FP7-SMARTCITIES-2013
Project number: 609062
http://www.smartie-project.eu/

SMARTIE

Deliverable D4.2

Event Processing and Framework Specification for Privacy and Access Control

Editor:	UMU
Dissemination level: (Confidentiality)	PU
Suggested readers:	Consortium/Experts/other reader groups
Version:	1.0
Total number of pages:	52
Keywords:	Event Processing, Framework, Privacy, Access Control

Abstract

Previous deliverables (D2.2, D3.1 and D4.1)analysed the security requirements of IoT ecosystems and defined the individual components proposed in SMARTIE. Moreover, D2.3presented the integration of all components in combination with the Smartdata Platform to establish the initial architecture.This deliverable describes the three mainframeworks and techniquesto enable Event Processing, Privacy and Access Control in IoTreal world scenarios.

Impressum

[Full project title] Secure and sMArterciTIes data management

[Short project title] SMARTIE

[Number and title of work-package] WP4 Secure information distribution & access control
[Document title] Event Processing and Framewok Specification for Privacy and Access Control
[Editor: Name, company]Antonio Skarmeta, UMU

[Work-package leader: Name, company] Antonio Skarmeta, UMU

Executive Summary

This deliverable is based on the work done in previous deliverables such as D2.2, D3.1, D4.1 and D2.3. First, D2.2 analysed the security requirements of IoT ecosystems. Later, the deliverables D3.1 and D4.1 defined the individual components proposed in SMARTIE to accomplish secure information gathering, storage, access control and preserve the data privacy. D2.3 presented the integration of all components in combination with the Smartdata Platform in order to describe the initial architecture of SMARTIE and the main interactions among the components.This deliverable goes one step further in relation to the previous deliverables. This deliverable describes in detail the 3 main frameworks of the SMARTIE project to provide Event Processing, Privacy and Access Control in IoT ecosystems.

According to the DOW, this deliverable describes the solution to allow service providers to gather the minimum essential knowledge (appropriately extracted from protected data) needed to offer an enhanced user experience on its interactions with IoT devices. The objective is to define primitives to extract complex features from encrypted users' data, and correlate two encrypted data streams for event detection. Due to the fact that searchable encryption and private set intersection protocols are very focused on single user setting and only support equal operations, this deliverable defines new primitives like less than/greater than of searchable encryption.

Additionally this deliverable defines fine-grained access control for privacy-sensitive data, providing tools for allowing a user-centric approach enabling access policy for personal information and appling distinct access policies in different situations. Also the integration of data minimization techniques to control the level and exposition of certain attributes and/or data generated by IoT devices are envisaged. Moreover, the deliverable proposes an improved protocol of Attribute-Based Encryption (ABE) schemes for fine-grained access control without a lengthy user authorization process and its integration with minimal disclosure technologies in order to provide an efficient framework for the IoT community of services. This deliverable also presents how content-centric security can be applied to data and information to provide end-to-end security, but it minimises the exposure of such data where it needs to be searched and/or processed in the context of IoT communications.

List of authors

Company	Author
UMU	Antonio Skarmeta, Jose Luis Hernandez Ramos, Alfredo Quesada, Rafael Marín Pérez, Dan García, Miguel Angel Zamora
NEC	Jens-Matthias Bohli, Felix Gomez-Marmol
PTIN	Fábio Gonçalves, Rui Pedro Bernardino, Luis Cortesão

Table of Contents

List of Figures

Abbreviations

ARM	Architecture Reference Model
CoAP	Constrained Application Protocol
HTTP	Hypertext Transfer Protocol
XML	Extensible Mark-up Language
IoT	Internet of Things
IoT-A	Internet of Things Architecture
DTLS	Datagram Transport Layer Security
UDP	User Datagram Protocol
JSON	JavaScript Object Notation
6LowPAN	IPv6 over Low power Wireless Personal Area Networks
IEEE	Institute of Electrical and Electronics Engineers
WPAN	Wireless Personal Area Network
REST	Representational State Transfer
ACE	Authentication and Authorization for Constrained Environments
WG	Working Group
IETF	Internet Engineering Task Force
CoRE	Constrained RESTful Environments
RADIUS	Remote Authentication Dial-In User Service
XACML	eXtensible Access Control Markup Language
EAP	Extensible Authentication Protocol
PSK	Pre-Shared Key
ECC	Elliptic curve cryptography
HIP	Host Identity Protocol
RBAC	Role-based access control
PANA	Protocol for Carrying Authentication for Network Access
ABAC	Attribute-based access control
IKE	Internet key exchange
H2T	Human-to-Thing
T2T	Human-to-Human

1 Introduction

Recent advances in wireless communications and pervasive computing are driving the constant development of the so-called Internet of Things (IoT)[1], providing ubiquity and intelligence to our surrounding environment. IoT represents the extension of Information Technology (IT) to all areas of our lives, transforming current isolated networks and infrastructures into a global network of interconnected heterogeneous objects as a key enabler of the future Big Data era [2]. In fact, the increasing interest on IoT from academia and industry is promoting the emergence of innovative services to be leveraged by societies, and enabling unprecedented economic and social opportunities for governmental and private organizations in the envisioned Smart Cities ecosystems [3].

The two important features Smart Cities need to be enforced in order to be acceptable by the population are the security and privacy. In this document we address these two issues from the point of view of 3 frameworks that are proposed to achieve these goals. Moreover, we present a real application scenario for each framework that exemplifies their use in the real world.

- Privacy-Preserving Event Processing Framework

  We explore techniques how to process and detect certain events from encrypted data. The encryption schemes need to have properties that enable the processing and event detection. We describe our experience with encryption schemes from the literature and propose a framework for the privacy-preserving processing of location data in a geofencing application.

- Access Control Framework

  We propose a complete framework that will cover the lifecycle of the things to communicate and exchange information in a secure way providing a way of delivering information subject to proper authorization.

  This process will cover the bootstrapping of cryptographic information required to establish secure end-to-end communication. The generation of authorization tokens that will allow users to obtain the authorization needed to gather the queried information, and the process by which the user is authorized to obtain that token.

- Privacy Framework

  We propose a framework by which a user can express interest in certain kind of information, subscribe to a producer of that information and receive that information in a secure way preserving the user privacy in the process.

  This process will explain the crypto primitives used for this purpose and the way the information can be obtained.

1.1 Relation to S&T Objectives

This deliverable addresses the objectives of understanding requirements for data and application security and creating apolicy-enabled framework supporting data sharing across applications and developing new technologies that establish trust and security in the perception layer and network layer. This is accomplished by introducing frameworks that enable event processing, privacy and access control.
In addition, the deliverable presents new technologies for information retrieval and processing guided by access control policies in the application layer. The framework for access control enables the user to retrieve information from the sensors after the correspondent authorization.

1.2 Beyond State-of-the-Art Contributions

In this deliverable we go beyond the state of the art and deploy current protocols that are envisioned to be used in IoT environments such as CoAP, DTLS, etc. Moreover weextend currentframeworks that enable the exchange of information in a secure manner. In particular, the crypto systems allow the dissemination of information in IoTapplications preserving the privacy of the entities involved in the information exchange.

2 Event Processing Framework

2.1 Privacy preserving event detection and correlation

2.1.1 Technical requirements for clustering encrypted data

As stated in Section 3.1.3 of D4.1 [86], one type of data correlation we are thinking of consists of grouping the encrypted data given certain criteria, in order to build self-contained and meaningful so-called clusters of encrypted events.

To this end, we analysed one of the most popular clustering algorithms, namely, k-Means. k-Means clustering aims to partition *n* instances into *k* clusters in which each observation belongs to the cluster with the nearest mean, named centroid of the cluster.

In other words, given a set of *n*instances $(x_1, x_2, \ldots, x_n)$, where each instance is a *d*-dimensional real vector, *k*-means clustering aims to partition the *n* instances into $k(\leq n)$ sets $S = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the within-cluster sum of squares (WCSS). In other words, its objective is to find:

$$\arg\min_S \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2$$

where μ_i is the mean of points, or centroid, of S_i.

