisation reduces the sensitiveness of data and hence lower security would be required. As a consequence, for instance,

in BIG IoT, important saving can be obtained in development costs due to a reduced ASVS level compliance.

3.2. Strong Accountability

As aforementioned, deidentified data sets can reduce many privacy risks. However, there is always a chance that supposedly deidentified data could be reidentified; especially because of the technology advances. For this reason, companies should have accountability mechanisms in place. In this context, the FTC has stated that companies stating that they maintain deidentified or anonymous data must meet three actions: (1) take reasonable steps to deidentify data, including by keeping up with technological developments; (2) publicly commit not to reidentify the data; and (3) have enforceable contracts in place with any third parties with whom they share the data, requiring the third parties to commit not to reidentify the data. This approach ensures that if the data are reidentified in the future, regulators can hold the company responsible.

Consequently, BIG IoT platforms, services, and applications should provide proper accounting mechanisms to securely log any action by any actor dealing with sensitive data.

3.3. Transparency and Easy Access

The centrepiece legislation at EU level in the field of data protection is the “Data Protection Directive” [12] which is implemented in EU Member States through national laws. This directive aims to protect the rights and freedoms of persons with respect to the processing of personal data by laying down guidelines that determine when the processing is lawful. The guidelines mainly relate to the quality of the data, the legitimacy of the processing, the processing of special categories of data, information to be given to the data subject, the data subject’s right of access to data, the right to object to the processing of data, the confidentiality and security of processing and the notification of the processing to a supervisory authority. The Directive also

sets out principles for the transfer of personal data to third countries and provides for the establishment of data protection authorities in each EU Member State.

In general, the conclusion is that EU's individuals need better information on data protection policies and about what happens to their data when it is processed by online services. As a result, the EU will require European organizations to publish transparent and easily accessible data protection policies. In this context, simple icons on websites and applications could explain how, by whom and under whose responsibility personal data will be processed. As a consequence, users are better informed about how and if their personal data is being exploited.

In the context of BIG IoT, the semantic description of what a given service or application consumes and provides should account for privacy issues. For example, it should account for:

- **Purpose** *non intended*, data are allowed to be used for not initially intended purposes); or *intended*, data are allowed to be used only for intended purposes.
- **Business Transfers** *allowed*, data can be bartered or sold; *not allowed* (data cannot be bartered or sold); *allowed informing*, if business assets are sold or transferred, corresponding information regarding customers could also be transferred.
- **Law enforcement** *allowed*, data may be given to law enforcement even when legal process is not followed; or *non allowed*, data may be given to law enforcement only when legal process is followed.
- **Advertisers** *allowed*, your data can be given to advertisers; or *non allowed*, your data cannot be given to advertisers.
- **Retention Policy** time that your data is kept (including indefinitely).

To better picture this fact, Fig. 2 shows a set of privacy icons proposed by Aza Raskin [17]. BIG IoT services and/or applications should use similar icons (or even those) to clearly show end users how their data are being processed.

