mF2C

Towards an Open, Secure, Decentralized and Coordinated Fog-to-Cloud Management Ecosystem

D4.1 Security and privacy aspects for the mF2C Gearbox block (IT-1)

Project Number	**730929**
Start Date	**01/01/2017**
Duration	**36 months**
Topic	**ICT-06-2016 - Cloud Computing**

Work Package	**WP4, mF2C Gearbox block design and implementation**
Due Date:	*M6*
Submission Date:	*30/06/2017*
Version:	*0.8*
Status	*Final*

Author(s):	*Jens Jensen (STFC), Jolanda Modic (XLAB), Laura Val (WOS), Antonio Salis (ENG)*
Reviewer(s)	*Roi Sucasas (ATOS)* *Jordi Garcia (UPC)*

Project co-funded by the European Commission within the Seventh Framework Programme		
Dissemination Level		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission)	
RE	Restricted to a group specified by the consortium (including the Commission)	
CO	Confidential, only for members of the consortium (including the Commission)	

Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	10/06/2017	ToC and 1st draft	Jens Jensen (STFC)
0.2	13/06/2017	Merged UC contributions (generic, UC 1, 2)	Laura Val (WOS), Jolanda Modic (XLAB), Jens Jensen (STFC)
0.3	15/06/2017	Integration and editing	Jens Jensen (STFC)
0.4	16/06/2017	Contribution on UC3, analysis and tidying up	Antonio Salis (ENG), Jens Jensen (STFC)
0.5	22/06/2017	First internal review, comments addressed	Roi Sucasas (ATOS), Cheney Ketley (STFC)
0.6	25/06/2017	Second internal review, comments addressed	Jordi Garcia (UPC), Jens Jensen (STFC)
0.7	26/06/2017	Final internal round and comments addressed	Roi Sucasas (ATOS), Jordi Garcia (UPC), Jens Jensen (STFC)
0.8	29/06/2017	Final version and quality check	Lara Lopez (ATOS)

Table of Contents

List of figures

List of tables

Executive Summary

Deliverable D3.1 describes a security policy that defines three data protection levels: Public, Protected, and Private. Protected provides integrity protection and Private provides both integrity and confidentiality. This deliverable, D4.1, describes the application of this policy to the mF2C use cases, first by identifying two generic patterns of the use cases and then by going through the specifics of each use case individually. We provide a brief analysis of each use case (UC) with details in the Annexes, as well as highlighting the implications on the Platform Manager (PM) (whose responsibilities were listed in D2.6).

The analysis of the UCs shows that additional features are desirable and we make an attempt to list and prioritise them. Message delivery options – how hard the system should try to push a message through – and prioritisation of messages are highlighted as important; the ability to select alternative routes is interesting and should be investigated. In addition, message origin authentication is important for Protected data; this means that it is important not just to assert the integrity of the message but also who asserted the integrity (and/or who sent it – these are usually the same entity).

The analysis further shows that we need rules for processing Private data: while Private data is by definition (see D3.1 section 2) owned by an individual, processing Private data owned by two or more individuals does not necessarily mean they will share it; rather, the ownership would by default be the intersection of their access control lists (which would likely be empty). There is a need to define rules for when data has been sufficiently anonymised – or when there is a situation such as an emergency where normal access control can be overruled. This is likely to be an automated process, so we need a means to define policy and ruleset for determining the ownership, in particular in a way that is not surprising to the original owner of the data, but offers a level of transparency (and perhaps incentive). This is one of the core security targets for the policy feature of the PM.

1. Introduction

1.1 Introduction

D3.1 defines a security policy for an mF2C infrastructure; comprising three different levels of data security (see section 2.3). It is necessary to investigate how it applies to the platform running the use cases – hence this deliverable. The data flows – and their associated security requirements – described in this deliverable are implemented on platform manager functionality (previously known as "gearbox"), so we summarise those requirements as well.

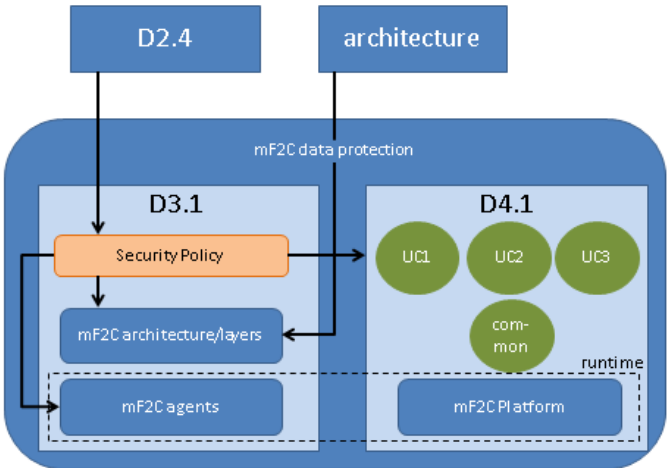

Figure 1: deliverable context

This document should/could be read in conjunction with D3.1 which describes the implementation of the security policy in communications in the different architectural layers and in the agents.

Figure 1 shows the context of the present deliverable. It describes the implementation of the security policy on the platform that supports the UCs, as well as the generic and specific data flows for the use cases themselves.

The structure of this deliverable is as follows:

- Section 1 (this section) is the introduction, which describes the aim and the context of the deliverable.
- Section 2 describes security from the point of view of the common aspects of the use cases, including the responsibilities of the Platform Manager in the context.
- Section 3 is an analysis of the specific requirements of the Use Cases.
- Section 4 briefly discusses the challenges identified in D3.1 from the point of view of the use cases.
- Section 5 lists a summary of high-level requirements.
- Annex 1 describes the implementation of the Platform Manager and prioritises its requirements.
- Annex 2 describes Use Case 1 (Alarm manager for smart infrastructure - WOS).
- Annex 3 describes Use Case 2 (Smart Boat System - XLAB).
- Annex 4 describes Use Case 3 (Smart Fog Hub Service - Tiscali).

1.2 Purpose

The objective of this deliverable is to cover the implementation of the security policy, described in D3.1, for the use cases. In particular, this deliverable describes the data flow for a generic Internet of Things (IoT) use case and then the specifics of each use case, with detailed descriptions in the Annexes. The deliverable finally summarises the requirements, taking into account also the requirements defined in D2.4.

In the DoW, the aim of this deliverable is to describe the security and privacy requirements of the "gearbox" (i.e. Platform Manager, or PM, cf. section 2.2.3 of D2.6); the aim is basically to describe the security of the components of mF2C which underpin the implementation of the use cases. Instead of repeating the description from D2.4, we focus specifically on the implementation of the security policy defined in D3.1.

1.3 Glossary of Acronyms

Acronym	Definition
AC	Agent Controller
ACL	Access Control List
API	Application Programming Interface
BT	Bluetooth
DoW	Description of Work
E2EE	End-to-End Encryption
FEX	Fair Exchange
IoT	Internet of Things
IT-x	mF2C software release iteration
LoRa	Low-power wide-area network wireless protocol
MQTT	Message Queue Telemetry Transport protocol
OPEX	Operational Expense
PM	Platform Manager
QoS	Quality of Service
SDR	Software Defined Radio
SFHS	Smart Fog Hub System
SLA	Service Level Agreement
TCP	Transmission Control Protocol
UC	Use Case
WiFi	Wireless protocol

Table 1. Acronyms

2. Use Case Data Flow – Security and Privacy aspects

The following sections describe the security functionalities arising from the UCs. Some control flows will be generic as they are common to IoT applications; others will be specific to each UC, e.g. which data needs which degree of protection. We initially cover the common use cases; then summarise the specifics for each individual use case.

2.1 The Platform Manager and the Agent Controller

One of the duties of this deliverable is to cover the security/privacy requirements of the PM.

For the reader's convenience, we list the responsibilities of the PM (D2.6, section 2.2.3):

- Service orchestration
  - Lifecycle management
  - Landscaper
  - SLA management
  - Recommender
- Distributed Execution Runtime
  - Task management
  - Task scheduling
  - Data management
  - Policies
- Telemetry
  - Intelligent instrumentation
  - Distributed query engine
  - Analytics

The security/privacy requirements of the agent controller (AC) (D2.6, section 2.2.2) were identified in D3.1 (along with some specific challenges for the agent controller). In the present deliverable, we shall thus assume that the AC already provides the security features identified in D3.1, and we can focus here on the additional features imposed on the PM.

Nevertheless, it makes sense to highlight where specific needs arise from the mF2C UCs also for the AC, as this deliverable covers the UCs from the perspective of the security policy defined in D3.1, and because there are shared challenges across both PM and AC. The main security features provided by (required of) the AC are:

- Identity management services for participants in the mF2C infrastructure
- Communication of data according to the security policy (namely, Public, Protected, Private)
- Shared security services, i.e. some types of services can be available to agents via a callout, as in a service oriented architecture.

A specific challenge shared by the PM and the AC is the classification and protection of processed data (see section 4.)

2.2 Generic Use case data flow

The generic use case – common to all three mF2C use cases – is summarised as follows:

> An edge device/entity arrives and registers, and selectively shares data or resources with other edge devices within a fog.

The purpose of doing it in generic form is that this flow will be more or less the same for each of the mF2C UCs. Thus, we avoid duplication (or triplication) of the description.

2.3 Summary of security policy

There are three data security levels (see D3.1):

- **Public** – no special protection requirements. The color used for this data is black. Note that Public is not the same as *published*; it just means there are no special protection requirements.
- **Protected** – needs to be integrity protected, but otherwise is not particularly sensitive. The color used for this data is green.
- **Private** – needs to be confidential in order to meet the security goals of mF2C. The color used for this data in the diagrams is blue. Security levels are incremental, so Private data includes the integrity protection of Protected data.

In addition, there are *boundaries* between the trust domains; these are indicated with a dashed line in *red.* Whenever data passes between trust domains, one needs to be particularly careful with the security parameters.

Annotation lines in the diagrams are thinner and coloured grey. Data lines are thicker than metadata/control lines.

2.3.1. Edge device arrives and (re)connects to a fog

Description: a client entity arrives and an edge device searches for local fog connections and connects (or reconnects). (No data/resources are shared at this stage.)

Explanation of flow (see Figure 2); the number refers to the port number in the diagram:

- Each edge device has an interface with which it can register and receive registration notifications from peers.
- This diagram assumes that the client identity is not based on a shared secret.
- The communication plane has an interface which can receive registration requests/notifications from edge clients, and can send notifications to subscribed edge clients.
- There is a means for the communications plane to publish available services endpoints to clients.
- This information needs to be integrity protected because a malicious client could alter the endpoint and substitute it with their own.
- There is a port for clients to listen/receive notification of available services endpoints.
- There is a means for the communications plane to store and receive services endpoints.