Next we present the most common structure for the k-Means algorithm, also known as Lloyd's algorithm where we can distinguish three differentiated phases or steps: initialization, assignment and update.

1. **Initialization phase**

Given an initial set of *k* means (or centroids) $c_1, c_2, \ldots, c_k$

2. **Assignment step**

Assign each instance x_i $(1 \leq i \leq n)$ to the cluster S_j whose centroid μ_j is closest according to the Euclidean distance $ed(x_i, \mu_j)$

3. **Update step**

Calculate the new means μ'_j to be the centroids of the instances in the new clusters

After the initialization phase, the assignment and update ones repeat iteratively and the algorithm converges when the assignments no longer change.

Applying such algorithm over the plaintext domain is easy. The challenge comes when the instances $(x_1, x_2, \ldots, x_n)$ are actually **encrypted***d*-dimensional real vectors. To achieve that goal, we analysed the fundamental features required by the clustering system which, in turn, will be the features required by the encryption scheme used to cypher the instances to be clustered.

First, we see that the system would need to compute the distance between two encrypted instances $D(E(x_i), E(\mu_j))$. However, to be more specific, the only information that is really required by the clustering system would be whether an encrypted distance $D'(E(x_1), E(\mu_j))$ is greater, equal or less than another given encrypted distance $D'(E(x_2), E(\mu_j))$.

On the other hand, the clustering system should be able to compute the centroid of a given set of instances $(x_1, x_2, \ldots, x_n)$.

Once we have identified the fundamental properties required by the clustering system, the next step will be to develop a novel one fulfilling such requirements. We have already initiated such path and we believe that a smart combination of multi-party computation techniques and homomorphic encryption solutions might be the key to reach such goal.

2.1.2 K-Means clustering over encrypted data: an example solution

After a thorough survey of the state-of-the-art in the field of clustering over encrypted data, next we present a representative recent solution in this direction [87]:

In this solution a cryptographic key $K(m)$ is a list $[(k_1, s_1, t_1), \ldots, (k_m, s_m, t_m)]$, where k_i, s_i and t_i are real numbers. This scheme requires $m \geq 3$, $k_i \neq 0 \; (1 \leq i \leq m-1)$, $k_m + s_m + t_m \neq 0$ and only one i $(1 \leq i \leq m-1)$ such that $t_i \neq 0$. Given the key $K(m)$, the algorithm Enc encrypts a real number v into the ciphertext $(e_1, e_2, \ldots, e_m)$, denoted by $\mathrm{Enc}(K(m), v) = (e_1, e_2, \ldots, e_m)$, where e_i is called a subciphertext.

The algorithm Enc is defined by the following steps:

- Uniformly sample m random real numbers $r_1, \ldots, r_m$ which can be arbitrarily large
- Compute $e_1 = k_1 * t_1 * v + s_1 * r_m + k_1 * (r_1 - r_{m-1})$
- Compute $e_i = k_i * t_i * v + s_i * r_m + k_i * (r_i - r_{i-1})$ for $2 \leq i \leq m-1$
- Compute $e_m = (k_m + s_m + t_m) * r_m$

Given a ciphertext $(e_1, e_2, \ldots, e_m)$ and the key $K(m)$, the decryption algorithm Dec returns the plaintext v by the following steps. The decryption operation is denoted as $\mathrm{Dec}(K(m), (e_1, e_2, \ldots, e_m)) = v$ with

$$T = \sum_{i=1}^{m=1} t_i$$

$$S = \frac{e_m}{k_m + s_m + t_m} = r_m$$

$$v = \frac{\sum_{i=1}^{m-1} \frac{e_i - S * s_i}{k_i}}{T}$$

As we can observe, such encryption scheme supports both addition and multiplication over ciphertexts. In other words,

$$\mathrm{Dec}(K(m), (e_1 + e'_1, e_2 + e'_2, \ldots, e_m + e'_m)) = v + v' \text{ and } \mathrm{Dec}(K(m), (e_1 * v', e_2 * v', \ldots, e_m * v')) = v * v'.$$

Given an instance x_i and a centroid c_t, each of which have d dimensions (i.e. $x_i = (x_{i1}, \ldots, x_{id})$ and $c_t = (c_{t1}, \ldots, c_{td})$), the squared distance between x_i and c_t is computed as $D^2(x_i, c_t) = \sum_{j=1}^{d} (x_{ij} - c_{tj})^2$. Assuming $\lambda \in [1, m-1]$ is the index where $t_\lambda \neq 0$ within the key $K(m)$, and being $\mathrm{Enc}(K(m), x_{ij}) = (e_{ij1}, \ldots, e_{ijm})$ and $\mathrm{Enc}(K(m), c_{tj}) = (e'_{tj1}, \ldots, e'_{tjm})$, then the encrypted squared distance between x_i and c_t is computed as

$$\begin{aligned} ED^2(x_i, c_t) &= \sum_{j=1}^{d} (e_{ij\lambda} - e'_{tj\lambda})^2 \\ &= \sum_{j=1}^{d} \left(k_\lambda * t_\lambda * (x_{ij} - c_{tj}) + (NX_{ij} - NC_{tj})\right)^2 \\ &= (k_\lambda * t_\lambda)^2 * D^2(x_i, c_t) + 2 * k_\lambda * t_\lambda * \sum_{j=1}^{d} (x_{ij} - c_{tj}) * (NX_{ij} - NC_{tj}) + \sum_{j=1}^{d} (NX_{ij} - NC_{tj})^2 \end{aligned}$$

where$NX_{ij} = s_\lambda * r_{ijm} + k_\lambda * \left(r_{ij1} - r_{ij(m-1)}\right)$

$NC_{tj} = s_\lambda * r'_{tjm} + k_\lambda * (r'_{tj1} - r'_{tj(m-1)})$

However, in order to convert this encrypted distance $ED^2(x_i, c_t)$ into an order-preserving distance $OED^2(x_i, c_t)$, authors propose to add a trapdoor; that is to say $OED^2(x_i, c_t) = ED^2(x_i, c_t) + Trap_{it}$. Such trapdoor $Trap_{it}$ is computed as

$$Trap_{it} = -2 * k_\lambda * t_\lambda * \sum_{j=1}^{d} \left(x_{ij} - c_{tj}\right) * \left(NX_{ij} - NC_{tj}\right) + \sum_{j=1}^{d} \left(NX_{ij} - NC_{tj}\right)^2$$