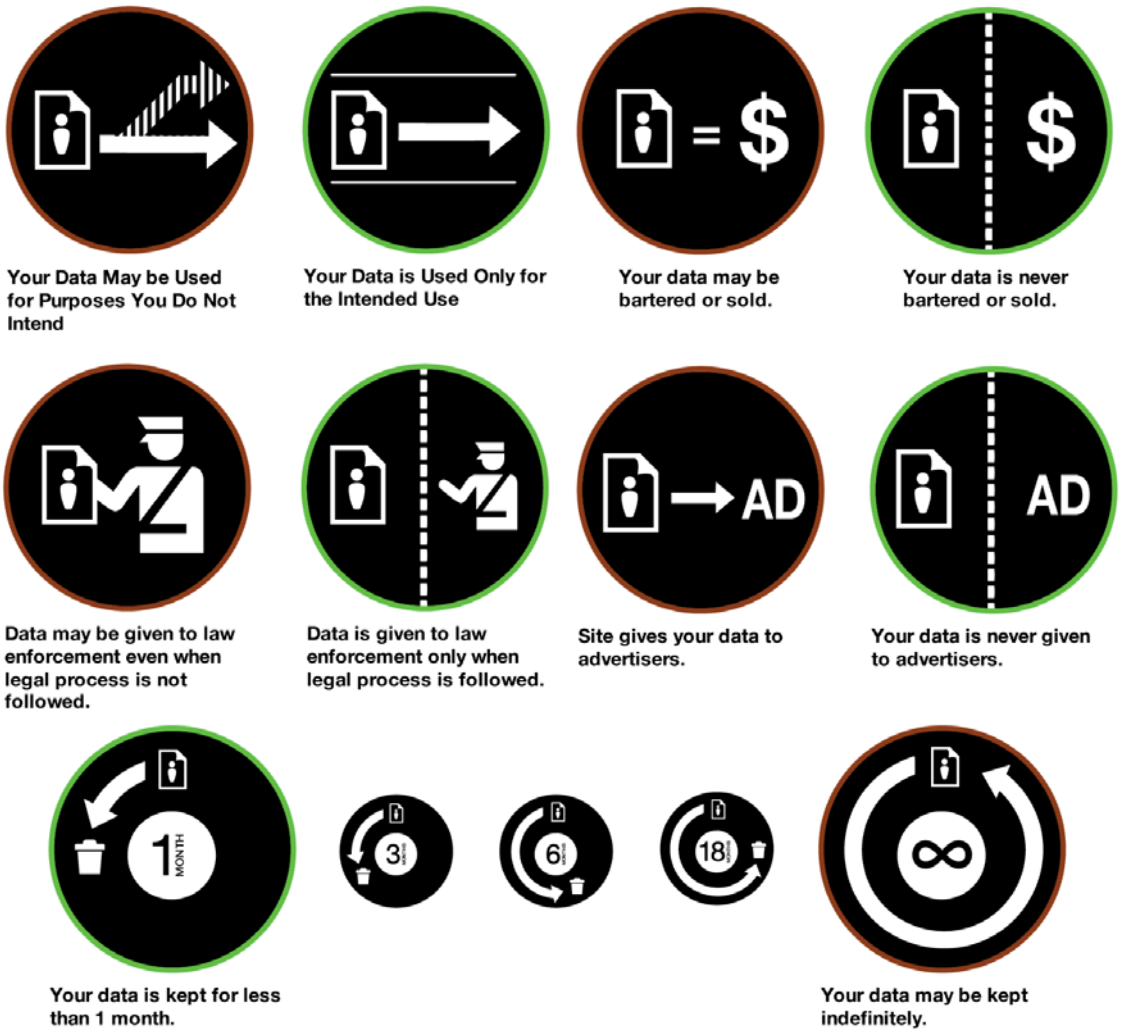

Fig. 2. Privacy Icons Proposal by Aza Raskin [17]

4. BIG IoT Interface and Marketplace

This section gives a quick overview of the access control in the BIG IoT API and Marketplace, the main focus of the implementation support within T3.3 during the creation of D3.3.a.

4.1. Overview

The analysis of the authentication/access control process with regard to the BIG IoT ecosystem and API in this deliverable covers the following stakeholders and architectural components[3]:

- Developer/User: The human authenticating to the marketplace to e.g. register BIG IoT Consumers or Providers
- Identity Provider (IdP): an external authentication service the developer uses as entry point for single sign on, authenticating to the IdP, which in turn provides a user identifier to the BIG IoT ecosystem.
- Provider: A BIG IoT software component providing information
- Consumer: A BIG IoT software component receiving and consuming information
- Marketplace: A BIG IoT component acting as an intermediary between users, developers, providers, and consumers

This section is work in progress, and is included to reflect the current state of the discussion in the context of BIG IoT marketplace and API implementation. At the time of writing, developer/user authentication is implemented using Auth0[4], while producer/consumer authentication is not implemented yet. We are also assuming in this discussion that we are using platforms with BIG IoT user management (i.e. not integration mode 3, see D2.4.a).

[3] See also BIG IoT D2.4(.a)
[4] https://auth0.com/

4.2. Developer or User accesses a Marketplace Portal

Initially, a developer needs to register and then log into the marketplace. To support Single Sign-on, this is performed using IdP. GitHub is a typical example for developers.

During the initial access, the Web request gets redirected to a login page. Since we want to use existing IdP (at least initially), a user gets redirected to external site (e.g. GitLab, Google) to authenticate himself there (based on Oauth2 Implicit Grant Type). Upon success, the Browser obtains an Oauth2 bearer token, which is used in subsequent accesses to the Marketplace (in the authorization header of the HTTP request). This is illustrated in Fig. 3, and will be described in more detail below. Note that the description only applies to the OAuth case, and Auth0 used in the actual implementation supports additional protocols beyond this. The discussion is based on material available on the Web[5].

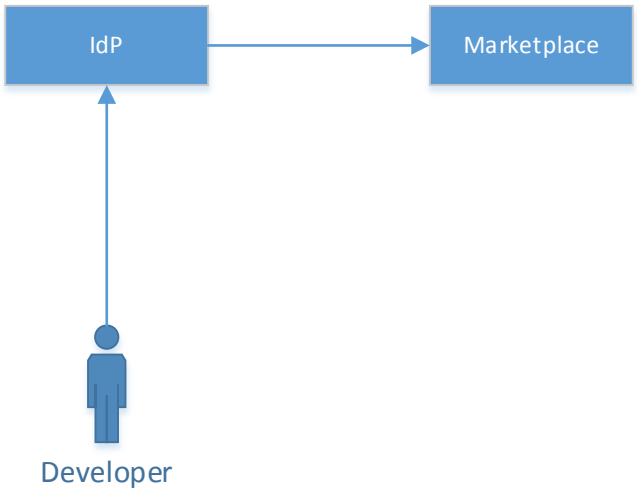

Fig. 3. User authentication using IdP

[5] https://stormpath.com/blog/what-the-heck-is-oauth

The BIG IoT Marketplace will provide a HTML button/Link that contains something like the following URL as a target:

```
https://login.idp.com/oauth?response_type=code&client_id=xxx&redirect_uri=xxx&scope=email
```

When the developer/user clicks this button, their browser will be directed to `login.idp.com`. There, the user/developer will have to authenticate, and may have to allow/deny additional information or operations the BIG IoT Marketplace has requested.

After this, there will be a redirection of the browser, returning it to the BIG IoT marketplace, which was encoded in the `redirect_uri` parameter. As a result of the authentication process, an *authorization code* is now also available, which will be encoded in the URL like this:

```
https://bigiot.eu/oauth/callback?code=xxx
```

The `code` query string value will then be exchanged for an access token using the IdP:

```
POST https://api.idp.com/oauth/token?grant_type=authorization_code&code=xxx&redirect_uri=xxx&client_id=xxx&client_secret=xxx
```

As response to this post request, the marketplace will receive an *access token* which then can be used to make additional calls to the IdP and retrieve the user's/developers information needed.