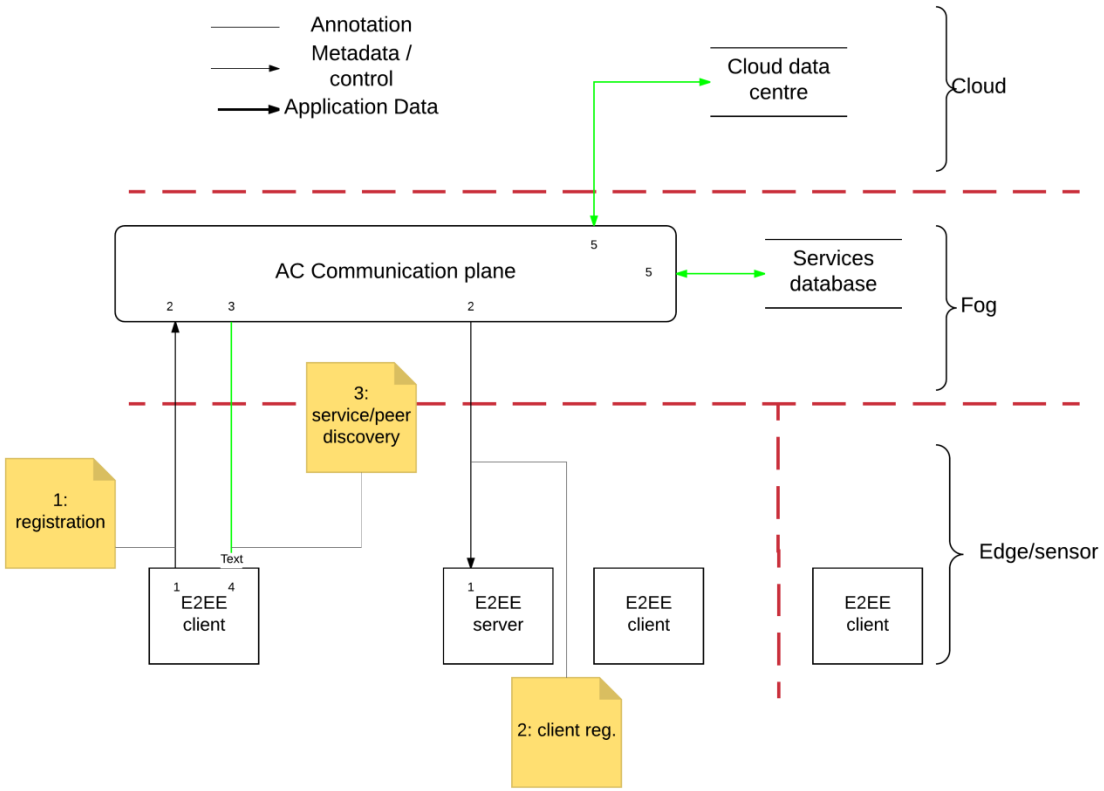

Figure 2: generic UC registration data flow

The security requirements expected to be met in the PM responsibilities are as follows:

- *Lifecycle* – credential, key, and identity management require Private, data for keys/credentials and Protected data for services/trust, e.g. trust anchors and revocation lists.
- *Landscaper* – Protected data for service availability and planning, perhaps particularly historical data upon which future planning is made; perhaps Public data for public information.
- *SLA management* is associated with the negotiation of services when the client registers. The security requirements were identified in D2.4; data is Protected and may even be Private.
- The *Recommender* data is expected to have to be Protected in order to prevent an attacker subverting it (e.g. directing traffic to particular services)

2.3.2. Edge devices share (sensitive) data with selected edge devices

Description: edge devices wish to selectively share sensitive (Private) data with each other. The sharing is done through an edge server which offers the capability to distribute the data; it may obviously also offer additional services such as processing of the unencrypted data.

Prerequisite: edge clients are already registered (**Error! Reference source not found.**)

Explanation (numbers refer to port numbers in Figure 3). From left to right, an edge client discovers the identities of the peers with which it wishes to share data (or alternatively discovers which of the peers are present). Having optionally done some processing on the data, the server sends the data onward to the additional recipients in encrypted form. The ports are as follows:

- Interface for discovering peers – and in particular whether a server is available (in which case the server identity/address/endpoint is sent)
- Interface for aggregating and sending list of peers and their capabilities.
- Interface for edge client to send data to fog. Data MUST be appropriately protected.
- Interface for edge clients to connect to and/or send data to edge clients.
- Optionally, data from the fog could be *broadcast* to the edge rather than sent to confidentiality.
- (Optional) interface for gathering anonymized information, e.g. usage metrics, monitoring, performance.
- Private interface (e.g. wire) where an edge client communicates within a physical link in the edge to a private data store or database.
- Data store/database interface communicating via physical link to edge client.
- Connection from fog to cloud.
- Cloud endpoint for aggregating information from fog.

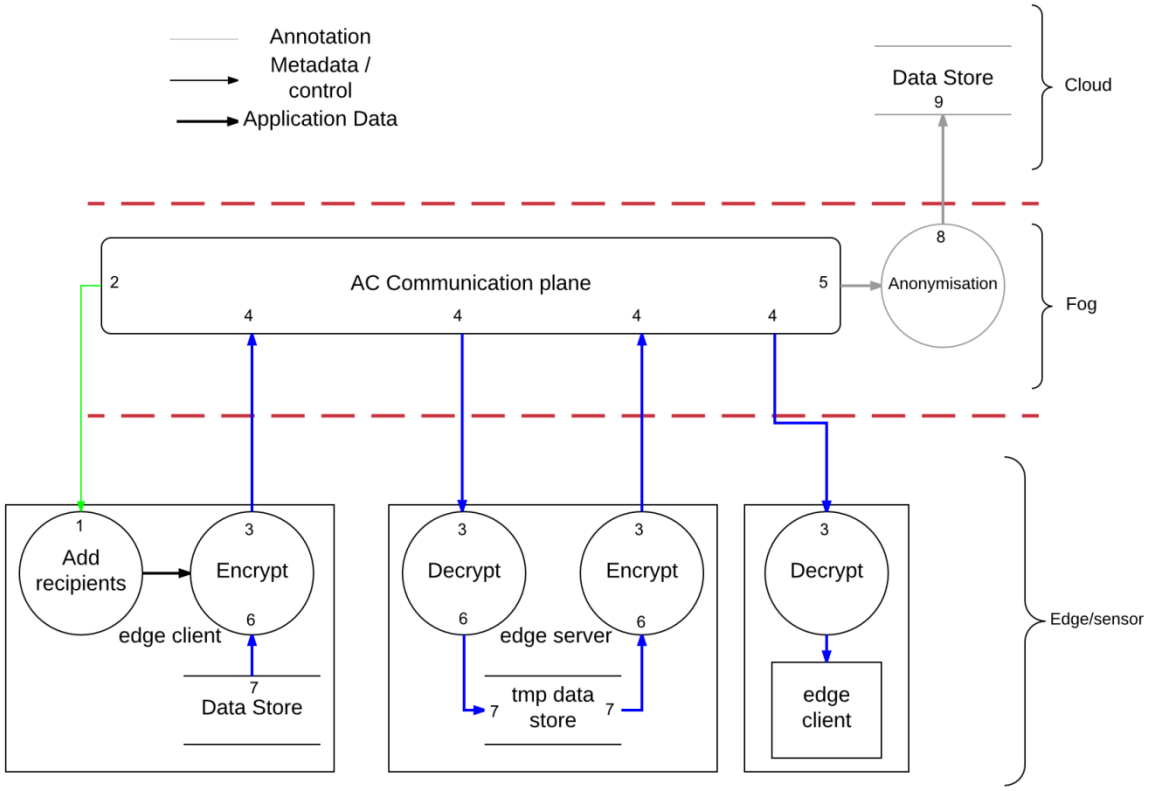

Figure 3: generic UC data sharing data flow

In assessing the security requirements of data sharing, we may assume the server has already been selected (orchestration), and the peers are known (as a part of the registration use case.) The additional expected security requirements on the PM are:

- *Task management* – tasks may be offloaded from the client to the server; the server will run a number of tasks appropriate for the number of clients. Depending on the sensitivity of the task, the related information is expected to be Private.

- *Task scheduling* is more likely Protected, as it may need scheduling in relation to other tasks on the server. However, one should be cautious in case there are only a few clients and information can be derived about their activity based on scheduling information.
- The runtime support for *data management* must follow the classification of the data. In other words, the PM's data management features must help the clients, agents, and apps implement the data protection required by the security policy.
- *Policies* for data sharing can give rise to interesting problems, particularly if Private data belongs to several people; see D3.1 section 5.2 and this deliverable's section 4.
- *Intelligent instrumentation* and *telemetry* may need to be Protected in order to prevent an attacker from subverting the system. One should also be careful about information being inadvertently released if there are few clients or clients with predictable usage patterns. More detailed *accounting* and *billing* information will likely be Private.
- A *distributed query engine* needs to be protected according to the classification of the data it processes; also as with any service discovery there will be Protected metadata.
- *Analytics* raises another one of the interesting challenges, if it processes data from multiple streams which may be Private (e.g. accounting data); see section 4. If less detailed telemetry information is processed – which need be only Protected – the problem is less acute.

3. Analysis of Use Case Specific Requirements

All use cases and uses of the PM are expected to include the generic use case description from the previous section. However, it is also worth looking into each use case individually, in order to see whether there are special data protection requirements.

The security policy is concerned with the protection of data – confidentiality and integrity. An analysis of the use cases highlights a different dimension, namely message delivery: whether (as in MQTT) messages shall be delivered at most once, at least once, or precisely once. Also message priorities (as in TCP's PUSH flag) could be considered but would probably be out of scope, or at least lower priority, for IT-1.

3.1 UC1 – Smart Infrastructure

The full details of data flow in this UC are provided in Annex 2.

Essentially four parties are communicating in UC1: LoadSensing, the Gateway, the alarm device(s) (Wearable), and the Jammer Detector. They communicate over different media; communication over Ethernet can be secure – if it's a private network – whereas those over WiFi, LoRa, BT, and cellular networks need to have security implemented and integrated with mF2C security. Indeed, it may make sense to be able to route messages across different networks, in case of unavailability of one network (e.g. through a jammer).

In terms of the Security Policy (section 2.3), the control data sent from, for example, the Gateway to the Jammer Detector, needs to be at least Protected; in particular, it makes sense to implement message origin authentication (in practical terms via a signed checksum). It also makes sense to ensure that only legitimate alarms can be raised, so the Wearable can detect whether an alarm originates from a trusted Gateway.

By default, no sensitive data is handled, so protection levels Public and Protected are required.