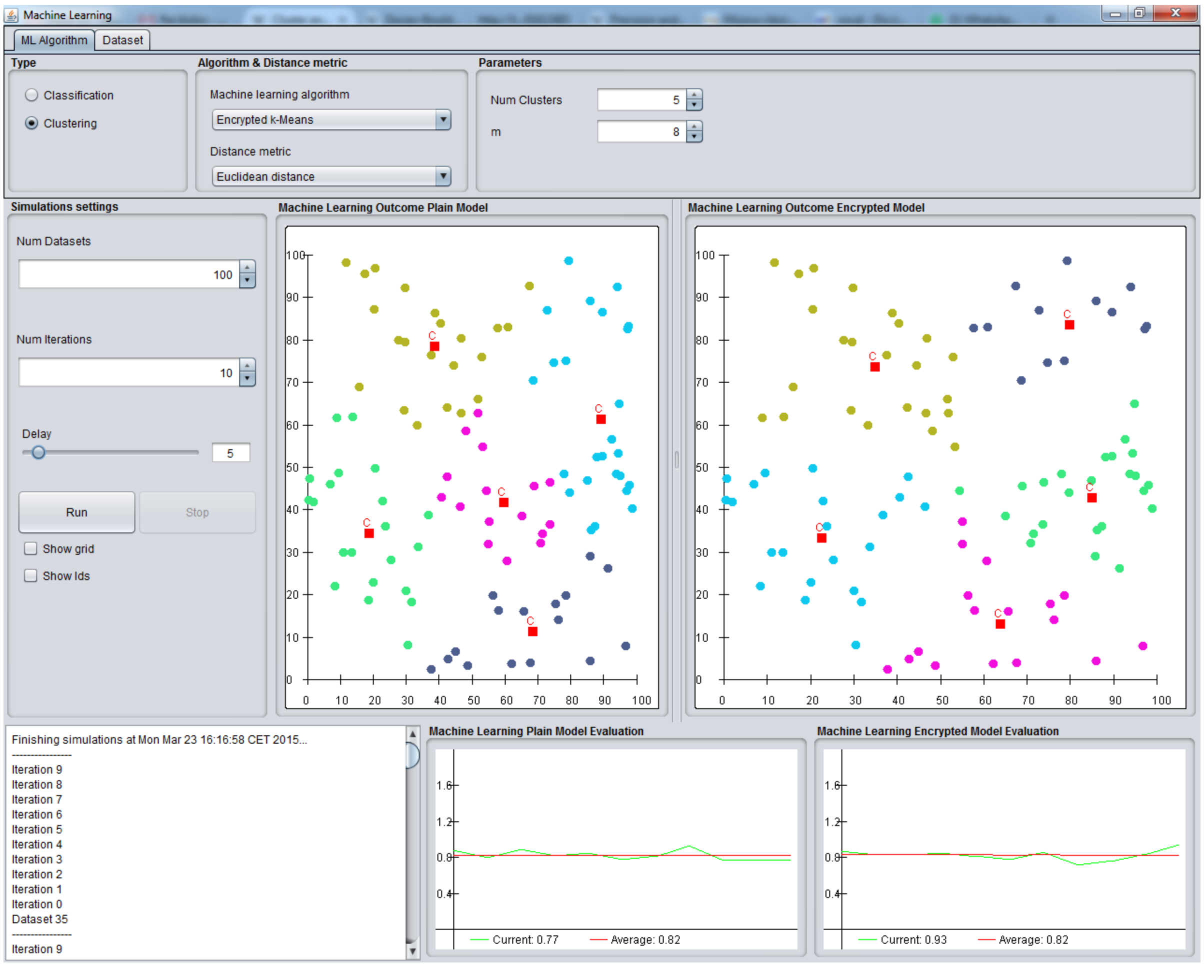
By doing so, this algorithm would be able to determine whether an encrypted distance$D'\big(E(x_1), E(\mu_t)\big) = OED^2(x_1, c_t)$ is greater, equal or less than another given encrypted distance $D'\big(E(x_2), E(\mu_t)\big) = OED^2(x_2, c_t)$ and, therefore, able to apply the k-Means clustering algorithm.

2.1.3 Security Consideration

The scheme as it is described in [87] comes without a security analysis. While indeed the correctness of the scheme is easy to check, this is harder for its security. The system relies on the hardness of solving multivariate quadratic equation systems. While this problem is NP-hard in general, the suggested parameters are not derived from any hardness assumption and therefore appear to be too small (see e.g. [88]). Thus, a detailed analysis with optimised parameters needs to be performed. This will however affect negatively also the performance of the scheme.

2.1.4 Machine learning algorithms over encrypted data simulator

With the aim of providing a framework to easily develop and compare new solutions devoted to correlate encrypted data, we designed and implemented a Java simulator as shown in Figure 1.


Machine Learning

ML Algorithm | Dataset

Type
- Classification
- [✓ Clustering]

Algorithm & Distance metric
- Machine learning algorithm: Encrypted k-Means
- Distance metric: Euclidean distance

Parameters
- Num Clusters: 5
- m: 8

Simulations settings
- Num Datasets: 100
- Num Iterations: 10
- Delay: 5
- Show grid
- Show Ids

```
Finishing simulations at Mon Mar 23 16:16:58 CET 2015...
-----------------
Iteration 9
Iteration 8
Iteration 7
Iteration 6
Iteration 5
Iteration 4
Iteration 3
Iteration 2
Iteration 1
Iteration 0
Dataset 35
-----------------
Iteration 9
```

Run
Stop
Machine Learning Outcome Plain Model
Machine Learning Outcome Encrypted Model
Machine Learning Plain Model Evaluation
Current: 0.77
Average: 0.82
Machine Learning Encrypted Model Evaluation
Current: 0.93
Average: 0.82

Figure 1: Simulator to test solutions to correlate encrypted data. Overall overview.

Next we will explain each of the panels and options of the simulator. In this regard, Figure 2 shows the panel that allows us to select which particular machine learning algorithm (either classification or clustering) we want to simulate, as well as which distance metric we want to use for the selected algorithm. Furthermore, we can also set some parameters specific for the selected algorithm.

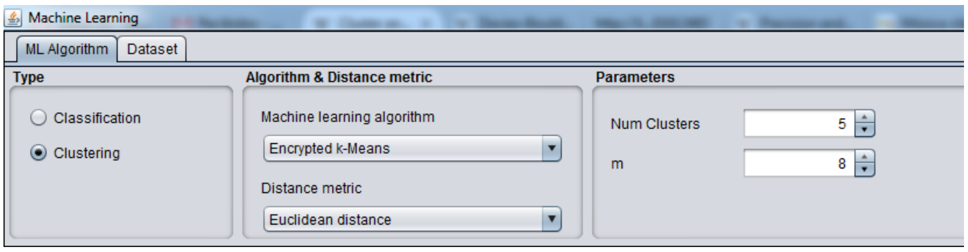
In the current version of the simulator, only the k-Means algorithm and the model presented in previous section [87] (named "Encrypted k-Means") are implemented. However, the simulator is ready to easily implement the following additional machine learning algorithms:

- Decision Tree
- k-NN (k-Nearest Neighbors)
- Neural Network
- Perceptron
- SVM (Support Vector Machine)

As for the distance metrics, the current version of the simulator implements the following ones (although new ones can be easily integrated):

- Euclidean distance
  - $d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$

- Manhattan distance
  - $d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$
- Norm distance
  - $d(x, y) = \sqrt[p]{\sum_{i=1}^{n} (x_i - y_i)^p}$


Machine Learning

ML Algorithm | Dataset

Type
- Classification
- [✓ Clustering]

Algorithm & Distance metric
- Machine learning algorithm: Encrypted k-Means
- Distance metric: Euclidean distance

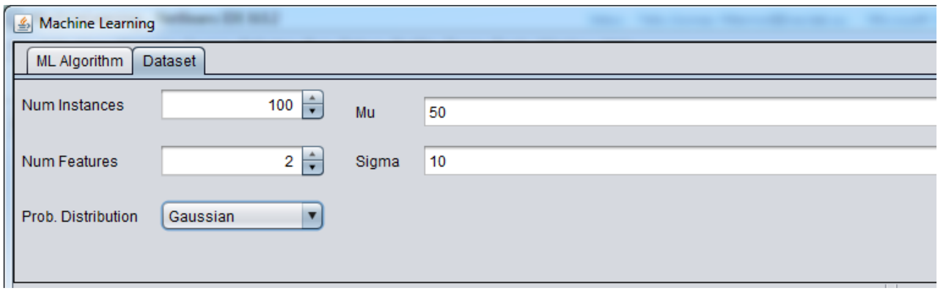
Parameters
- Num Clusters: 5
- m: 8

Figure 2: Simulator to test solutions to correlate encrypted data. Machine learning algorithm selection and settings panel.

In turn, Figure 3 shows the panel where to specify the properties of the dataset used to test the simulated machine learning algorithms. Thus for example, it is possible to select the number of instances within the dataset, as well as the number of (real) features. Note that if the instances have more than 2 features, only the first two will be used to plot them.