Response:

```
HTTP/1.1 200 OK
Content-Type: application/json;charset=UTF-8
Cache-Control: no-store
Pragma: no-cache
Set-Cookie: account=eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiIxNDQyNmQxMy1mNThiLTRhNDEtYmVkZS0wYjM0M2ZjZDFhYzAiLCJpYXQiOjE0Mzg3MDQxMjgsInN1YiI6Imh0dHBzOi8vYXBpLnN0b3JtcGF0aC5jb20vdjEvYWNjb3VudHMvNW9NNFdJM1A0eEl3cDRXaURiUmo4MCIsImV4cCI6MTQzODk2MzMyOH0.wcXrS5yGtUoewAKqoqL5JhIQ109s1FMNopL_50HR_t4; Expires=Wed, 05-Nov-2016 16:02:08 GMT; Path=/; HttpOnly
{
    "access_token": "eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiIxNDQyNmQxMy1mNThiLTRhNDEtYmVkZS0wYjM0M2ZjZDFhYzAiLCJpYXQiOjE0Mzg3MDQxMjgsInN1YiI6Imh0dHBzOi8vYXBpLnN0b3JtcGF0aC5jb20vdjEvYWNjb3VudHMvNW9NNFdJM1A0eEl3cDRXaURiUmo4MCIsImV4cCI6MTQzODk2MzMyOH0.wcXrS5yGtUoewAKqoqL5JhIQ109s1FMNopL_50HR_t4",
    "expires_in": 259200,
    "token_type": "Bearer"
}
```

The JSON Web Token (JWT) / OAuth2 Bearer Token can also later be used like this:

```
GET /admin HTTP/1.1
Authorization: Bearer 2YotnFZFEjr1zCsicMW...
```

After successful authentication, the developer can then interact with the marketplace, and e.g. register providers and consumers (see Fig. 4). This forms the basis for the next steps, where providers and consumers may have to authenticate to the marketplace independently of the developer/user.

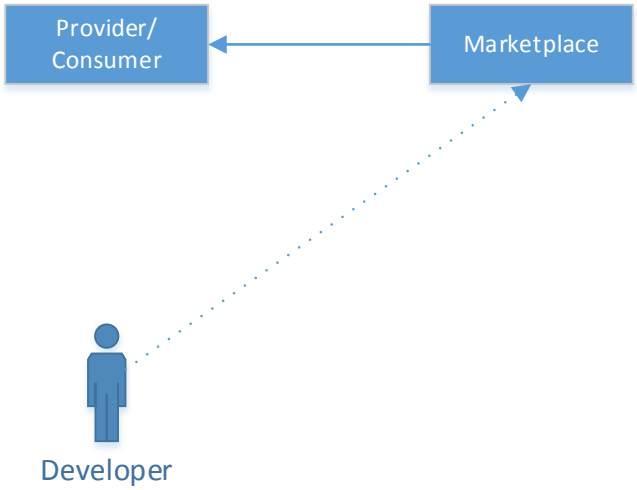

Fig. 4. User authentication using IdP

4.3. Provider or Consumer Instance accesses a Marketplace (Mx interfaces)

When a run-time instance of a Provider or Consumer accesses the Marketplace to authenticate, register, and discover information via the Mx interfaces, the Consumer/Provider requests a JWT / Oauth2 Bearer token from Marketplace (based

on *Oauth2 Client Credentials Grant Type*), following the same basic flow as above, but using *grant type*: *client_credentials* which uses *client_id* and *client_secret*.

Upon successful authentication, the Marketplace will provide a JWT / Oauth2 Bearer Token in the response, which is used in all subsequent interactions on the Mx interfaces.

The use of *client_credentials* will look something like this:

```
POST
https://login.idp.com/oauth/token?grant_type=client_credentials&client_id=x
xx&client_secret=xxx
```

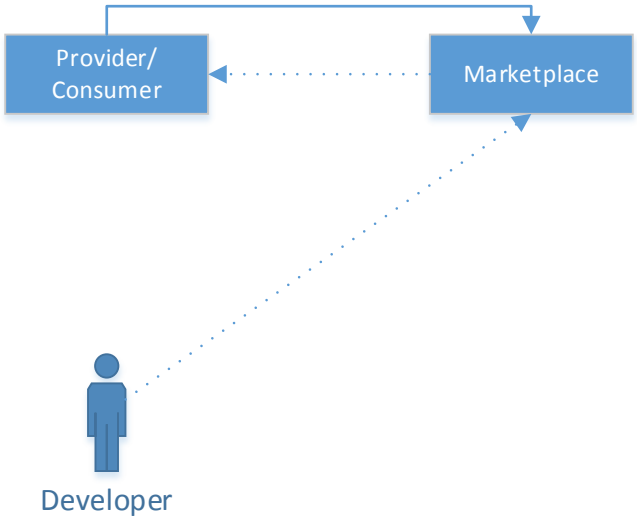

Fig. 5. Providers/consumers authenticate at marketplace (Mx interfaces)

The overall flow for Providers/Consumers authenticating to the marketplace is illustrated in Fig. 5.