In terms of the PM, we expect that no further protection requirements arise (beyond those described in sections 2.3.1 and 2.3.2), due to the absence of Private data. However, as the UC specifically includes an availability attack – by an attacker with a jammer, Annex 2, A2.2 – one has to be careful not to give the attacker information that they could use against the infrastructure, so the use of Private data within the PM should not be fully excluded.

3.2 UC2 – Smart Boats

The full details of data flow in this UC are provided in Annex 3.

The parties participating in UC2 are the boats (through their Sentinel devices) and people (via their mobile devices). Data specific to UC2 is Private; UC2 data routed over non-private networks thus requires end-to-end encryption (E2EE). Sentinel devices are still authenticating, even when they are anonymous, since they hold key credentials. Devices communicate through cellular protocols (3G, 4G) or WiFi.

In terms of data sharing between agents, it is clear that the intended recipients – and only they – must have access to the data that is being shared with them. Framing this in the view of the security policy, this could be achieved either by adding all the intended recipients to the ACL for the Private data, or it could be achieved through the server decrypting the data and re-encrypting it for each

recipient (as in a secure broadcast.) The choice here may depend on convenience or it may depend on whether additional security features are desired[1].

Notably, the security system needs to interface not just with mF2C but also provide access to a BitCoin Wallet, in order to enable the customer's payment of harbour services.

In terms of the PM responsibilities, we highlight the following security expectations:

- As discussed in 2.3.2, there may be few boats in a particular area, so one should be careful with telemetry data inadvertently leaking information.
- As almost all of the data passing through the mF2C system is Private, it may be necessary to require that the some of the services that could be Protected in the generic UCs be Private in the case of UC2. See the summary in section 5.

3.3 UC3 – Smart Fog Hub System

The full details of data flow in this UC are provided in Annex 4.

The parties participating in this UC are people (through their mobile devices or potentially laptops), communicating with location-dependent (airport) services. All three data security levels are required; as generally personal data is involved, it may make sense to default to Private. Communications are generally over public (wired) networks as well as WiFi and BlueTooth.

This use case in particular contains a need for processing Private data and turning it into data which is no longer necessarily Private, e.g. building on users' location and behaviour to provide "heat maps" of where people are in the building, although the suggestion is to retain the Private level (but potentially with different ownership, e.g. assigned to airport security staff or managers).

Note there may be message delivery and/or priority options requirements, as for example, a user may rely on their device informing them that their flight is boarding and they may not be happy if they do not receive the notification.

In terms of the PM requirements, we highlight the following expectations:

- Users will have consented to different levels of data sharing. Providing *policies* for processing shared private data is an interesting problem (section 4);
- In particular, location is an interesting piece of data, as it is required in the UC (location-based advertisement or services, time to get to gate)
- Location is likely to be of interest also to people travelling together (where are my friends, where are my children), so the runtime must support secure identification of companions even when they have not shared Private data.
- The participation of children is a particular challenge in the runtime environment and the implementation of policies, as legally their parent/guardian take decisions for them; also advertising services to children is necessarily different from advertising services to adults (e.g. highlight a play area of the airport – or a place where they can see planes taking off – rather than advertising bars or smoking areas. Also advertising junk food to children is controversial and some parents/guardians will object.)
- It follows that the fact that a participant is a child may need to be available as Protected – possibly also the id of their parent/guardian – and the remaining data pertaining to the child must be Private (or at least determined by the parent/guardian.)

[1] For example, if the recipients are not wholly trusted to not leak the data, the conventional security measure ("traitor tracing") is that the server watermarks the data for each targeted recipients along with the targeted encryption.

- In case of an emergency, airport security and emergency services may need to have immediate access to user's data and clients.
- In case of criminal activity, investigators may require access to data after the event.
- In an airport in particular, there will be a large number of clients, and there will be likely be malicious clients. The PM's runtime support for execution needs to be sufficiently scalable (availability) in order for an attacker to not ruin the service for other customers (or worse, hinder handling of emergency situations.)

4. Discussion, addressing the challenges

In D3.1, we identified a set of challenges in implementing, or associated with enforcing, the data security policy. In this section, we discuss briefly these challenges from the point of view of the use cases. In summary, the challenges (D3.1, section 5.2) were:

- Multi-layered messages – a message can contain, for example, both encrypted and unencrypted information.
- Responsibility for control: Normally the sender is expected to ensure that the security policy is enforced, but this may not always be possible.
- Inferring the required policy for data which is not already tagged with a required security policy level.
- Processing Private data owned by different people. Who owns the outcome, and when can a service claim the data has been anonymised?
- Mobility of devices: when devices can connect via multiple networks, or to multiple Fogs, extra data breach risks can arise.
- Which security services to provide remotely (if locally) as shared services.

When we combine this with the requirements for the PM (section 2.1), we obtain some further extensions of the above challenges, plus a few new ones:

A particular challenge, number 4 in the list above, is what happens when Private data is processed that belongs to more than one person. Who owns the result? In particular, UC3 raises the question of emergency responders getting access to data, as well as the processing of data belonging to children.

5. Summary of Recommendations and Specifications

The PM itself, by virtue of its responsibilities in the mF2C infrastructure, will have protection requirements (such as telemetry data, which, if manipulated by an attacker, could lead to undesirable outcomes.) Conversely, due to its role in the lifecycle (2.3.1) and execution (2.3.2), it will also need to help clients, apps, and agents implement the security policy.

It turns out that the PM security requirements identified in the generic use cases, namely, sections 2.3.1 and 2.3.2, capture the PM requirements pretty well; the summary of the UCs in section 3 provide few additional requirements, but do highlight some interesting challenges which we have summarised in section 4. For completeness, we list in Annex 1 the list of security requirements identified in D2.4 as they apply to the PM.

The analysis of the use cases has highlighted some specific high-level requirements which we summarise (and prioritise) here. Priorities are given – estimated! – using the well-known MoSCoW system, referring to "Must have", "Should have", "Could have", and "Wishlist". If a priority is left blank, it is because there may not be a need.

As we gain more experience with the use cases, we may be able to factor out more common generic patterns, such as data anonymization (turning Private data into Protected or Public, e.g. by publishing aggregate numbers), and patterns related to the ownership of processed/aggregated data.

The following two tables provides an overview of requirements for the development and testing of the security – and the security features – arising from the use cases, but may influence the implementation of the AC as well as the PM. We don't know *a priori* whether all UCs need all the PM features, but may assume that they do.

The [*] below for UC3 indicates that all information pertaining to children should be considered extra carefully and may have to be Private.

PM functionality	Protection Requirement	UC1	UC2	UC3
Service orchestration				
Lifecycle mgmt.	Private/Protected	Protected	Private	Private
Landscaper	At least Protected (some data maybe also Public)	Protected	Private	Protected
SLA mgmt.	Protected (possibly Private, depending on application)	Protected	Private	Protected [*]
Recommender	Protected	Protected	Protected	Protected
Runtime				
Task mgmt.	Private	Private	Private	Private
Task scheduling	Protected	Private	Private	Protected
Data management	Depends on data	Protected	Private	Private
Policies	Protected. Must help enforce data security policies	Protected	Protected	Protected
Telemetry				
Instrumentation	Protected (and Private for more sensitive, such as accounting and billing information.)	Protected	Private	Protected

Query engine	Depends on data	Depends	Depends	Depends
Analytics	Protected or Private (depending on application)	Protected	Private	Protected

Table 2. PM security requirements summary

Moreover, the UCs highlight the following additional requirements (with estimated priorities):

#	Issue/requirement	UC1	UC2	UC3
1	Message delivery options (at least/most/exactly once)	**MUST**	**COULD**	**SHOULD**
2	Message priority options	**SHOULD**	**COULD**	**SHOULD**
3	Multirouting options	**COULD**	**COULD**	**-**
4	Message origin authentication for Protected/Private data	**SHOULD**	**COULD**	**COULD**
5	Data “declassification” process (e.g. anonymization)	**COULD**	**COULD**	**MUST**
6	Access to (external) payment services	**-**	**MUST**	**-**
7	(Private) Data ownership reassignment	**-**	**-**	**SHOULD**

Table 3. Additional security requirements from the UCs

Annex 1: Platform Manager Implementation

Like the generic use case, the mF2C platform must support the implementation of the use cases on mF2C. The purpose of this section is to cover the requirements that were identified in D2.4, i.e., they are covered by architectural layer (see also D3.1 for the general description), taking into account the requirements from the use cases (as opposed to D3.1 which summarises them from an architectural view), and prioritise them.

A1.1 Layer 0 Platform Manager

Requirement	PM Data Protection Requirements	Priority
Secure storage	**Data management.** In order to deal with Private data, PM should provide encryption at rest.	COULD
User and device authentication and authorization	**Lifecycle.** Private (authentication and authorisation) and Protected (revocation lists), as described in D3.1.	MUST
Key management	**Lifecycle.** Private (as described in D3.1).	MUST
Identity management	**Lifecycle.** Private/Protected (as described in D3.1; typically secrets are Private and revocation lists, if used, Protected.)	MUST
Policy management	**Policy.** Protected.	WISH
Logging protection mechanism	**Telemetry/monitoring.** Protected (but see also D3.1)	SHOULD
Access control	**Data management.** Private (access control lists are sensitive)	MUST
Trust	**Lifecycle.** Public/Protected (i.e. information is published; trust anchors such as certification authority certificates are Protected)	MUST
Data security and protection	**Data management.** Data should be protected according to the relevant policy. D3.1 recommends that data be tagged with its protection requirement.	MUST
API security	**(ALL)** Public/Protected/Private according to security policy: in particular, control information (e.g. for service orchestration) should be Private. See also the threat model in D2.4.	MUST
Web application security	**(ALL)** Public/Protected/Private according to security policy: in particular, control information (e.g. for web services implementation of APIs) should be Private. See also the threat model in D2.4.	MUST
Federation of security among multi clouds	**Orchestration/runtime** Considered out of scope for IT-1; but see also D3.1 which stipulates that data must be protected across multi-clouds	MUST
Heterogeneity	**Runtime.** Considered out of scope for IT-1; but the basic rule is that devices honour the security policy.	MUST
Integrity	**(ALL)** By definition, Protected and Private data offer integrity protection (MUST). As suggested by this deliverable (see section 4), we are recommending that integrity be implemented with data origin authentication (SHOULD). Note that integrity is also important for service offerings, e.g. SLAs.	MUST SHOULD
Confidentiality and	**(ALL)** By definition (see D3.1), Private data is owned by a user. However, this deliverable has raised the interesting	MUST

privacy	question of ownership of processed private data if private data from several owners are processed with a joint result; see section 4 for a discussion. Importantly for the PM implementation, control data is Private.	
Availability	**(ALL)** Not applicable to the policy. However, see also the threat model discussion in D2.4. Expected to be an issue particularly in UC1 (the jammer) and UC3 (malicious client.)	COULD

Table 4. PM layer 0 / cloud security

A1.2 Layer 1 Platform Manager

Requirement	PM Data Protection Requirements	Priority
Security management	**(ALL)** Layer 1 PM must honour the protection tag of data	MUST
Authentication and authorization	**Lifecycle.** Private.	
Access control	**Data management.** Private – only Private data have ACL, and conversely, the ACL itself is Private .	MUST
Data protection	**(ALL)** Private data – the PM at layer 1 must offer protection for Private data (e.g. through at-rest encryption, or through the physical security of the device)	MUST
Secure communication	**(ALL)** Communication must honour the protection requirement of data, i.e. both sender (MUST) and receiver (SHOULD) check the data against the required policy,	MUST SHOULD
Secure gateway	**Runtime.** Layer 1 devices must honour data protection requirements; they are often seen as gateways between edge and cloud, and are thus often responsible for enforcing the protection requirements, particularly when edge devices can only offer physical protection.	MUST
Intrusion detection	**Runtime, Telemetry**. Not applicable to data security policy directly; but see threat model in D2.4 (data protection may be compromised if a node is compromised.)	N/A
Virtualization security	**Runtime**. Like intrusion detection, is generally not directly related to the data security policy. It is discussed in the threat model in D2.4.	N/A
Identity management	**Lifecycle.** Private.	MUST
Integrity	As in Layer 0.	
Confidentiality and privacy	As in Layer 0.	
Availability	As in Layer 0.	