Additionally, it is possible to select the probability distribution of the dataset, as well as any associated parameters (for example, μ and σ for the Gaussian distribution). In the current version of the simulator, the following probability distribution functions are implemented:

- Random distribution
  - $x_i \xleftarrow{Random} [0,100]$
- Gaussian distribution
  - $x_i \xleftarrow{Random} [0,100] \text{ iff } \left(r \xleftarrow{Random} [0,1]\right) \geq \frac{e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}$


Machine Learning

ML Algorithm | Dataset

- Num Instances: 100
- Num Features: 2
- Prob. Distribution: Gaussian
- Mu: 50
- Sigma: 10

Figure 3: Simulator to test solutions to correlate encrypted data. Dataset settings panel.

Next, in Figure 4 the specific settings for the simulations themselves are shown. Here we can determine the number of datasets we want the simulator to test, as well as the maximum number of iterations of the selected machine learning algorithm.

It is also possible to set a small delay to the execution of the algorithm in order to better watch the evolution of such algorithm. Finally, a couple of buttons to run and stop the simulations, respectively, as well as two checkboxes to show or hide the ids of the instances and a grid are offered.

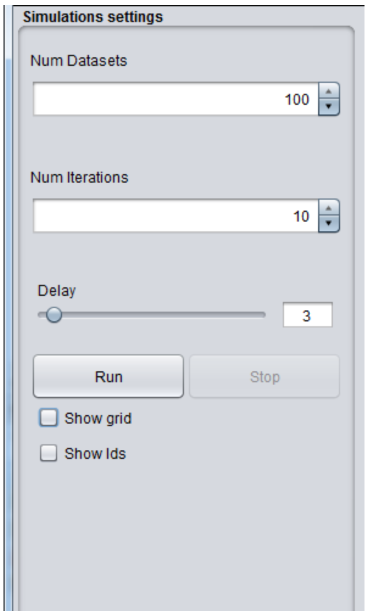
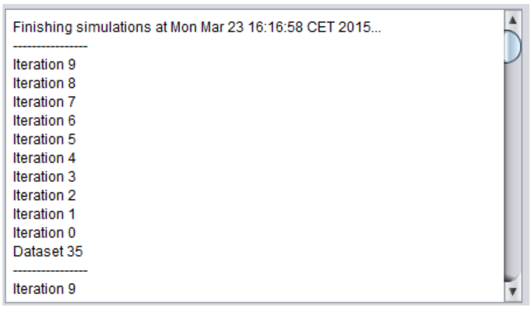


Figure 4: Simulator to test solutions to correlate encrypted data. Simulations settings panel.

A panel showing important or relevant log messages of the simulator is shown in Figure 5. It indicates, for example, the number of iterations already executed in a given simulation, as well as the date and time when the simulations start and finish, amongst other information.



```
Finishing simulations at Mon Mar 23 16:16:58 CET 2015...
----------------
Iteration 9
Iteration 8
Iteration 7
Iteration 6
Iteration 5
Iteration 4
Iteration 3
Iteration 2
Iteration 1
Iteration 0
Dataset 35
----------------
Iteration 9
```

Figure 5: Simulator to test solutions to correlate encrypted data. Log messages panel.

As we can observe in Figure 6, the model outputted by the selected machine learning algorithm is depicted in a twofold panel. On the left-hand side, we observe the models gotten from those algorithms working on plain data. On the right-hand side, however, the models obtained from the corresponding machine learning algorithm version working on encrypted data are drawn.

This visual panel can be extremely helpful to determine the suitability and accuracy of any machine learning algorithm implemented within the simulator, either working over plaintext data or encrypted data.

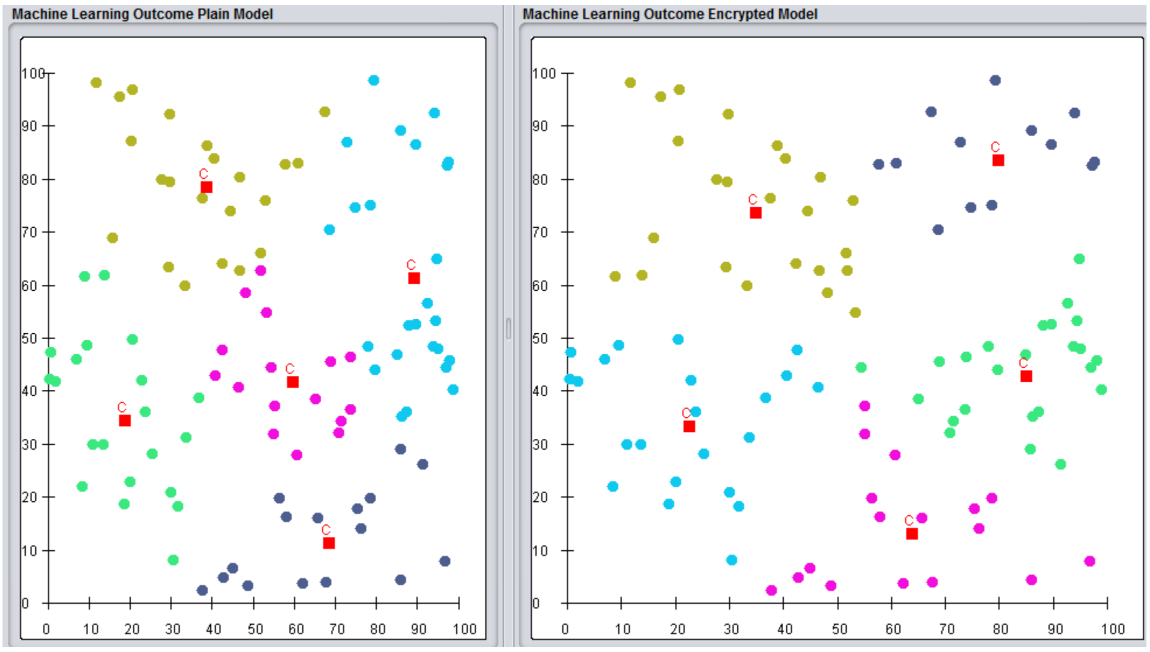

Machine Learning Outcome Plain Model
Machine Learning Outcome Encrypted Model
C

Figure 6: Simulator to test solutions to correlate encrypted data. Machine learning model panel.

Finally, Figure 7 shows again two panels where the final models for each simulated dataset are evaluated. In the current version of the simulator only the Davies–Bouldin index[89]is implemented to assess the accuracy of clustering algorithms. Such index is computed as follows

$$DB = \frac{1}{n}\sum_{i=1}^{n} \max_{j \neq i}\left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)}\right)$$

where n is the number of clusters, c_x is the centroid of cluster x, σ_x is the average distance of all elements in cluster x to centroid c_x, and $d(c_i, c_j)$ is the distance between centroids c_i and c_j.

As for the classification algorithms, the precision and recall [90] metrics are intended to be implemented in the next version of the simulator.

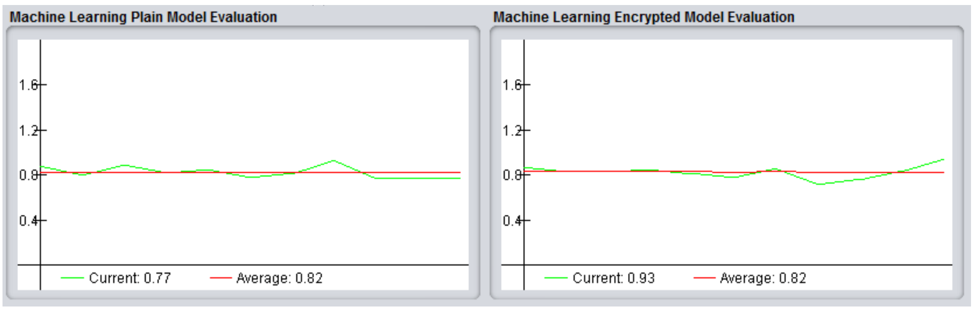

Machine Learning Plain Model Evaluation
Current: 0.77
Average: 0.82
Machine Learning Encrypted Model Evaluation
Current: 0.93
Average: 0.82

Figure 7: Simulator to test solutions to correlate encrypted data. Machine learning model evaluation panel.