4.4. Consumer Instance accesses a Provider Endpoint (A1 interface)

The communication between consumers and providers is a special case in that no delegation is involved. Therefore, it does not follow any of the OAuth flows, but

requires that the previously performed OAuth flows and interactions with the marketplace provide appropriate credentials to the involved providers and consumers to enable secure communications.

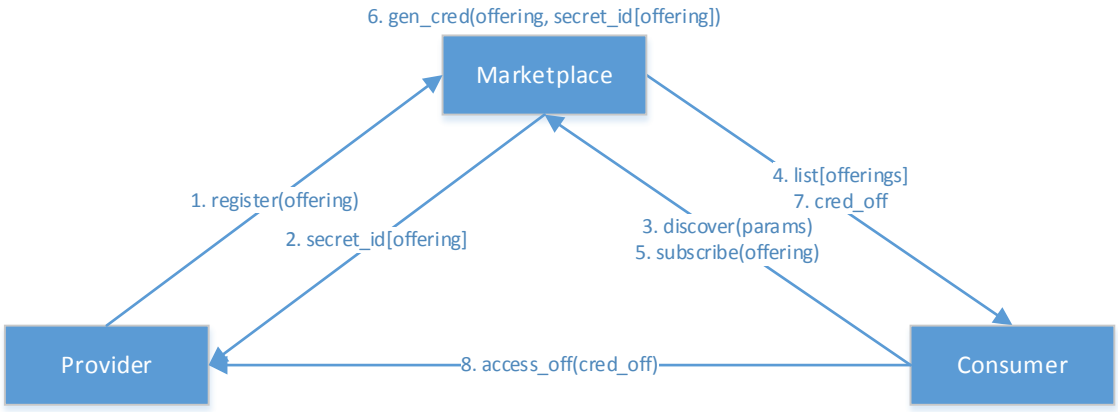

Fig. 6. Detailed Flow of Provider/Consumer Access Control (Mx -> A1 interfaces)

As depicted in Fig. 6, a run-time instance of a Consumer accesses the offered resources via the Offering Endpoint. In this process, the marketplace generated a shared secret (or key pair, discussions regarding symmetric vs. asymmetric cryptography are ongoing, see below).

The run-time instance of consumer obtains this as a JWT Access Token enabling it to access the Provider from the Marketplace, either upon subscription to an offering and/or as part of the accounting interactions with the Marketplace. It is assumed that the access token is refreshed on a regular basis.

The JWT Access Token provided by the Marketplace is signed either using:

- a shared secret for the Offering access, which is provided by the Marketplace to the Provider in the response to an offering registration
- the Marketplace's private key

The Consumer uses the JWT Access Token in all access requests (on A1) using the HTTP Authorization Header, e.g.

```
GET /bigiot/access/<endpoint> HTTP/1.1
Authorization: Bearer 2YotnFZFEjr1zCsicMW...
```

5. Use Case Security Analysis

In this chapter we will analyse selected use cases within BIG IoT in terms of security and privacy.

First we describe some use cases and analyse their security needs/requirements using the ASVS approach

Second we look at the interaction between the services, apps and objects in the use cases and the market place and/or the BIG IoT API in terms of which modes are used, and from this analyse the security and privacy needs.

5.1. Use Case Example: Smart Transportation Assistant

In this section we describe the use case of a transportation assistant in context of BIG IoT and analyse and discuss its security and privacy aspects. In this case, a subscriber of the app is at home and she wants to go to a specific place. The BIG IoT app allows her to be assisted in this decision by providing information about private and public transportation. Regarding private transportation, she can receive information about current traffic conditions. If she decides to use her private vehicle, she is assisted with navigation information while driving and is also assisted in finding available parking spots.

Regarding public transportation, the app can suggest several possible ways of transportation as an alternative to using the private vehicle. For instance, the app allows the user to select a bus line of interest. From this she can see live information about the next bus arriving at the selected stop. This information includes indications of where the bus is located currently, if the bus is delayed, the number of people on the bus, and a forecast of the number of people on the bus when it reaches her stop.

Based on this information, she can choose if she takes this bus or if she should switch to take another line, a later bus, or another mode of transportation. She can also choose to see historical information about the number of people on the bus based on location and time of day. This will allow her to plan ahead, i.e. if she wants to avoid overfilled buses she can see at which times of the day the buses are less loaded.