Table 5. PM Layer 1 / Fog security requirement

A1.3 Layer 2 Platform Manager

Requirement	PM Implementation	Priority
Authentication and authorization	**(ALL)** Private (ACLs are private data)	MUST
Access control	**(ALL)** Private.	MUST
Secure bootstrapping mechanism	**Lifecycle.** Protected or Private, depending on application.	MUST
Data security	**(ALL)** Also edge devices are required to honour the data	MUST

	security policy – otherwise they would leak data out of mF2C. However, the protection may be decided by ownership (data can copy to devices owned by the owner of the data) or by physical protection.	
Identity management	**Lifecycle.** Private.	MUST
Integrity	**(ALL)** As Layer 0 and Layer 1, but note that data origin authentication (as suggested above and in section 4) could be implemented through a private network connection instead of cryptographically.	MUST
Availability	**(ALL).** As in Layer 0 and 1.	
Confidentiality and privacy	**(ALL)** Private data must be protected also in the edge.	MUST

Table 6. PM Layer 2 / edge security requirements

Annex 2: Detailed UC1 description

A2.1 Background

The use case that WorldSensing (WOS) is planning is an alarm manager for smart infrastructure.

In this solution there are several elements which are deployed as micro agents (**LoadSensing**, **Wearables**, **Jammer detector**) and the smart agent (**Gateway**).

The alarm manager has to detect if the **LoadSensing** and the **Gateway** aren't able to communicate (using LoRa interface) with each other (number 1 in Figure 4). After detecting this lack of communication, the **Jammer detector** has to be powered up and configured to detect jammers in the channel used for the LoRa communication (number 2 in Figure 4). After these two steps have been done, an alarm has to be given through WiFi/BT or other non-wireless interfaces (number 3); these non-wireless interfaces could be signalling lights.

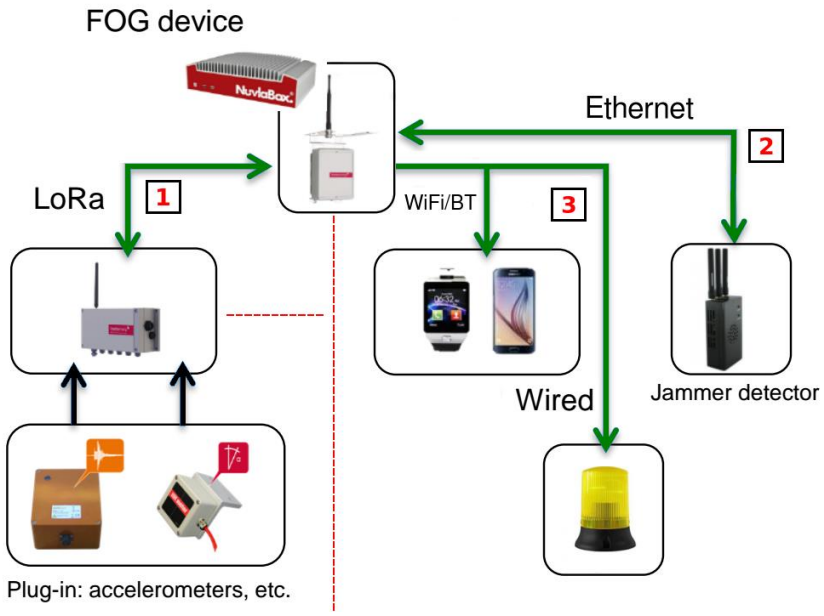

Figure 4: UC1 overview

From the technical point of view, it would be interesting to create a **Gateway** (smart agent) to communicate with the **LoadSensing** (micro agent) through a LoRa connection and control it with the **Jammer detector**, to reduce latencies and improve reliability compared with cloud based solution.

From the commercial point of view, it would be interesting to create a platform that could let us have more control, which would reduce the total cost of the solution due to independence of network providers compared with the cloud based solution.

The main characteristics of the agents implementing the use case are:

- Micro agents:
  - LoadSensing: It needs to have a LoRa interface used in the communication with the Gateway. As the LoadSensing is going to have the possibility of being battery powered, its consumption has to be as low as possible.
  - Jammer detector: It has to be hardware powerful enough to be able to process a large amount of incoming SDR data and it has to be able to get Gateway configurations. It needs an ethernet interface to communicate with the Gateway.
  - Wearable: It has to have WiFi/BT wireless communication.
- Smart agent:
  - Gateway: It has to manage all the micro agents, so it has to have LoRa interface (LoadSensing), WiFi/BT (Wearables) and ethernet (Jammer detector).

A2.2 Data flow

The data flows in the WorldSensing (WOS) solution are (the numbers are equivalent to those in Figure 4):

1. LoadSensing to Gateway: The LoadSensing gets sensors information and sends it to the Gateway (LoRa), which gets this information and stores it. It is also possible that the Gateway sends configuration messages (LoRa) to the LoadSensing to set the characteristics wanted.
2. Jammer detector to Gateway: The Gateway passes the needed parameters (ethernet) to locate the jammer easily (for example the frequency and the channel used for the LoadSensing that is having problems). The Jammer detector handles all of the SDR information and passes (ethernet) the final decision (one jammer has been detected or not) to the Gateway.
3. Gateway to Wearables: The Gateway sends (WiFi/BT) alarm information (it is possible to have no alarm, alarm without jammer and alarm with jammer) to the Wearables.

There are four different possible cases data flows.

The first one happens when no messages are received from the LoadSensing, the Jammer detector is powered on and no jammer is detected. An alarm is given to the BT wearable.

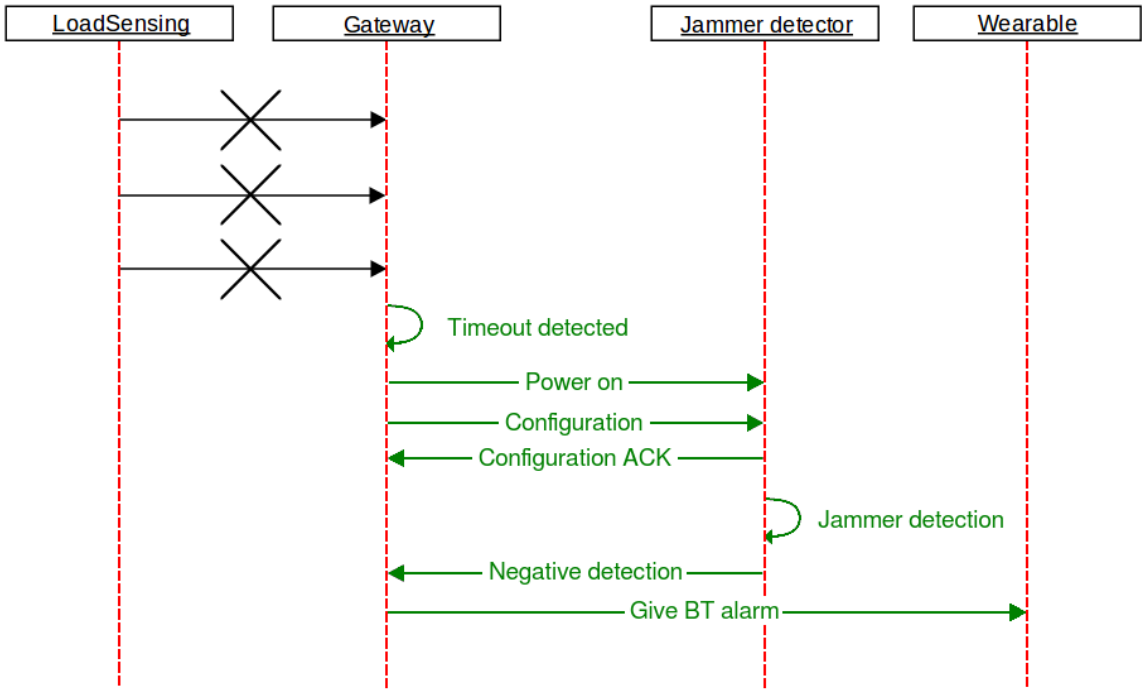

Figure 5: UC1 dataflow 1/4

The second one happens when no messages are received from the LoadSensing, the Jammer detector is powered on and a jammer is detected in the BT bandwidth. An alarm is given to the non-wireless alarm.