To finalize this section, next we show the basic steps to be followed in order to develop and integrate a new machine learning algorithm in the simulator:

- Develop a new class implementing either `ClassificationAlgorithm` or `ClusteringAlgorithm` interfaces
- (If needed) develop a new class implementing either `ClassificationModel` or `ClusteringModel` interfaces
- (If needed) Develop a new class extending either `ClassificationAlgorithmParameters` or `ClusteringAlgorithmParameters` abstract classes
- (If needed) Develop a new class extending `MachineLearningAlgorithmParametersPanel` abstract class

- (If needed) Develop a new class extending `MachineLearningModelPanel` abstract class
- (If needed) Develop a new class implementing `DistanceMetric` interface
- (If needed) Develop new classes implementing `EncryptionAlgorithm` and `EncryptionKey` interfaces, respectively
- (If needed) Develop a new class extending `EncryptedFeature` class

2.2 Processing encrypted location information

One example for processing encrypted information that was developed in the project is PrivLoc. PrivLoc is a geofencing solution, i.e. the events that are detected are moving objects that cross the borders of a defined fenced area. In PrivLoc both, the location reports and the fenced areas are encrypted. Particularly, PrivLoc uses an encryption function that has the property that the encrypted information are again locations, and an intersect operation is preserved. If the encrypted points and lines intersect, also the plaintext objects intersect. This property allows that encrypted locations can be processed with standard geofencing systems using standard location databases.

We assume that the moving devices cannot be compromised by the adversary. Moreover, we assume that the providers operating the database servers are honest-but-curious. I.e. each database server will correctly follow the protocol (authenticate the nodes, output correct notifications) but is interested in acquiring information about the locations of the nodes in the system, and about the queries that are issued by the customers. Ideally, different database servers do not collude; this assumption especially holds when the database servers are hosted by different competing cloud providers (e.g., Amazon, Google). Moreover, we assume that the adversary cannot physically track the mobile users to acquire information about their movements, as this would trivially reveal their locations. Finally, we assume that the adversary is computationally bounded (i.e., she cannot acquire secrets, break secure encryption functions, etc.).

As mentioned earlier, the main premise behind our work is to design a privacy-preserving solution for a geofencing service hosted in the cloud, without incurring any modifications on the database servers D, and while ensuring that D does not learn any meaningful information about the location of the users and subscriptions in the system. Since the adversary can compromise a database server, we can express these security properties as follows:

- Confidentiality of Stored Records: Each database server should not learn any meaningful information about the stored subscriptions. This can be ensured if the transcript of interaction between the stakeholders is (computationally) independent of the actual subscription coordinates.
- Confidentiality of Queries: Similarly, each database server should not learn any meaningful information about the location of the nodes. This includes the direction of the movement, the trajectory taken by each node, the distance travelled by each node, etc.

Recall that both properties should be achieved without compromising the functionality of the database server, i.e., while enabling efficient geo-spatial indexing, search over encrypted data, etc.

This section gives a short overview of the PrivLoc system. Further detailed specifications can be found in the publication [80].

2.2.1 PrivLoc protocol

PrivLoc unfolds as follows. The protocol introduces a trusted proxy T that takes the role of encrypting the location information of position reports and subscription request. This avoids that the key for encryption and decryption needs to be distributed to objects and users. To prepare the system, T first proceeds to dividing the original map by a regulargrid into fixed-size square tiles. For privacy reasons, the area covered by a tile shouldfall above a given threshold.

We assume the availability of a pseudo-random function PRF, which can be realized by a HMAC construction, and a pseudorandom permutation PRP on the grid of tiles. Exemplary PRP constructions can be found in [81].

In the *setup* phase, T generates key k as a uniformly random bit string of a length depending on the intended security level, e.g. 128 bit. As mentioned earlier, PrivLoc requires that T encrypts the coordinates reported by devices before transmitting them to D. Besides hiding location information, one important goal here is to hide the fine-grained user movements (i.e., direction and distance of movement). Recall that, in our setting, the subscribers are interested in knowing whether a device has crossed a Geofence. For the database servers to determine that, it is therefore necessary to be able to compare the relative coordinates between the origin and the destination of every sensor movement segment along with the boundaries of Geofences (or subscriptions).

PrivLoc achieves the aforementioned goals by dividing the map into tiles and distorting the original coordinate system into three different variant maps (for the reasoning why, see following paragraphs). By doing so, PrivLoc encrypts the location of each tile within the maps, but ensures that every received location report can be fitted to at least one full tile in one of the maps. Subscriptions that cross more than one tile are also subsequently split to several smaller Geofences that are completely contained in a single tile. Within that tile, PrivLoc further distorts the direction and distance of the user movement while enabling the spatial database to find all intersections between the movement vector and translated Geofences in the distorted tile using existing indexing techniques.

More specifically, PrivLoc applies the encrypt procedure

- encrypt takes as input real world coordinates (x, y) and transforms them into obfuscated coordinates (newx, newy).
- encrypt is executed on movement vectors of objects represented by the movement end-points, and on subscriptions to areas represented by the south-west, and the north-east coordinates that define the minimum bounding box of each Geofence.

The algorithm encrypt consists of four main routines:

- permuteTiles is used to divide the map into equal-sized tiles and permute these tiles. By doing so, permuteTiles hides the location of the devices within the map.
- rotateTile and flipTile are used to rotate and flip each tile in the distorted map. Both routines serve to hide the direction of devices' movements within the original map.
- Order preserving encryption (OPE) is used to hide the distance between any two locations within each tile. OPE keeps the notion of intersections, i.e. the order of points on a line is preserved.

Here, permuteTiles, rotateTile, and flipTile hide the location and direction of the movement of each device from D, while OPE distorts the distances within each tile. All four routines, however, enable D to rely on existing indexing techniques to compare different locations within each tile.

When encrypting a vector two cases need to be distinguished based on the tiling of map:

Hiding movements within a tile:

> Recall that PrivLoc operates on movement vectors reported by the users. Upon receiving the vector from devices, T encrypts the vector, by applying the encrypt procedure in Algorithm 1, to both the start and end-point of the movement vector. These points are represented by 2D coordinates (x, y). The encryption function invokes the procedure coordinatesOnTile, which returns the tile number Num of the tile where the point (x, y) lies on, and the relative coordinates (dx, dy) on the respective tile. As the map is split in n × m tiles, we have Num∈Zn×m. This coordinate translation is achieved by means of Algorithm 2.
>
> As the first step of hiding the location of these points within the map, T permutes the tiles using the key k by applying the pseudorandom function: PRP(k; n × m; Num). Note that k is only held by the trusted server T. The total size of the grid of tiles is n × m. This function computes a pseudorandom permutation of the numbers in $\mathbb{Z}_{n\times \mathrm{m}}$ and outputs the new position Num for the tile T.
>
> To further hide the linkability between two consecutive location reports (i.e., to prevent leakage of movement information), T first rotates each tile using rotateTile (rotation chosen at random among 0, π/2, π, 3π/4), and then (individually) flips them using flipTile, in order to obfuscate the direction of the movement. Both operations result in a transformation entropy of three bits per tile. When combined with permuteTile, this results in a total of 3⌊log2(nm)⌋ bits of entropy per tile.

rotateTile takes the output of permuteTile; the position of the tile T is not changed in this algorithm. The rotation angle of T is determined using the key-based pseudo-random function PRF. T then applies tile flipping. Given the outputs of rotateTile, flipTile outputs the newly flipped coordinates (newx, newy). Note that flipTile is analogous to rotateTile, except that the coordinates are transformed by a mirror matrix.

In order to hide the distances of movements executed by devices, PrivLoc further relies on the use of order-preserving encryption (OPE) within each tile. Order-preservingencryption has the property that the relative order among coordinates is preserved after encryption. That is, let OPE(p) denote the order-preserving encryption of plaintext p.