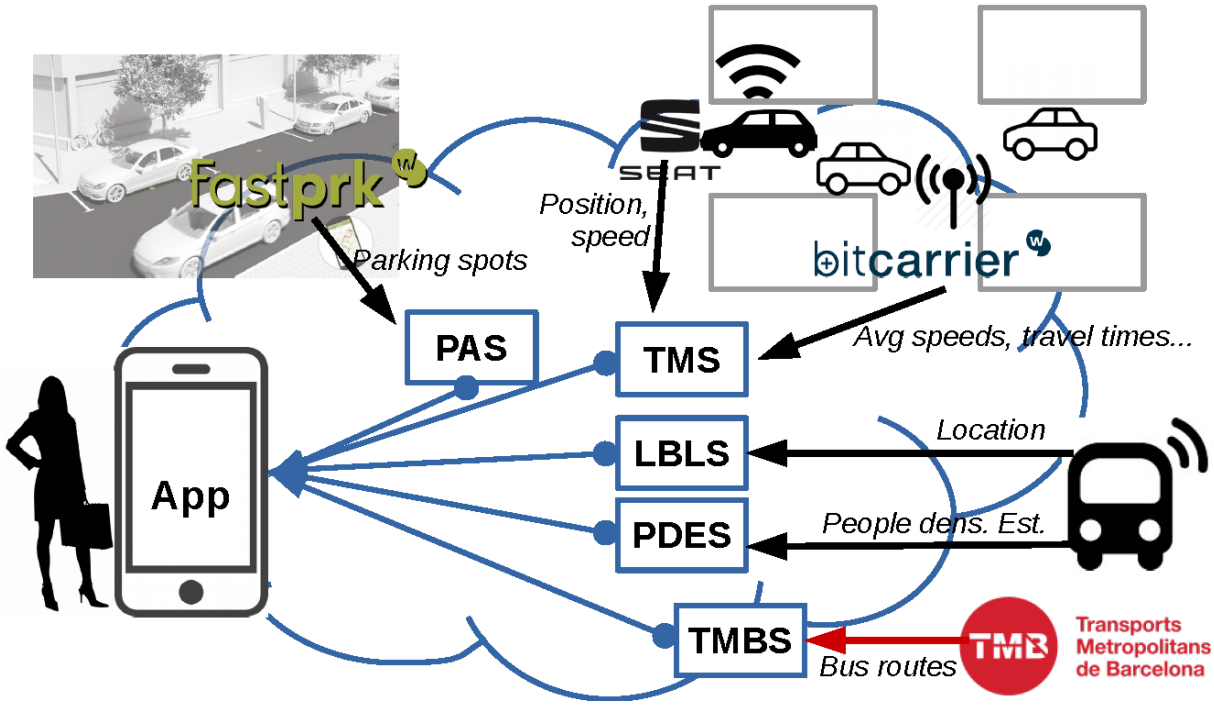

Fig. 7. Use case: a transportation assistant in the BIG ecosystem

We would like to mention that the previous use case is in fact implemented in two different pilots of the BIG IoT project. One pilot (mainly focused on private transportation) is going to be deployed in Barcelona and the other one (mainly focused in public transportation) is going to be deployed in Wolfsburg.

Finally, it is important to note that from the users' point of view, all the functionality is accessed with a single smartphone app; however, behind the scenes this app is consuming data from several services, which in turn consume data provided by

physical sensors operated by different platforms. A complete picture is presented in Fig. 7.

Services are an important abstraction layer in BIG IoT because they allow code re-utilization and simplify the process of building BIG IoT applications. For example, a service can aggregate data acquired from different platforms and then, present a unified dataset to apps. Another service could manage the history of data, allowing the user to access to past data (data not currently available in the source platform). Another service could create forecasts from data (acquiring data from a platform or from another service). Finally, another interesting use could be a service that anonymises an underlying dataset. This could allow lower levels of ASVS security in the upper layer (app). In the next sections we describe the involved BIG IoT components and discuss their security and privacy aspects.

5.1.1. Platform 1: Bitcarrier's WiFi/Bluetooth antennas [23]

This platform is fed by data gathered by WiFi/Bluetooth antennas placed at several street crosses. The technology detects vehicles/users by their unique MAC addresses and provides average travel times, speed, and even street congestion. MAC addresses are very sensitive data, which could be used to track or profile vehicles' (and even users') habits. This potential privacy invasion should be avoided. To do so, all the parties involved have to agree the necessary legal contracts in which they accept to properly use the data stored/exchanged. Also, these data must be appropriately secured/anonymised to avoid any kind of leakage (mistakenly or on purpose). Under this assumption, BIG IoT approach is to immediately anonymise unique addresses using a one way cryptographic-hash function. This function uses as inputs: (1) the address to be anonymised and (2) a key. This key is updated periodically, e.g. ranging from minutes to days. In this manner, one device cannot be tracked for more than a period. How often the key is updated is part of the privacy policy.

The use of cryptographic hash functions allows anonymising the data while keeping a trapdoor that could be used to re-identify vehicles/users. The platform operator must keep secret the temporal keys used to anonymise the identifiers.

However, operators may be forced to disclose these keys under some circumstances, e.g. a law enforcement requirement when a legal process is followed. From the above reasoning, this platform should comply at least with recommended baseline ASVS level 2 "standard". In addition, the management of the anonymisation keys should comply with ASVS level 3 "advanced", as it may allow an attacker to identify/track users and/or vehicles.

5.1.2. Platform 2: SEAT's cars

SEAT has put several cars with integrated sensors at BIG IoT disposal. These cars send their current position and speed. Providing these data may allow to track a vehicle and thus can be considered a threat to privacy. Same reasoning as for the platform 1 is applied: if identifiers cannot be removed from provided data, at least they must be properly anonymised, e.g. with a cryptographic hash function. Therefore, the same ASVS security requirements apply for both Bitcarrier's platform and SEAT cars; that is, cars should comply at least with standard ASVS level 2, but the key manager (if needed) would require ASVS level 3.