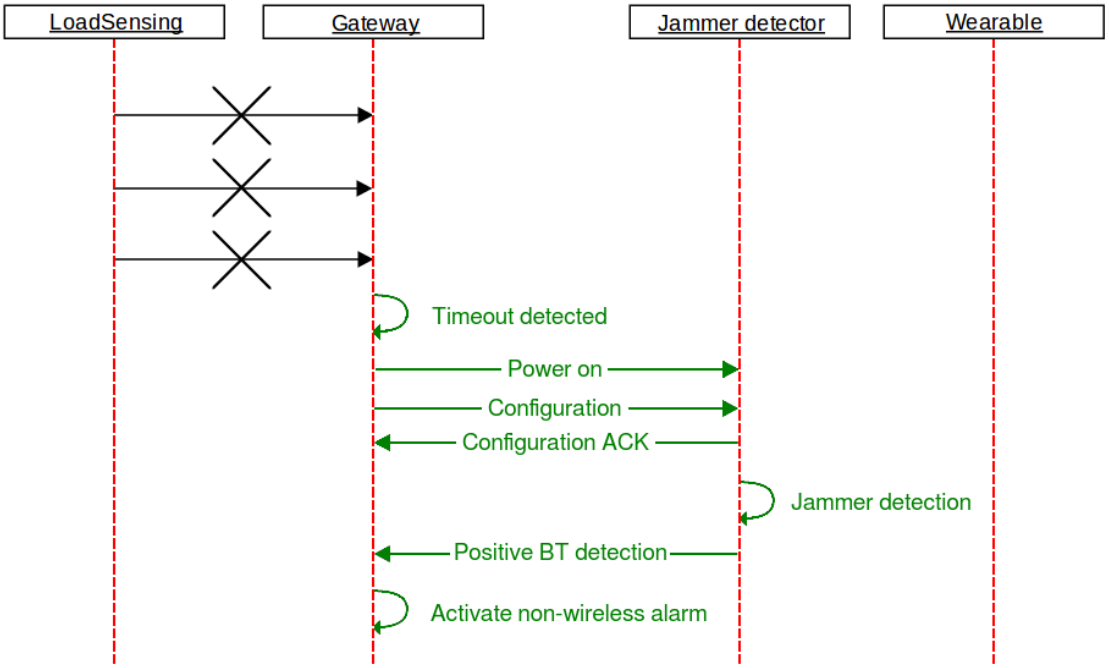

Figure 6: UC1 dataflow 2/4

The third one happens when no messages are received from the LoadSensing, the Jammer detector is powered on and a jammer is detected in the LoRa bandwidth. An alarm is given to the BT wearable.

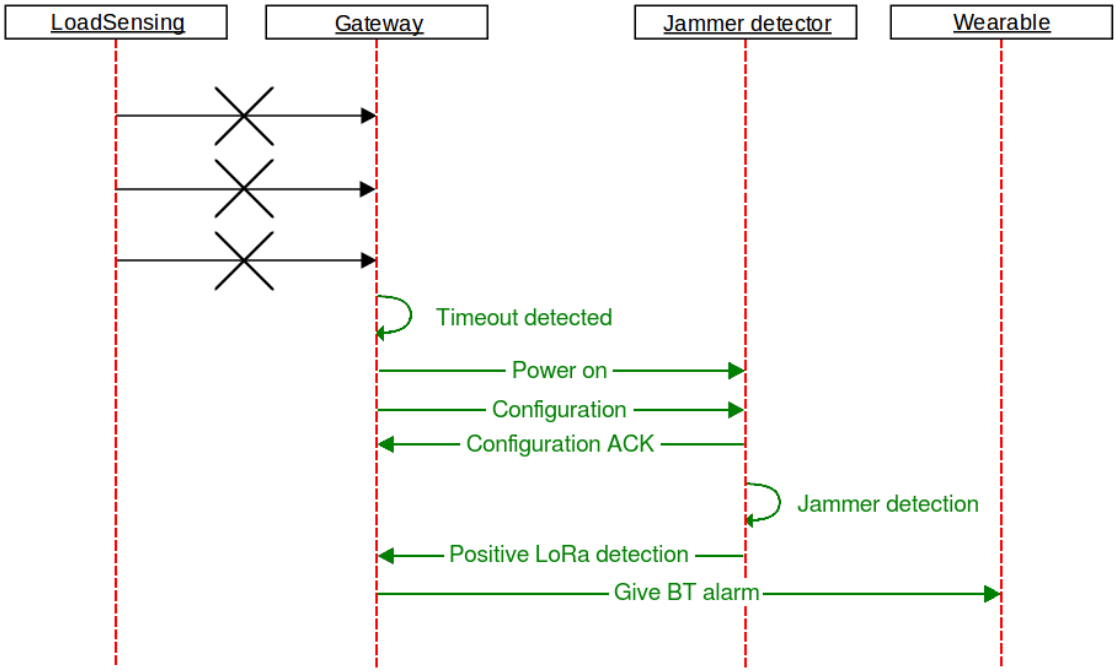

Figure 7: UC1 dataflow 3/4

The last one happens when no messages are received from the LoadSensing, the Jammer detector is powered on and a jammer is detected in the LoRa and BT bandwidth. An alarm is given to the non-wireless alarm.

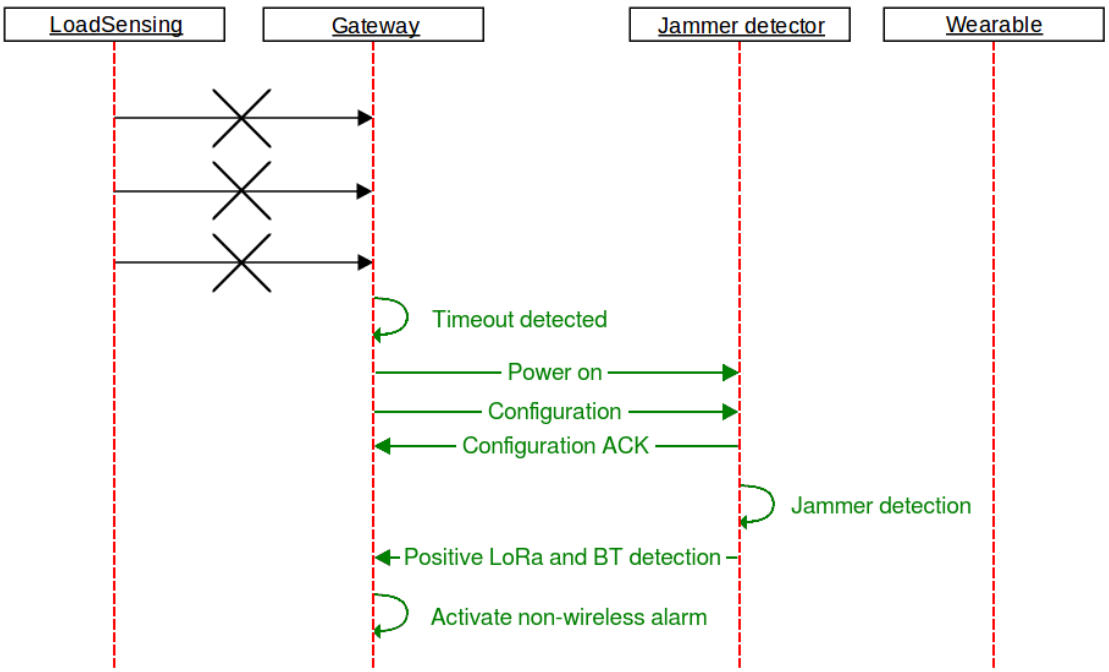

Figure 8: UC1 dataflow 4/4

A2.3 Discussion

The aims of the communications defined above are as follows:

- Reliability, as elements involved in the solution, is locally controlled. Security is improved.
- QoS is improved, as it is independent of other network providers.
- Delay, as all the data flow is faster in a controlled and proprietary network.
- OPEX, as there are no periodical taxes associated.

The security and privacy model for WorldSensing use case is based on the data security levels previously defined in the mF2C project (Public, Protected and Private). Worldsensing use case doesn't deal with private information from users so Private level is not considered in the whole dataflow. Network maps and graphics have been labelled and coloured accordingly.

1. Loadsensing to gateway: Security level Public. The information going through is Public because it is basically environmental data, anyway it has to be minimally protected to assure it can't be modified.
2. Jammer detector to Gateway: Security Level Protected. This information has to be protected because the alert system relies on it. The information has to be available and unmodified.
3. Gateway to wearables: Security Level Protected. This information has to be protected because the alert system relies on it. The information has to be available and unmodified.

Annex 3: Detailed UC2 description

A3.1 UC 2a: Continuous Boat Monitoring

The owner of a vessel can, with the Smart Boat System, remotely monitor the status of the boat through a Smart Boat Application deployed on his/her mobile device (mobile phone, tablet, etc.). Since the sensors on a vessel collect, among others, sensitive personal data (e.g., location) that is routed to the user's device through public networks and the cloud, the data needs to be properly secured. To this end, the Smart Boat System integrates an **End-2-End Encryption mechanism (E2EE), which allows local protection of files so that they can be securely shared among different devices** (among sensors and mobile devices).

As seen below, the vessel sensors continuously send monitoring data about the state of the boat and the environment to the **Sentinel Sensor Hub** (step 1). The **Smart Boat Processor-Server** then collects data from the **Sentinel Sensor Hub** and processes it (step 2). When the data is ready to be uploaded to the Cloud for storage or sent to the user's mobile device, the **E2EE Client 1** deployed on the **Smart Boat Processor-Server** locally encrypts the data and stores the master encryption key locally (step 3). Only the E2EE Client that encrypts the data, and other E2EE Clients, with which the data is shared, can decrypt it. The **Smart Boat Processor-Server** sends a request to the **Smart Boat Proxy** to forward the protected data to the relevant end-point – the cloud or further to the user's device (step 5). The **Smart Boat Proxy** sends the protected data to the **Smart Boat Cloud Service** (step 6), which stores it, processes it further if needed, and forwards it to the user's mobile device (step 7). Finally, the **E2EE Client 2** locally decrypts the protected data on the user's mobile device (step 8).

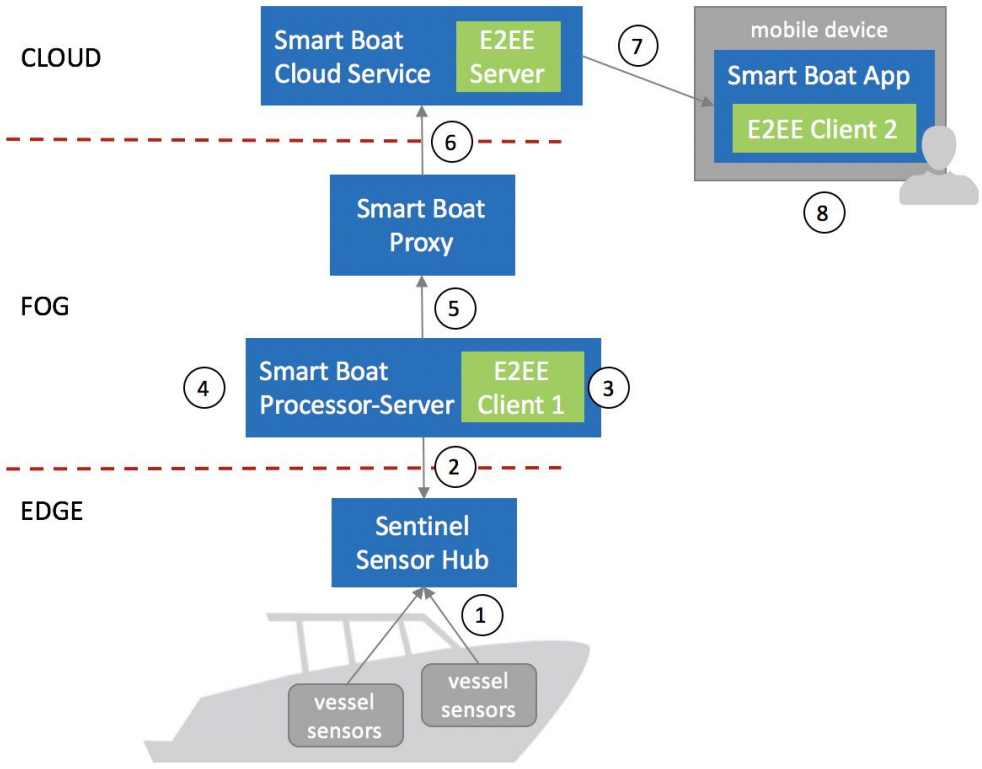

Figure 9: UC2a – overview/architecture

A3.2 UC 2b: Anomaly Detection

The Sentinel devices collect data relevant for the vessel on which they are deployed. It may happen that, at some point, some sensor on the boat or some Sentinel device experiences a failure, or that the Smart Boat Processor-Server, which collects and processes the data from the sensors, experiences a cyber-attack that alters the software or the data. To this end, the Smart Boat System, if possible and if relevant, utilizes data from several vessels in a near surrounding and analyzes it collectively to identify anomalies in the measurements and to provide users with highly accurate data.