Then, the following holds:

$$\begin{cases} \mathrm{OPE}(p_1) \leq \mathrm{OPE}(p_2) \ if p_1 \leq p_2, \\ \mathrm{OPE}(p_1) > \mathrm{OPE}(p_2) \ if p_1 > p_2. \end{cases}$$

Hiding Cross-Tile Movements

Clearly, encrypt can only hide the trajectory, and distance exhibited by location reports from D when they fall within the same tile. However, if two consecutive location reports cross the boundary of a single tile and are matched to different tiles, then

(i) D cannot compare the relative location advertised by these reports and

(ii) D might be able to guess that these tiles correspond to physically connected tiles in the original map.

Thus, a query for a movement which crosses the tile boundary must be avoided in PrivLoc. To achieve that, PrivLoc relies on more than one tiling of the same map (see Figure 8); although a movement might cross the boundary of one map tiling, PrivLoc ensures that there is at least one tiling, in which the movement falls completely within a single tile. Here, by different map tilings, we refer to slightly shifted variants of the original map, which have been processed (i.e., using the encrypt routine) by means of different keys. As shown in Figure 8, the maximum magnitude of a movement vector that PrivLoc accepts is bounded by l = t/3 + ǫ, where t is the length of one tile. Thus, all vectors with magnitude d(x, y) < r = t/3 can be located within one complete tile stored at least on one of the three servers. Within that tile, this enables the comparison between all closely located points x and y, where d(x, y) < t/3, while ensuring that the movement from location x to y does not cross tile1. An example of a map with three tilings is depicted in Figure 8.

As shown in Figure 8, we point out that relying on two tilings of the same map is not enough. This is the case since there exist movement vectors which can still cross two different tiles in any two map tilings. However, it is easy to show that given three shifted tilings of the same map, there is no point where all boundaries intersect, thus ensuring that every movement can be fitted to one tile pertaining to at least one tiling. This is exactly why PrivLoc requires the presence of three database servers D1, D2, and D3, which store subscriptions pertaining to the three map tilings. All received subscriptions are stored encrypted with three different keys k1, k2 and k3 on all three servers respectively. Whenever T receives a location report, it will choose to query the database server D_i, i ∈ {1, 2, 3} for whom this report falls in a full tile. Here, T can efficiently find out which server to query for each movement.

As a by-product, we point out that the reliance on multiple database servers in our scheme inherently achieves load balancing of the load on the servers and increases the load capacity of the entire system. This is the case since T only queries one server for each received location report.

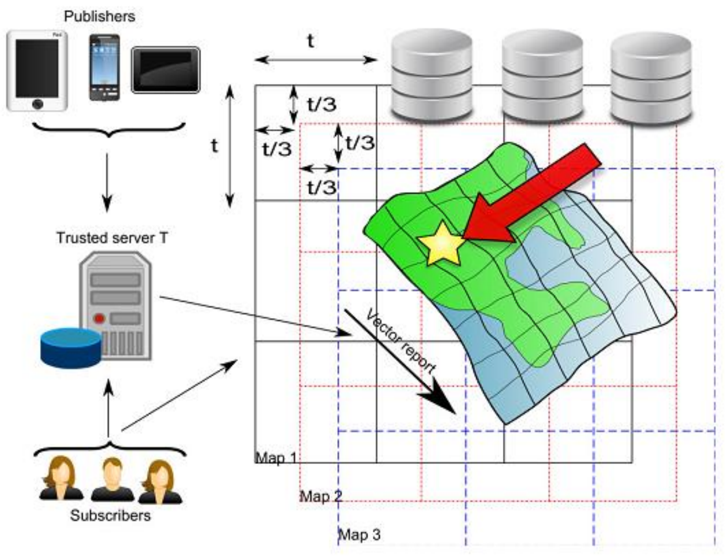


Figure 8: Example of three tilings of the original map. Notice here that the movement vector crossesthe boundaries of tiles in “Map 1” (shown in solid lines) and “Map 2” (shown in dotted lines) but can be fitted complexity within a tile in “Map 3” (shown in dashed lines).

Encrypting Subscriptions

We now proceed to describing how T encrypts subscriptions and stores them at the database servers D. Clearly, for D to be able to use existing functionality to compute the intersections of the movements with subscriptions, the subscriptions must be encrypted using the same routine that is used to process the location reports. More specifically, T uses the encrypt routine to encrypt the north-east and the south-west coordinates which define the minimum bounding box for each subscription. This is done using the three keys k1, k2, and k3, respectively. The resulting encrypted coordinates are stored in database servers D1, D2, and D3 (see Figure 9). Here, T must ensure that in case a subscription crosses the boundaries of tiles for the map tiling at a database server, the subscription needs to be split into parts, that each completely fit within one tile of the corresponding map tiling. This process has to be repeated for each of the three servers independently. While this incurs additional storage overhead per server to store the tiles, as storage is becoming rather cheap, we believe it is tolerable.

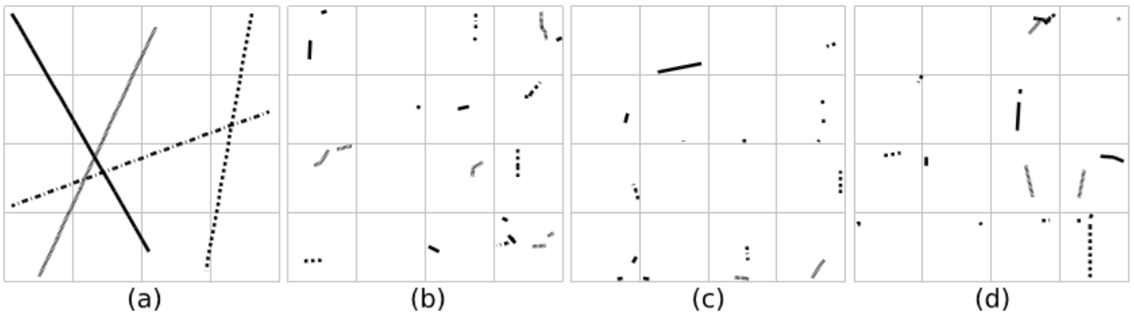


Figure 9: Exemplary run of PrivLoc with four different movements. Each path displayed in (a) consists of 20 movement reports, each of a length of 30% of the tile width. We depict the snapshots seen by database servers D1, D2, and D3 in (b), (c) and (d), respectively. Note that the marking of the vectors in (b)-(d) is only for visualisation, since different location reports cannot be linked by the database.

2.2.2 Security considerations

Figure 9 gives an impression what the data seen by the database servers D looks like. When looking at the possibilities of the adversary to identify a real-world trace of a single node, two cases emerge:

1) Analyzing single location reports: When receiving an encrypted location vector from T, we note that A cannot infer the location of the corresponding user who generated it in the original map. This is the case since the keyed PRP used in the permuteTiles function ensures that the adversary cannot guess the actual location of the tile which hosts the report in the original map. Moreover, OPE ensures that A cannot acquire the actual distance travelled by the user while the keyed tile rotation and flipping ensure that A cannot guess the direction of the user movement given the received location vector.

2) Correlating two or more location reports: Recall that D_i will only receive location reports which correspond to a complete tile given the tiling hosted by D_i. As mentioned earlier, the combined use of the permuted tile location of the report, the OPE-obfuscated distance travelled by the user, and tilted/rotated movement direction does not offer any distinguisher for A to correlate two or more location reports.

Similarly, it is easy to show that A cannot acquire any meaningful information about the subscriptions in the system. Note that when tiling and permuting tiles at each server, each subscription might be split into a number of smaller subscriptions within each tile. However, assuming that the system hosts a number of subscriptions, this does not give any advantage to A in inferring information about the subscriptions. Note, here, that A can acquire considerable information as T populates D_i. For instance, it is straightforward for A to guess with high probability that consecutive subscriptions in D_i correspond to the same actual subscription which was subsequently split by T to prevent subscriptions from crossing tile borders. This is exactly why PrivLoc requires that T batches the processing of several subscriptions at a time. On one hand, this enhances the anonymity of the subscriptions, and on the other hand, this prevents A from correlating stored encrypted subscriptions (e.g., map them to the same subscription) and acquiring information about the tile permutation.