5.1.3. Platform 3: Fastprk's on-street parking spot status [24]

This platform can provide individual status of parking spots over a predefined monitored area. For instance, for the BIG IoT Barcelona use cases, this platform currently offers status information for 600 on-street parking spots. This kind of data entails specific privacy risks due to correlation with other sources: if an attacker knows where someone has parked their car, it can monitor when he/she leaves by checking the spot status. Obviously, a straightforward countermeasure would be, e.g., to provide free spots in a given street segment (a virtual lot). This approach will guarantee k-anonymity (a given individual cannot be differentiated from another k = 1) of moni-

tored vehicles/users with k being the number of vehicles parked on the same segment. The greater the segments are, the more anonymous the service is, but the PAS will provide less specific, potentially less useful, information. Intuitively, it seems that obtaining the exact free parking spot position or the segment where there is one (or more) free parking spots is likely to be equally useful for the end user; although looking for the appropriate trade-off between privacy and usability requires further technical discussion and studies of real users' needs. While some applications/services may allow different per-user degrees of privacy, this is not the case for this scenario. Therefore, testing user feedback about the suitability (or not) of just providing free spots on the street without their specific location cannot be done on an individual basis; it should be a global approach with, e.g., a pilot project. Since the data stored by the platform can be somehow used to track/monitor end users, we recommend securing this BIG IoT platform following ASVS verification level 2. However, if the platform just stores and provides free parking spots in a predefined segment/lot, ASVS verification level 1 could be considered.

5.1.4. Platform 4: WiFi probe catching sensors on buses

WiFi probe catching sensors are sensors placed on buses that collect WiFi probe requests, which contain MAC addresses, emitted from users WiFi enabled devices. Since the MAC addresses are unique, they must be anonymised in the same way as it is done for platform 1. Additional to the WiFi probe information, some context information is attached. This includes GPS location, timestamp, and sensor ID. This is both useful in the scenario of people density estimation, but also in providing live location of buses. Indeed, the operation of this platform is very similar to the operation of platform 1 and therefore the same security and privacy recommendations apply.

5.1.5. Service 1: Trac monitoring service (TMS)

This service is providing routes of cars to destinations based on current traffic conditions. With such a purpose, in this use case, the TMS consumes data provided by platforms 1 and 2 as well as a city map. Assuming that data provided by platforms 1 and 2 is anonymised, no specific requirements in terms of privacy are required. Regarding security, the software development has to comply with standard ASVS level 2 and, if properly justified, even with ASVS level 1.

5.1.6. Service 2: Parking availability service (PAS)

In this use case, the PAS is fed with on-street parking spot status provided by platform 3. In addition, the PAS may store statistics/historic of parking spot status and therefore the same privacy recommendations as for the Fastprk's platform applies here. Both the service and their connections must be protected with ASVS level 2 or ASVS level 1 if an acceptable level of k-anonymity is provided.

5.1.7. Service 3: External TMB bus routing data service (TMBS)

Transports Metropolitans de Barcelona (TMB) is the main public transport operator in the Barcelona metropolitan area. It already has an open data API [3] where to obtain routes to destinations with different public transports: trains, metro, buses. The TMBS makes the TMB API in the BIG IoT ecosystem. Since all the data involved in the service are public, no specific requirements in terms of privacy are required. Regarding security, unless properly justified, the software development has to comply with standard ASVS level 2.

5.1.8. Service 4: People density estimation on bus service (PDES)

The PEDS consumes data from the WiFi probe catching sensors and it provides information about the number of people on buses to Public transport load application.

The provided data consists of a bus id, estimation of number of people, accuracy indicator and timestamp. The provided data is stored for a fixed duration at the service, meaning that detailed load information on a specific bus can be requested within this duration. After the duration the data is minimised, such that only more general historic information is stored at the service, which is also made available to apps and services. It is recommended that the service complies with ASVS level 2, but it could even be ASVS level 1 as no user specific data is handled. However, to protect the business case of the service, i.e. to control who has access to the data and can use it, ASVS level 2 is recommended.

5.1.9. Service 5: Live bus location service (LBLS)

This service consumes data from the location sensors on buses and provides information to Live bus location app. The provided data consists of sensor ID, location, and timestamp. The data is stored at the service for a short fixed time duration, after which it is deleted. The service does not handle any user specific data but more publicly available information, why it is recommended to comply with ASVS level 1.

5.1.10. Smartphone App for end-user

The App consumes data provided by the 5 services. It would be expected to offer transparency functions, as mentioned in the privacy recommendations above, including privacy icons. Developers would also have to consider privacy requirements going beyond the scope of the BIG IoT infrastructure, such as informed consent or right of access to personal information, rectification, or deletion. Regarding secure development, apps should follow standard ASVS level 2.

5.2. Use Case Integration Modes with Interface and Marketplace

In this section we will analyze the app, the services and platforms, mentioned in the section above, in terms of which interface method they will use in interaction with the marketplace. The interface types are described in more detail BIG IoT D2.4.a.

The goal of this analysis is to highlight how usage of the different interface types will support security and privacy.

5.2.1. Platforms (objects)

For the platforms described and analysed in previous section we will now evaluate interfaces used. The setup and approach of the platforms are quite similar, meaning that they will all use the same list of interfaces.

The sensors will make their data available through various IoT platforms which will use the different integration modes. The standard mode is integration mode 1, integrating the BIG IoT Provider Lib directly into the IoT platform. The IoT platforms will all make the data from the sensors available to services by exposing it using the BIG IoT Provider Lib.

Interfaces the platforms will implement:

- P1 - used to interact with the BIG IoT Provider Lib. This lib will make the other interfaces available that are relevant as an offering providers.

Interfaces the platforms will use:

- M1 - used for authentication, to control who is allowed to access the data collected by the sensors. This will ensure that not everyone is allowed direct access to the data that potentially could be privacy sensitive.
- M2 - used for registration of the data as offerings in the marketplace. This is needed to enable services and apps to discover and access the data via the Marketplace.