Anomaly detection requires sharing data among different vessels, i.e. among different users. The data is shared directly among the vessels through a secure channel and is never routed through the cloud. Thus, the use of the E2EE mechanism is not required. On the other hand, the vessels that exchange the data that is relevant for the anomaly detection (environmental data, for example, wind speed) do not need to exchange any personally identifiable information about the owners of the vessels and their Smart Boat System accounts. The boats that exchange the data can communicate anonymously. To this end, the Smart Boat System integrates a **mechanism that enables anonymous authentication among Smart Boat System devices** (among the Smart Boat Processors-Servers).

1. As seen in Figure 10, the sensors on vessel 1 continuously send monitoring data about the state of the boat and the environment to the Sentinel Sensor Hub 1.
2. The Smart Boat Processor-Server 1 then collects the data from the Sentinel Sensor Hub 1
3. To determine the correctness of the data and to identify anomalies, the Smart Boat Processor-Server 1 checks if there are any vessels with the Smart Boat System in the near surroundings. To this end, the Smart Boat Processor-Server 1 communicates through the Smart Boat Proxy 1 with the Smart Boat Cloud Service, which responds with the IDs of the boats in the near surroundings (for example, the ID of the vessel with the Smart Boat Processor-Server 2).
4. The Smart Boat Processor-Server 1 communicates, through a secure channel, with the Smart Boat Processor-Server 2 through both proxies. The initial communication between both proxies involves mutual authentication to ensure both sides that the request is sent on behalf of a Smart Boat System to a device from the same Smart Boat System. Specifically, the Credential Client 1 deployed on the Smart Boat Server-Processor 1 adds to the request the anonymous credentials (Figure 11) from the first vessel.
5. The Credential Client 2 deployed on the Smart Boat Server-Processor 2 verifies the validity of the credentials with the Credential Verifier deployed in the cloud.
6. Once the credentials of the first vessel are verified, the Smart Boat Processor-Server 2 gathers the requested data from the Sentinel Sensor Hub 2 (and sends it to the first vessel.
7. With this data, the Credential Client 2 from the second vessel attaches its anonymous credentials.
8. The Credential Client 1 verifies them with the Credential Verifier
9. The Smart Boat Server-Processor 1 performs the data analysis.
10. When the data is ready to be uploaded to the Cloud for storage or sent to the user's mobile device, the Smart Boat Processor-Server 1 sends a request to the Smart Boat Proxy 1 to forward the analyzed data (or an alert in case some anomaly has been detected) to the relevant end-point – the cloud or further to the user's device.
11. The Smart Boat Proxy 1 sends the data to the Smart Boat Cloud Service,
12. The SmartBoat Cloud Service stores it, processes it further if needed, and forwards it to the user's mobile device.

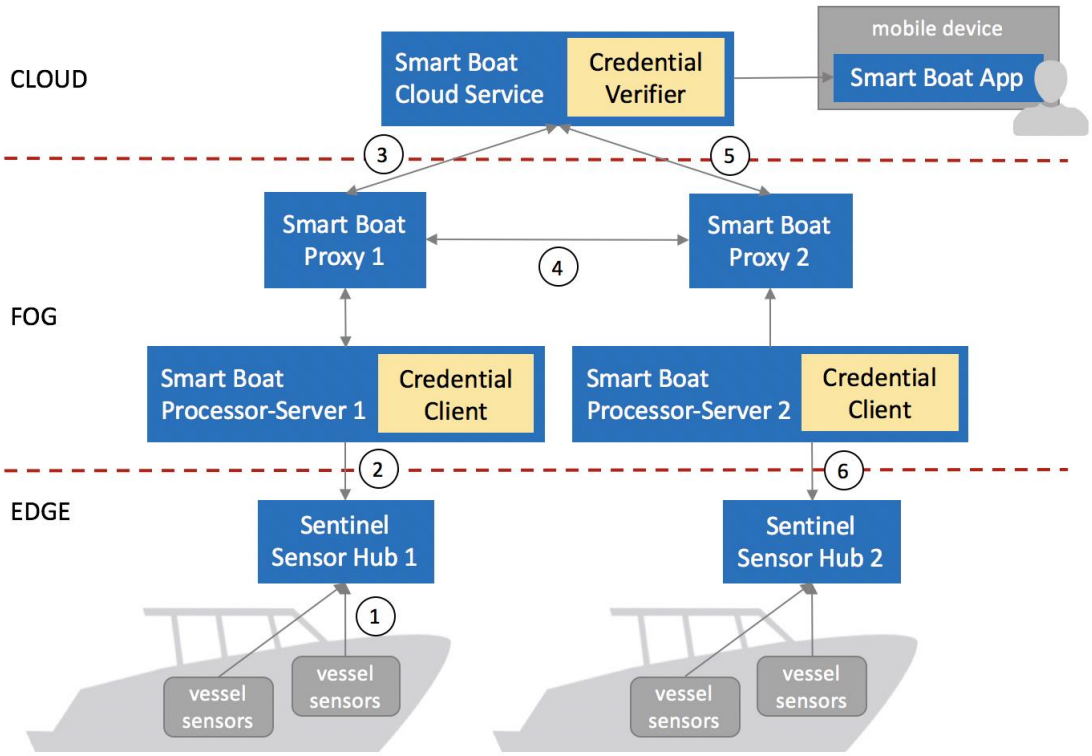

Figure 10: UC2b – anomaly detection

In Figure 11, anonymous credentials are obtained from the Credential Issuer (step 1) on the basis of a certificate obtained from the Certificate Authority (step 2). Both entities are external trusted third parties deployed in the cloud.

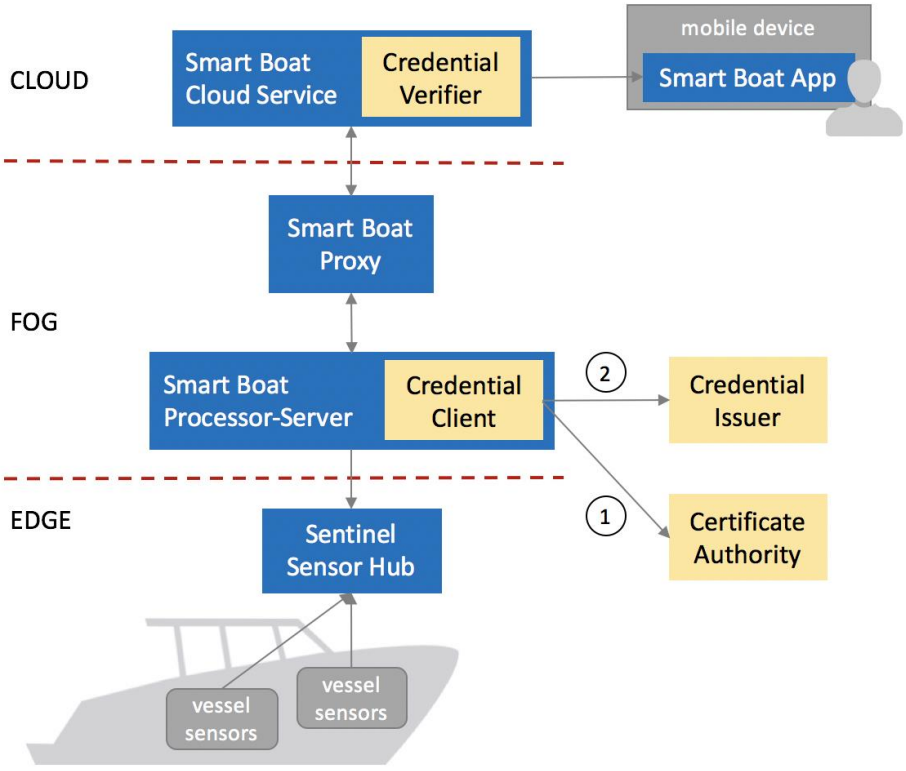

Figure 11: UC2b – obtaining credentials

A3.3 UC 2c: Online Docking and Anchoring Reservation

Many harbors today offer online payments for docking and anchoring permits. With the new European GDPR, it will become more and more important to be able to do that anonymously. To this end, the Smart Boat System integrates a **mechanism that enables anonymous proof of payment** (for the docking and anchorage permit in the harbor).

As seen below, a Smart Boat System user connects to some harbor's online payment web application through his/her **Smart Boat App**, and requests a docking or anchoring permit (step 1). The **Credential Issuer** as part of the online payment system returns the price and the Bitcoin wallet address, to which the user needs to transfer the money. The user transfers anonymized Bitcoins to the provided address (step 2) and once the **Credential Issuer** detects that the payment has been made (step 3), it issues an anonymous credential to the user (step 4). The user transfers the credential to the **Credential Client** deployed on the **Smart Boat Processor-Server** on the boat (step 5). When the vessel enters the harbor and docks, the credential is presented to the harbor registration service (to the **Credential Verifier**) through the **Smart Boat Proxy** (step 6).