As mentioned in Section 2.3, PrivLoc can only ensure confidentiality of the input/output, and does not prevent the correlation among the inputs and outputs of D_i. That is, A can learn whether a given location report has triggered a response from $D_i$and therefore corresponds to a user entering/exiting a Geofence. The literature features a number of solutions for this problem (see [82][83][84][85]).

3 Framework for Access Control

Over recent years, significant technological challenges have been solved through the extension and adaptation of wireless communication technologies and protocols. In particular, several IETF working groups, such as IPv6 over Low power WPAN (6LoWPAN) and Constrained RESTful Environments(CoRE), are focused on the adaptation of existing Internet protocols to more efficient, interoperable and lightweight versions to be used on constrained environments. These protocols are intended to enable a seamless inclusion of smart objects into the Internet to realize the scenarios which are envisaged by IoT community. Specifically, the main goal of the 6LoWPAN WG is the adaptation of the IPv6 protocol to be employed on constrained environments such as IEEE 802.15.4 networks, in order to obtain end-to-end connectivity between constrained devices and any entity connected to the Internet [4],[5]. These adjustments are based on header compression and encapsulation mechanisms. Moreover, the CoRE WG was specifically founded to define an application layer protocol for resource constrained devices. As a result, the Constrained Application Protocol (CoAP) [6] was designed. This protocol, based on the same RESTful principles as HTTP, allows the realization of embedded services but accommodated to the requirements of constrained devices and networks [7].

In spite of such remarkable efforts, the application of security mechanisms to be deployed on this new generation of pervasive scenarios still remains as the main concern for a global IoT deployment[8]–[10]. In fact, the realization of these scenarios requires addressing significant security and access control implications, since physical and constrained devices are being seamlessly integrated into the Internet infrastructure with network and processing abilities, making them vulnerable to attacks and abuse [11]. However, current security and access control solutions were not designed with these aspects in mind and they are not able to meet the needs of these incipient ecosystems regarding scalability, interoperability, lightness and end-to-end security[12],[13]. In this direction, the IETF Authentication and Authorization for Constrained Environments (ACE) WG has been recently established to produce a standardized security solution to be used by devices and networks with tight resource constraints. Specifically, the work of ACE WG is focused on the design and development of authentication and authorization mechanisms to enable authorized access to resources, which are hosted in constrained smart objects.

Under the main foundations of ACE WG, we propose a set of lightweight authentication and authorization mechanisms, as well as their application on IoT constrained environments. These mechanisms are integrated and extended with other standard security technologies in order to support smart objects during its life cycle. In particular, we consider the integrationof standard technologies, such as EAP [15] and Remote Authentication Dial In User Service (RADIUS) [16]. This process has been extended with Extensible Access Control Mark-up Language (XACML)-based authorization procedures [17] for obtaining lightweight access tokens[18]to be employed at operational plane achieving end-to-end secure communication between constrained devices. Furthermore, such mechanisms are framed within a security framework which is compliant with the Architectural Reference Model (ARM) [19], recently presented by the EU FP7 IoT-A initiative. While this work is focused on authentication and authorization mechanisms, the proposed framework is intended to provide a holistic security approach to be leveraged by IoT devices throughout their life cycle.

3.1 State of the Art

The application of security mechanisms on IoT scenarios has to address new requirements due to the nature and tight constraints of devices and networks composing these incipient ecosystems. This has given rise to a broad consensus among academia and industry to consider security as the main barrier to be overcome in next years for a global deployment of IoT[20]–[23]. These challenges have attracted huge attention from the research community, and recently several efforts are beginning to emerge addressing different security aspects during the life cycle of smart objects.

Regarding the application of security mechanisms at the bootstrapping stage for constrained devices, the authors in [24] provide an authentication and key establishment scheme for WSNs in distributed IoT applications. The proposal, which is also envisioned for bootstrapping phase, is based on a simplified Datagram Transport Layer Security (DTLS) [25] exchange and the use of TinyECC[26] for cryptographic operations [27] provides the main bootstrapping approaches and protocols to be considered on IoT environments. Specifically, EAP [15]is established as the standard authentication framework for this process due to its maturity and flexibility. Additionally, three alternative protocols are analysed for security bootstrapping: HIP Diet EXchange (HIP-DEX) [28], Protocol for Carrying Authentication for Network Access (PANA) [29] and 802.1X [30]. The use of HIP-DEX for the network access stage is analysed in [31]. Although the results shown are promising compared to DTLS, HIP-DEX is not widely adopted. This is mainly because it does not provide native support for certificate-based public key agreement and the high complexity of the puzzle mechanism to mitigate DoS attacks. Moreover, the authors in [32] provide a lightweight implementation of PANA called PANATIKI to be deployed on constrained devices. Due to the high cost of public key cryptographic operations, it is based on the use of Extensible Authentication Protocol-Pre-Shared Key (EAP-PSK) [33] as the authentication method, providing a lower degree of scalability and security. In addition, previous proposals assume that the device has already been configured with an IP address prior the network access stage, which can stand for a potential security threat for the network. Alternatively, our approach operates below the network layer, by using 802.1X to transport EAP messages at the bootstrapping stage, offering a lightweight mechanism suitable to the requirements of resource-constrained environments.

At operational plane, CoAP[6]has been recently declared as a standard specialised web transfer protocol to be deployed on constrained devices and networks. CoAP offers several modes for securing the protocol through a security binding to DTLS [25], which requires a heavy message exchange to agree security parameters. Furthermore, it does not cover the use of authorization and access control mechanisms at the application level. In this direction, the work presented in [34] provides an approach based on Elliptic Curve Cryptography (ECC) for key establishment and Role-Based Access Control (RBAC) model [35] for the definition of access control policies. They consider an inter-domain scenario in which different registration authorities are responsible for the authentication process. Several security gaps of this work are discussed in [36], in which different enhancements are proposed in order to satisfy the basic security properties of the scenario. Moreover, an authentication and access control scheme for the layer perception of IoT is given in [37]. It is based on an efficient key establishment making use of ECC and Attribute-Based Access Control (ABAC) [38], which requires a complex management and hinders its application to constrained devices. Consequently, they only provide theoretical results of the proposed model. Recently, several access control mechanisms based on authorization tokens have been proposed on IoT scenarios [39], [40]. These approaches are based on the externalization of authorization decisions in a central entity which issues privileges to be enforced at the end device [41]. Under the main foundations of these works, Distributed Capability-Based Access Control (DCapBAC) [18] has been recently introduced as a feasible access control approach to be deployed on constrained environments. DCapBAC allows a distributed approach in which constrained devices are enabled with authorization logic by adapting the communication technologies and data-interchange format. Specifically, it makes use of JavaScript Object Notation (JSON) [42] as representation format for the token, the use of emerging communication protocols such as CoAP and 6LoWPAN, as well as a set of cryptographic optimizations for ECC [43].

3.2 Security Bootstrapping

The bootstrapping process usually consists of a set of procedures in which a node is installed and commissioned within a network. Optionally, this stage can include authentication and access control mechanisms to get security parameters for trusted operation. For a successful and secure bootstrapping process, well-known mechanisms need to be on the basis. Additionally, in the context of IoT constrained scenarios, the application of such procedures need to be analysed due to implicit requirements of these environments.

Currently, EAP [15] is widely used and recognized as the standard mechanism to provide flexible authentication through different EAP methods. According to EAP terminology, these methods allow an EAP peer to be authenticated by an EAP server through EAP authenticator for network access. Moreover, these EAP methods can provide keying material after successful authentication. Depending on the place in where the EAP server is located, there are two possible configurations: standalone and pass-through. In the former option, the EAP Server is collocated with the EAP authenticator and the communication between the two components is local. In the pass-through configuration, the EAP Server is located on a different node and an additional AAA protocol is required for this communication (e.g. RADIUS). For communication between the EAP peer and the EAP authenticator, an additional protocol is needed to transport EAP messages. For this purpose, there are several options which operate at different communication layers, such as PANA [29], 802.1X [30]and Internet Key Exchange (IKE) [52]. The Internet Protocol security (IPsec) and IKE have been evaluated by [53] and [54], whose results show that both options could require excessive cost for constrained environments. Moreover, PANA operates on top of IP layer and nodes need to be addressable at the bootstrapping process before being authenticated. Moreover, the use of PANA requires more overhead and processing requirements than a solution operating at a lower level.