- A1 - used for the actual access of the data from the sensors by services and apps. Here it is important that the A1 interface supports encrypted communication to ensure privacy and security of the information while transferring it.

Looking at the interfaces used the security will depend much on the P1, M1 and M2 interfaces, especially in terms of authentication and authorization. These interfaces will however only concern metadata, i.e. not the actual data offered. The actual data is transferred via the A1 interface, where it is vital that sufficient security and privacy mechanisms are implemented. This should be in terms of encryption of data, and credentials checking for access.

5.2.2. Services

The services will consume the data provided by the various sensors, and after processing of the data they will in turn provide information to other services and apps.

Interfaces the service will implement

- P1 - used to interact with the BIG IoT Provider Lib. This lib will make the other interfaces available that are relevant as an offering provider.
- P2 - used to interact with the BIG IoT Consumer Lib. This lib will make other interfaces available, allowing the service to discover and consume offerings.

Interfaces the service will use:

- M1 - used for authentication, both to control who will access the information offered by the service, but also when requesting access to the data offered by the sensors.
- M2 - used for registering the information as an offering in the Marketplace. This will allow other services and apps to discover the offering and use it.

- M3 - used for discovering offerings from sensors and other services in the marketplace.
- M4 - used for subscribing to offerings in the Marketplace. This will allow the service to subscribe to the information from the sensors, meaning that it does not need to poll the relevant IoT platform to obtain the information provided by the sensor, but will receive it based on the subscription made.
- A1 - used for the actual access to the data. Here the service will need to support the same encrypted communication methods that the sensor defines to ensure privacy and security of the information while transferring it.

Most of the interfaces (P1, P2, M1, M2, M4) will be used to exchange metadata about the actual data, and/or authentication. The actual data is transferred via the A1 interface, i.e. both when consuming data from the platform and when providing data to other services or apps. For the first case the security will be dictated by the entity offering the data. So if credential checking and encryption is implemented then the service must adhere to these. For the second case the service must define the security measures to be implemented. Most likely some credential checking and encryption of the data transferred.

5.2.3. App

The app described in previous section will use information from various services, and for that reason must adhere to the security defined for these interfaces.

Interfaces the app will implement:

- P2 - used to interact with the BIG IoT Consumer Lib. This lib will make other interfaces available, allowing the application to discover and consume offerings from the services.

Interfaces the app will use:

- M1 - used for authentication when requesting access to offerings.
- M3 - used to discover offerings in the marketplace to be used for the purpose of the app.
- M4 - used for subscribing to offerings in the Marketplace. This will allow the application to subscribe to the information from the sensors and services, meaning that it does not need to poll the relevant provider to obtain the information, but will receive it based on the subscription made.
- A1 - used for the actual access to the data. Here the application will need to support the same encrypted communication methods that the provider defines to ensure privacy and security of the information while transferring it.

As for the services most of the interfaces that the application uses will handle metadata and credentials exchange that the application must support to be allowed to access the required offerings. And also as for the services the application must adhere to the various security mechanisms defined by the providers, such as encryption and credential checking, when accessing the actual data.

5.2.4. Conclusion

From this analysis of interfaces, it can be concluded that by implementing encryption on the A1 interface will significantly increase security, as here actual data is exchanged. This is especially a requirement posed to the providers of offerings, because if these define security mechanisms the consumers of the offerings must also follow these to gain access to the offerings.

Furthermore, the requirements to the security of the A1 interface, and the credentials needed, should be exchanged securely using the relevant interfaces (P and M). Here it is the responsibility of the definition of these interfaces that this be handled properly.

6. Conclusions & Outlook

Nowadays, a plethora of IoT platforms and solutions exist, but yet no large-scale and cross-platform IoT ecosystems have been developed. This is mainly due to the fragmentation of IoT platforms and interfaces, as this variety results in high market entry barriers. The BIG IoT project aims at establishing interoperability across platforms in order to ignite an IoT ecosystem. Core technological pillars of BIG IoT are a common API as well as a marketplace for all participants of the IoT ecosystem, including devices, end-users, and service providers. Key to its success is to dene appropriate levels of security and privacy. Regarding security, in this paper we have identified seven requirements to be followed when creating and/or deploying BIG IoT components. Such requirements affect the design of the BIG IoT API and the marketplace, as well as any software in the BIG IoT ecosystem. Following this analysis, we have outlined how these requirements will affect the architectural approach of BIG IoT. Regarding privacy, we have proposed three recommendations that need to be followed by any IoT ecosystem participant: 1) data minimisation, i.e., that a data controller should limit the collection of personal information to what is directly relevant and necessary to accomplish a specified purpose; 2) strong accountability, i.e., to provide mechanisms to securely log any action by any actor dealing with sensitive data; and 3) transparency and easy access, i.e., any data controller should publish transparent and easily accessible data protection policies that clearly show how their data is being processed to the end users.

Notice that protecting users' privacy does not necessarily imply added costs. In fact, storing anonymised data can help in saving development and operational costs due to a reduced ASVS level compliance. Finally, we introduce a use case, and we have analysed it from the perspective of security and privacy. This use case presents an application that helps a user to get to a destination and to easily find nearby parking spots, as well as propose alternative routes by public bus. This use case is being implemented in two pilots of the BIG IoT project, in Barcelona and Ber-

lin/Wolfsburg. Additionally we highlighted how security and privacy can be improved and enforced through the use of the BIG IoT defined interfaces for use of consumers and providers of offerings.