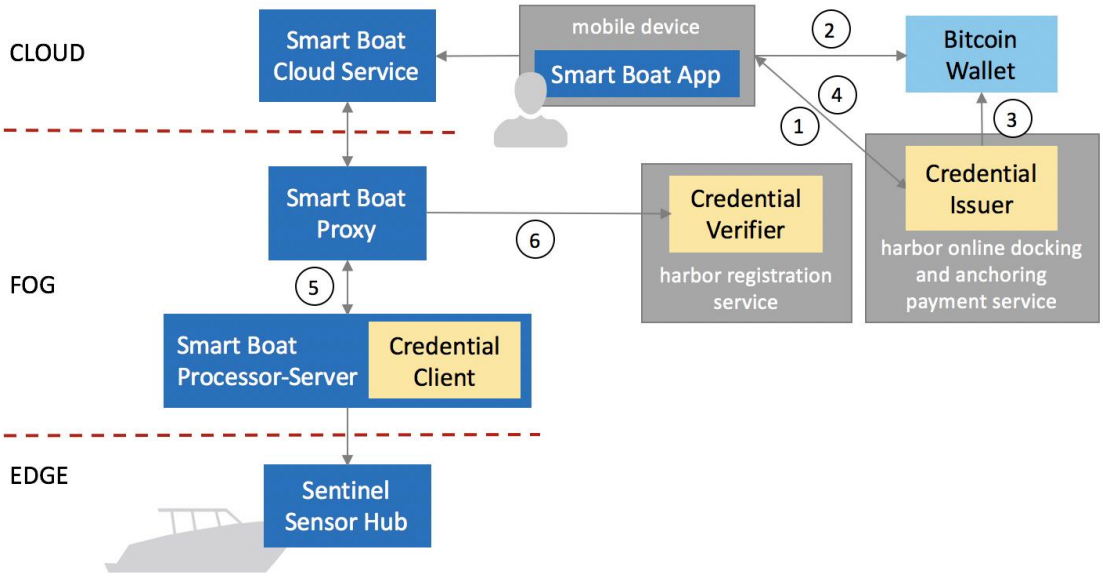

Figure 12: UC2c - architecture

A3.4 UC 2d: Data Plan Sharing

The system on a vessel communicates with the Cloud service either through 3G/4G or through the WiFi. Both are usually not for free. Thus, one of the main (business-wise) advantages a system can have, is an option to automatically detect free WiFi spots or other systems with better connection to the Cloud or cheaper data plan. This way the system on the vessel is always connected to the Cloud via the optimal communication channel (which can mean a channel with the strongest connection or the lowest cost, or some optimum of both – depends on the user's requirements). To this end, the Smart Boat System enables the option of an automatic data plan sharing. In practice, this means that the proxies on the vessels, if this is in the interest of the Smart Boat System owner/user, always monitor the surrounding network. In case another Smart Boat System or a free WiFi spot is detected and data plan sharing is available (i.e., sharing the data plan is in the interest of other system owners), it automatically re-routes all communication through that channel (if this means lower costs and/or better service for the user). In case a vessel requests to share the data plan from another Smart Boat System vessel that is in the near surrounding, it pays for it with the help of a

mechanism that ensures fair exchange of goods (exchange of data plan and money), which means that either both parties receive the requested goods or none of them does.

First, as seen below, each Smart Boat System (i.e., each **fair exchange of goods Client** deployed on the **Smart Boat Proxy**) obtains a certificate from the **Certificate Authority** that proves the validity of the system's data plan. This certificate is accessible to every external Smart Boat System that might request sharing the data plan.

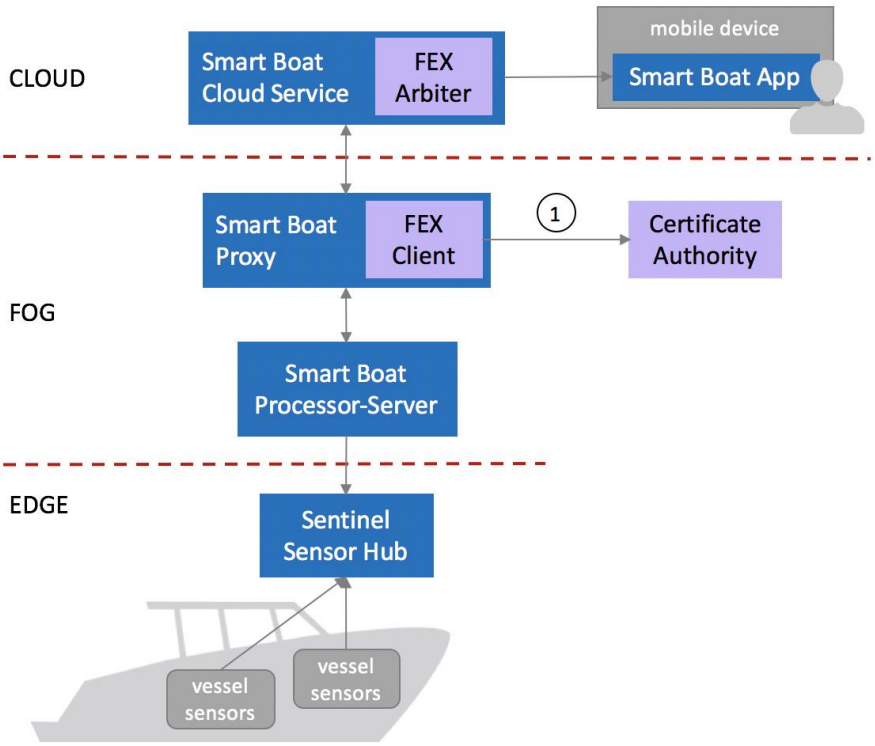

Figure 13: UC2d - obtaining certificate

As seen below, the **Smart Boat Proxy 1** continuously monitors network (step 1) to detect other Smart Boat Systems in the near vicinity or free Wi-Fi spots (for example, available in the harbor). Let's assume that, at some point, the **Smart Boat Proxy 1** detects a network from the second vessel (managed by the **Smart Boat Proxy 2**). The **Smart Boat Proxy 1** checks the quality of the connection, the properties of the data plan, and associated certificate from the second vessel (step 2). We assume that the owner of the second vessel is interested in data plan sharing and thus provides other systems with these details. The **fair exchange Client 1** from the first vessel then verifies the certificate with the **Certificate Authority** (step 3) and determines whether the second vessel has better connection or better data plan that is currently available on the vessel 1 (step 4). For the sake of the example, we assume that the connection/data plan is better on the vessel 2. Therefore, the **Smart Boat Proxy 1** requests data plan sharing from the second vessel, i.e., from the **Smart Boat Proxy 2** (step 5). Both **fair exchange Clients** perform the fair exchange protocol (exchange of electronic money and a guest internet account credentials) with the help of an **Arbiter** (step 6), which results in vessel 1 having the internet connection through the second vessel's network. All data that is gathered by the **Sentinel Sensor Hub 1** and **Smart Boat Processor-Server 1** (step 7) is now sent to the **Smart Boat Cloud Service** through the **Smart Boat Proxy 2** (step 8).

In the diagram below FEX refers to a fair exchange mechanism.

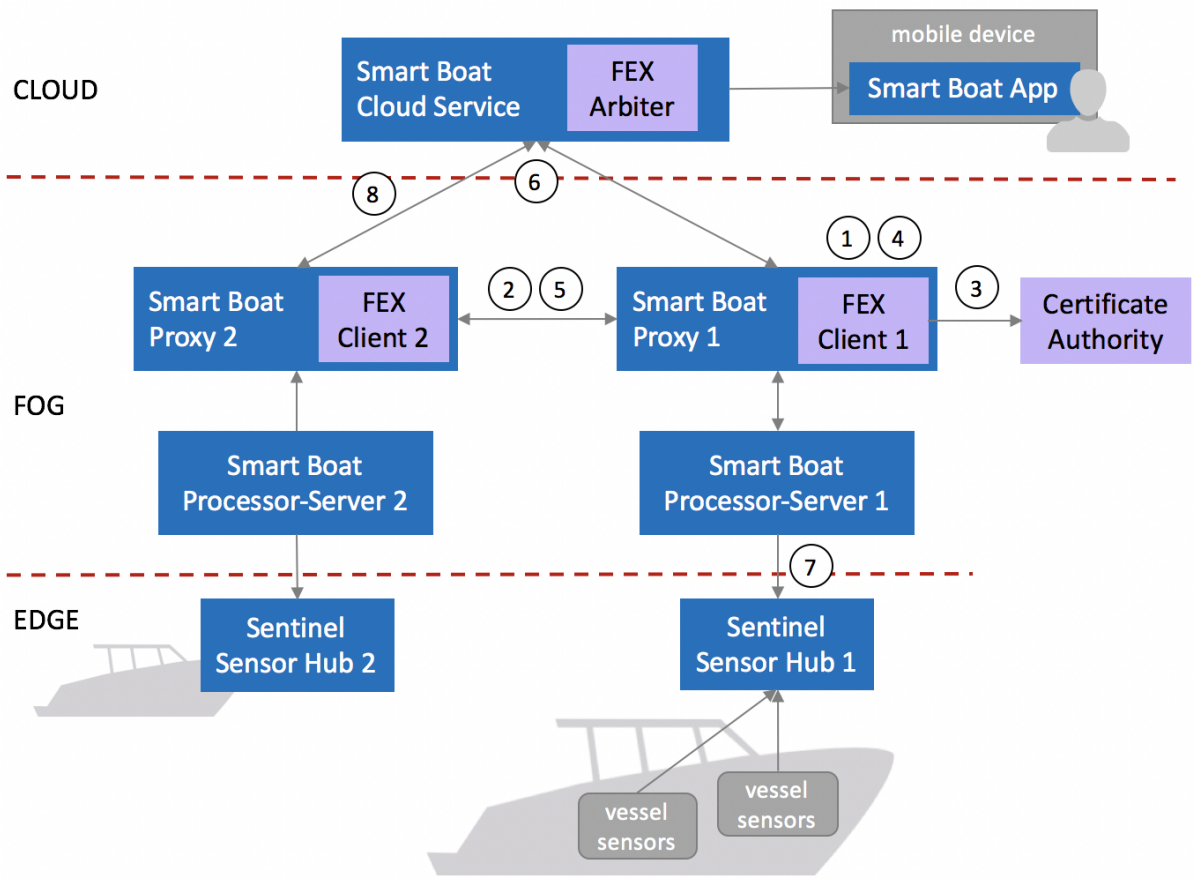

Figure 14: UC2d – fair exchange mechanism

Annex 4: Detailed UC3 description

In the following paragraphs the main business processes of the Use Case 3 (Smart Fog Hub Service – SFHS) are described with details on data-flow and data categorization in terms of Security and Privacy.

A4.1 UC 3a: Object Registration

All objects within the area covered by the SFHS can detect a WiFi seed and an invitation to register to use the available services. The invitation includes information about processing personal data, in order for the user to make an informed decision and consent to the processing.