In the future, our research will built up on the recommendations laid out in this article. By implementing various services and applications in the pilots of the BIG IoT project, which all need to follow the security and privacy framework outlined here, we will be able to evaluate our recommendations in terms of feasibility, practicability, and thoroughness. This will lead to sharpened and proven guidelines for the creation of IoT ecosystems in general, which we aim to contribute to our on-going engagement with standardization at W3C's Web of Things group [25].

Beyond the work on security and privacy best practices, we will focus our research agenda towards combining IoT security solutions with Semantic Web [5] technologies. The already available semantic descriptions of services and platforms in the BIG IoT project will enable us to develop ontologies that describe different security aspects. This will allow us to automate the selection of reasonable security measures and options per IoT ecosystem participant.

References

[1] Organization for the Advancement of Structured Information Standards (OASIS). Official Wiki of the OASIS Security Services (SAML) Technical Committee. url: https://wiki.oasis-open.org/security/FrontPage (visited on 09/14/2016).

[2] Allseen Alliance. Alljoyn Framework. Linux Foundation Collaborative Projects. url: https://allseenalliance.org/framework (visited on 09/14/2016).

[3] Transport Metropolitans de Barcelona. TMB Open Data. url: https://www.tmb.cat/en/web/tmb/about-tmb/open-data (visited on 09/22/2016).

[4] Mark Bartel et al. XML Signature Syntax and Processing (Second Edition). W3C Recommendation. June 10, 2008. url: https://www.w3.org/TR/xmldsig-core/ (visited on 09/14/2016).

[5] Tim Berners-Lee, James Hendler, Ora Lassila, et al. "The semantic web". In: Scientific American 284.5 (2001), pp. 28-37.

[6] Arne Bröring et al. "Enabling IoT Ecosystems through Platform Interoperability". In: IEEE Software Software Engineering for the Internet of Things (2017, forthcoming).

[7] OpenID Foundation. OpenID Connect. url: http://openid.net/connect/ (visited on 09/14/2016).

[8] Takeshi Imamura, Blair Dillaway, and Ed Simon. XML Encryption Syntax and Processing. W3C Recommendation. Dec. 10, 2002. url: https://www.w3.org/TR/xmlenc-core/ (visited on 09/14/2016).

[9] M. Jones, J. Bradley, and N. Sakimura. JSON Web Signature (JWS). Request for Comments: 7515. Ed. by Internet Engineering Task Force (IETF). May 2015. url: https://datatracker.ietf.org/doc/rfc7515/ (visited on 09/14/2016).

[10] M. Jones and J. Hildebrand. JSON Web Encryption (JWE). Request for Comments: 7516. Ed. by Internet Engineering Task Force (IETF). May 2015. url: https://datatracker.ietf.org/doc/rfc7516/ (visited on 09/14/2016).

[11] EU Legislation. Directive 45/2001/EC. 2001. url: https://secure.edps.europa.eu/EDPSWEB/edps/site/mySite/pid/86%5C#regulation (visited on 09/14/2016).

[12] EU Legislation. Directive 95/46/EC. 1995. url: https://secure.edps.europa.eu/EDPSWEB/edps/site/mySite/pid/74%5C#data%5C_directive (visited on 09/14/2016).

[13] Open Web Applications Security Project (OWASP). url: https://www.owasp.org/ (visited on 09/21/2016).

[14] OWASP. Application Security Verication Standard 3.0.1. July 2016. url: https://www.owasp.org/images/3/33/OWASP_Application_Security_Verification_Standard_3.0.1.pdf (visited on 09/14/2016).

[15] OWASP Code Review Project second edition guideline. url: https://www.owasp.org/index.php/Category:OWASP_Code_Review_Project (visited on 09/21/2016).

[16] OWASP testing guideline version 4. url: https://www.owasp.org/index.php/OWASP_Testing_Project (visited on 09/21/2016).

[17] Aza Raskin. Privacy Icons. url: https://www.flickr.com/photos/azaraskin/5304502420/sizes/o/ (visited on 09/14/2016).

[18] FTC Sta. Internet of Things: Privacy and Security in a Connected World.Technical report, Federal Trade Commission. Jan. 2015. url: https://www.ftc.gov/system/files/documents/reports/federal-trade-commission-staff-report-november-2013-workshop-entitled-internet-things-privacy/150127iotrpt.pdf (visited on 09/14/2016).

[19] The OWASP Software Assurance Maturity Model (SAMM). url: https://www.owasp.org/index.php/Category:Software_Assurance_Maturity_Model (visited on 09/21/2016).

[20] OWASP Internet of Things project. Principles of IoT Security. url: https://www.owasp.org/index.php/Principles_of_IoT_Security (visited on 09/14/2016).

[21] IETF OAuth WG. OAuth 1. url: https://oauth.net/1/ (visited on 09/14/2016).

[22] IETF OAuth WG. OAuth 2.0. url: https://oauth.net/2/ (visited on 09/14/2016).

[23] Worldsensing's Bitcarrier. url: http://www.bitcarrier.com/ (visited on 09/21/2016).

[24] Worldsensing's Fastprk. url: http://www.fastprk.com/ (visited on 09/21/2016).

[25] W3C Web of Things http://www.w3.org/WoT/ (visited on 9/21/2016)