All users could make a choice between the following:

- Full registration (the user provides its identity information, including an email address and flight information, in this way the user will take advantage of customised information on its own flight, the open desks for check-in and drop-off, boarding gate, etc.), in case we have Security Level **Private** in order to protect personal data
- Anonymous registration (the user chooses a nickname to be admitted in the portal, and get all general information), here we see Security Level **Protected**, as no personal data is transferred and all is needed is data integrity
- No registration (the user can decide to avoid registration, for this reason it will not be allowed to navigate the portal for advices; nonetheless through the "probe request" feature of WiFi it will be tracked and contribute to the statistics related to people clustering and movements), Security level **Public**. The information going through is Public because it is basically environmental data

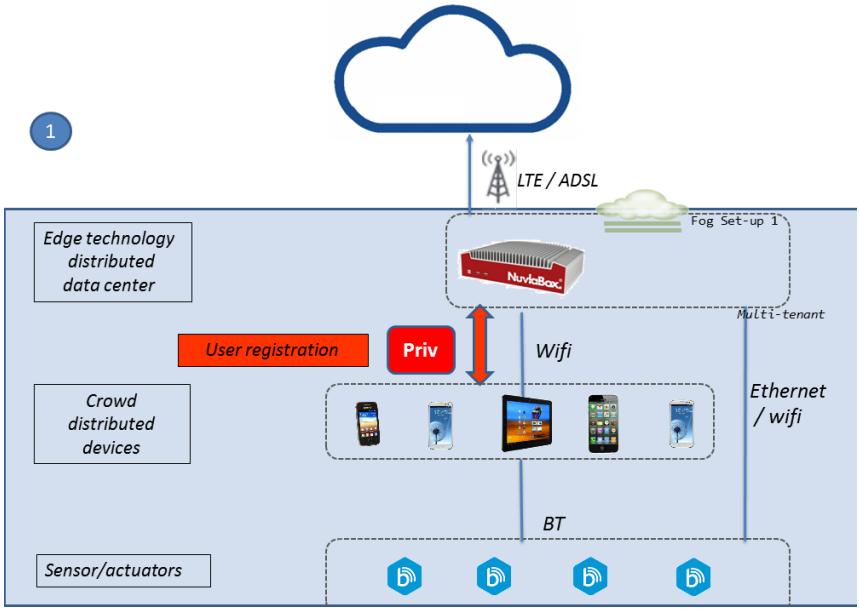

Figure 15: user registration process

A4.2 UC 3b: User Portal Engagement

Once completed the registration process, users will be able to access the services portal, with a menu that includes:

- airport services (only for full registered users), Security Level here is **Private**
- shopping area (all registered users), even if identity is not disclosed, since user behavior is still private data, it would be recommended to use **Private** as Security Level

- entertainment (all registered users), even if identity is not disclosed, since user behavior is still private data, it would be recommended to use **Private** as Security Level
- other services (all registered users), even if identity is not disclosed, since user behavior is still private data, it would be recommended to use **Private** as Security Level

Some of these services can include some booking of messages to be notified of some events (e.g. go to boarding gate, for full registered users).

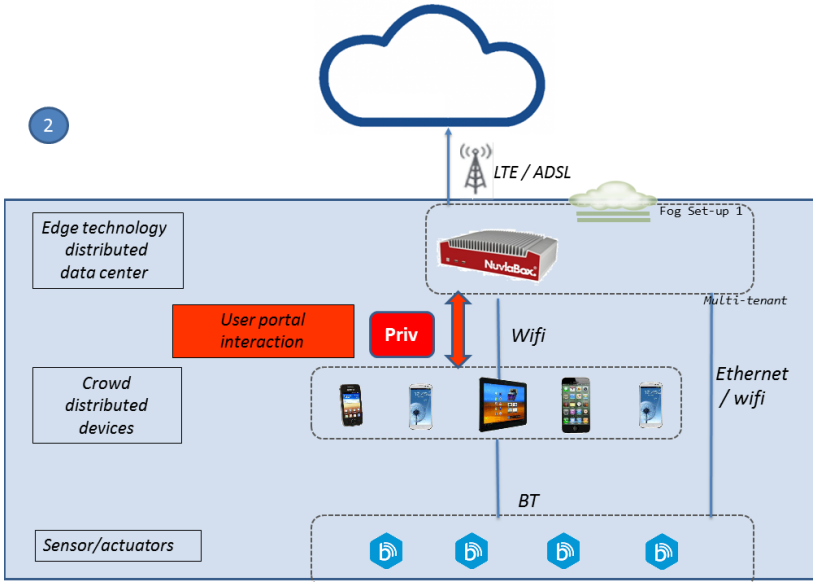

Figure 16: user portal engagement

A4.3 UC 3c: Beacon Announce

The ways of engagement with objects within the field of the SFHS include advertisements push messages sent by BluetoothLE iBeacons that are positioned in proximity of particular shops or other point of interest. This kind of messages can be used by all registered and non-registered users.

Since all information here is commercial, Security Level here is **Public.**

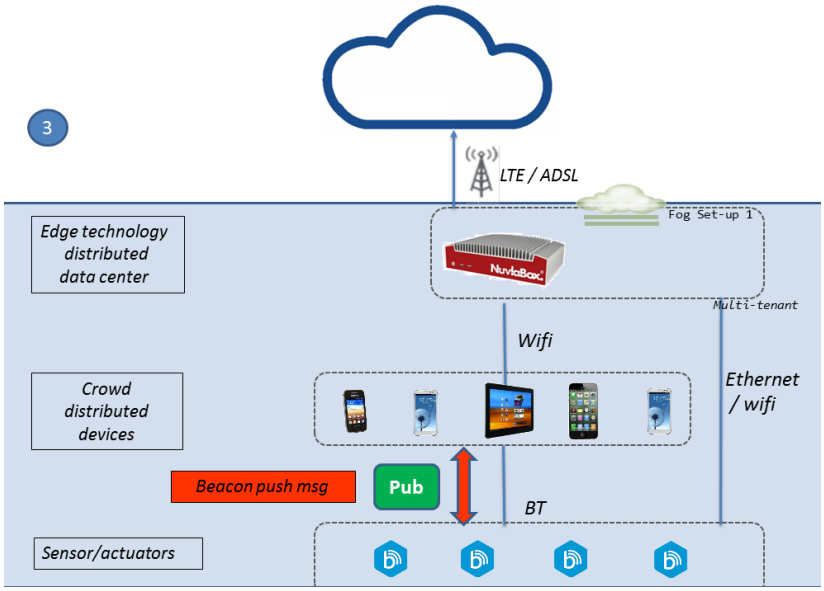

Figure 17: beacon advertising message

A4.4 UC 3d: Continuous Objects Monitoring & Position Tracking

In the Fog layer, the smart agent continuously monitors all objects in the field, including location, according to the data sharing permissions it has received. All resulting data can be displayed in different maps, both in terms of single objects and related paths, or spatial distribution and clustering of objects, with chance to generate several kinds of statistics in terms of:

- real-time maps of objects in the field
- How many unique objects have been detected in the field, and how many present currently
- Split of objects by registration type, device type, OS, processing category, etc.

Even if identity is not disclosed, since position and user behavior is still private data, it would be recommended to use **Private** as Security Level.

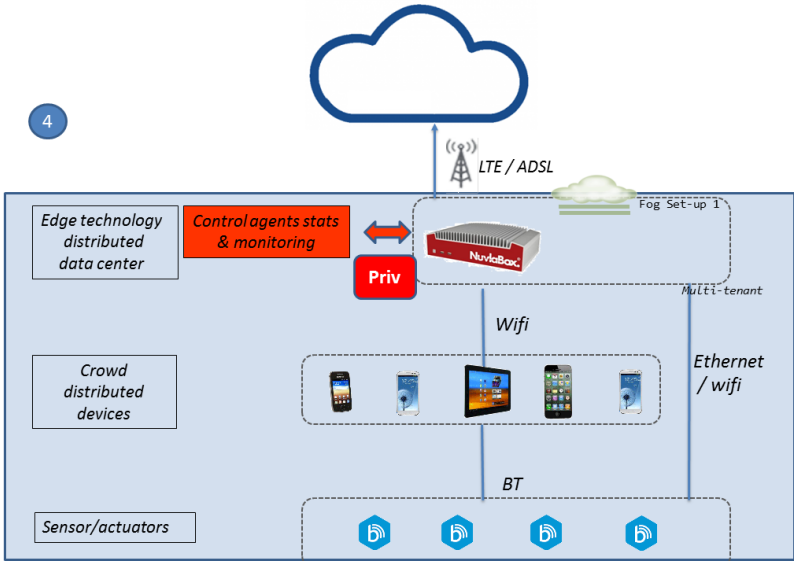

Figure 18: objects monitoring & tracking

A4.5 UC 3e: Accounting & Forecast on Movement/Behaviour

At Fog layer, the smart agent performs machine learning tasks to forecast behaviours and suggestions for users, and when needed it moves some processing to the cloud. All predictive tasks will work on collected data that are to be made anonymous or pseudo-anonymous before processing.

Different tasks will be available including:

- Determine the most used services
- Identification of recurrent behaviour and walking paths for passengers (the general dynamics of movements of passengers in transit)
- Calculate the effectiveness of push messages sent by beacons, both in terms of “interest” (visits to specific shops), or purchase of goods or services
- Determine forecasts and recommendations to be proposed to registered users through the service portal or through beacons

All information and results from previous tasks could suggest further prediction models and spot new operational and business models.

Even if identity is not disclosed, since position and user behavior is still private data, it would be recommended to use **Private** as Security Level.

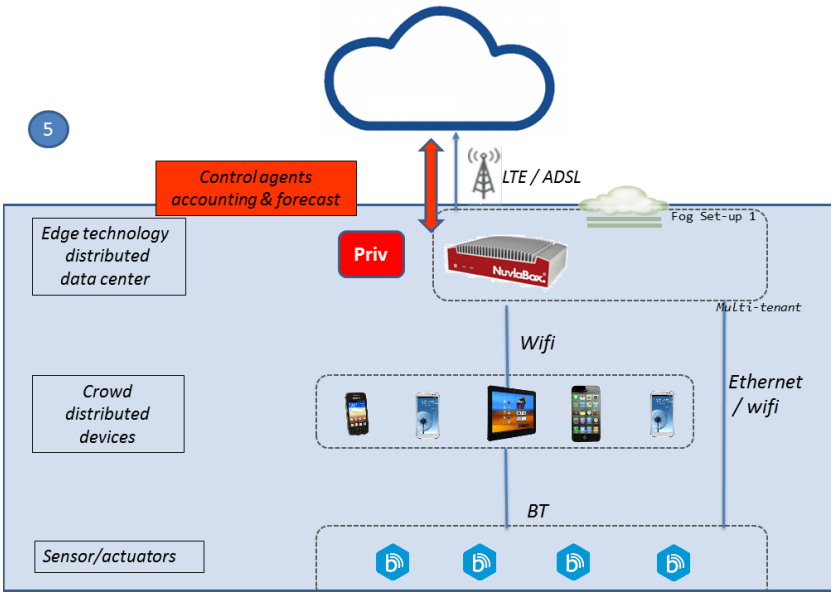

Figure 19: accounting & forecast