Further work in this area is being done to provide a lightweight mechanism for the bootstrapping phase and presented in the IETF draft [73].

3.3 Operational Security

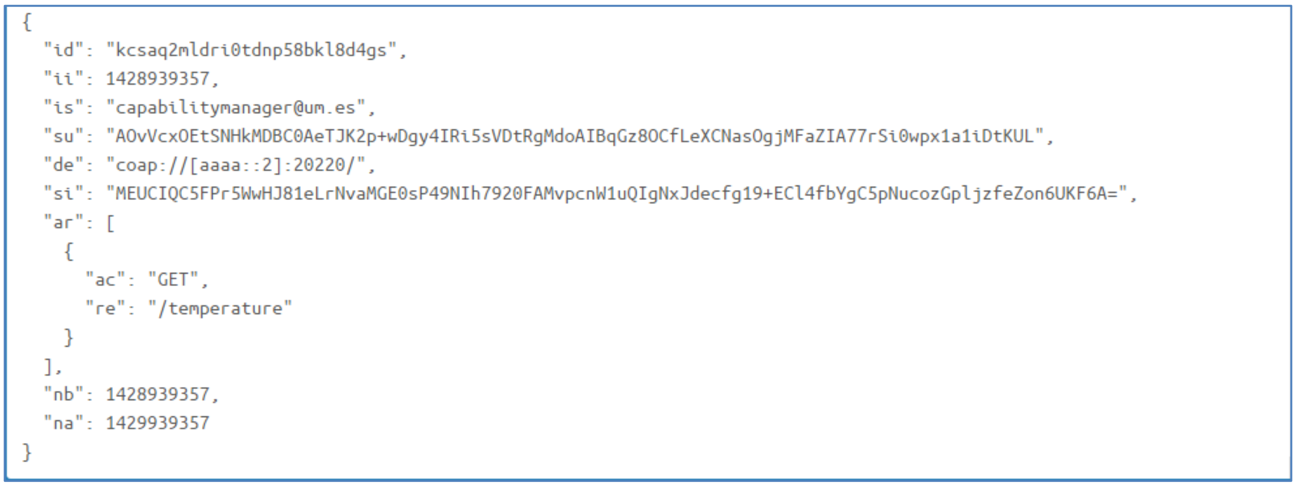
Security at operational level guarantees that only trusted and legitimate instances of an application running in the IoT can communicate with each other, through the use of the corresponding security mechanisms at the application layer. Specifically, CoAP[6] defines a security binding to DTLS [25] through the use of pre-shared keys, raw public keys or certificates. However, it does not cover the use of authorization and access control mechanisms at the application level. Because of the strong constraints of IoT devices and networks, in recent years, the protection of resources and services which are provided by smart objects, has been mainly addressed by centralized architectures, in which a back-end server or gateway is responsible for security tasks. While traditional access control models and security standard technologies and protocols can be used in these approaches, several drawbacks arise when they are considered on a real deployment. On the one hand, the inclusion of a central entity prevents end-to-end security to be achieved. On the other hand, these solutions cannot provide a suitable level of scalability for smart environments with a potentially huge amount of constrained devices. Furthermore, due to the fact that a single entity stores and manages all the data from a set of devices, any vulnerability might compromise a vast amount of sensitive information.

While most of mentioned drawbacks of centralized approaches can be solved, a distributed security approach requires the analysis and adaptation of the standard mechanisms to be deployed on devices with tight resource constraints. Recently, the Distributed Capability-Based Access Control model (DCapBAC) has been postulated as a realistic approach to be used on IoT scenarios [18]. This model offers several advantages over more established approaches and it is gaining attention from research community providing support for least privilege, and providing a greater level of suitability to cope with the inherent requirements of a distributed approach. The key element of DCapBAC is the capability. The concept of capability was originally introduced in [55] as “token, ticket, or key that gives the possessor permission to access an entity or object in a computer system”. This concept is applied to IoT environments and extended by defining conditions which are locally verified on the constrained device. This feature enhances the flexibility of DCapBAC since any parameter which is read by the smart object could be used in the authorization process. Moreover, it is based on technologies which have been proposed recently to be used in constrained environments. In particular, authorization tokens are specified with JSON [42], which are sent to the target device by using CoAP. Moreover,DCapBACrequires the use of a security protocol for the exchange of the token. For this purpose a DTLS exchange is chosen between the two entities.

The security approach at operational plane is based on DCapBAC. However, while the proposed scenario [18] is mainly intended for a Human-to-Thing (H2T) communication, here we intend to achieve a transparent Thing-to-Thing (T2T) communication in order to provide the benefits of a decentralized approach for IoT scenarios in terms of end-to-end security, scalability, interoperability, and flexibility. In addition, weaddress the token generation stage in order to provide an integral approach for the security management during the life cycle of resource-constrained devices.

3.4 DCapBAC in JSON format

In order to implement our distributed capability-based access control approach, we chose JSON[42]as format to represent the capability token. Compared to more traditional formats such as XML,JSON is getting more attention from academia and industry in IoT scenarios, since it is able toprovide a simple, lightweight, efficient, and expressive data representation which is suitable to beused on constrained networks and devices. Figure 10shows a capability token example, which hasbeen used to demonstrate the feasibility of our proposal. Below, a brief description of each field is provided.



```
{
  "id": "kcsaq2mldri0tdnp58bkl8d4gs",
  "ii": 1428939357,
  "is": "capabilitymanager@um.es",
  "su": "AOvVcxOEtSNHkMDBC0AeTJK2p+wDgy4IRi5sVDtRgMdoAIBqGz8OCfLeXCNasOgjMFaZIA77rSi0wpx1a1iDtKUL",
  "de": "coap://[aaaa::2]:20220/",
  "si": "MEUCIQC5FPr5WwHJ81eLrNvaMGE0sP49NIh7920FAMvpcnW1uQIgNxJdecfg19+ECl4fbYgC5pNucozGpljzfeZon6UKF6A=",
  "ar": [
    {
      "ac": "GET",
      "re": "/temperature"
    }
  ],
  "nb": 1428939357,
  "na": 1429939357
}
```

Figure 10: Capability Token Example.

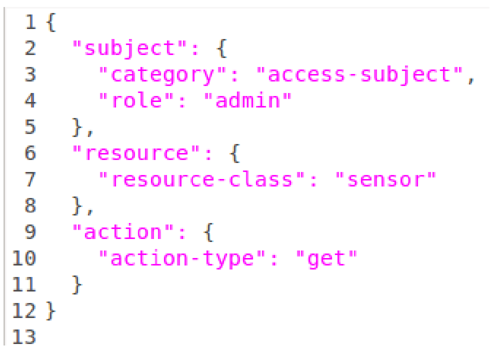
- Identifier (ID) (*16 bytes*). This field is used to unequivocally identify a capability token. A random or pseudorandom technique will be employed by the issuer to ensure this identifier is unique.
- Issued-time (II) (*10 bytes*). Following the notation of [75], it identifies the time at which the token was issued as the number of seconds from 1970-01-01T0:0:0Z.
- Issuer (IS) (*variable size*). The entity that issued the token and, therefore, the signer of it.
- Subject (SU) (variable size). It makes reference to the subject to which the rights from the token are granted. A public key has been used to validate the legitimacy of the subject. Specifically, it is based on ECC-based PKC, therefore, each half of the field represents a public key coordinate of the subject using Base64.
- Device (DE) (*variable size*). It is a URI used to unequivocally identify the device to which the token applies.
- Signature (SI) (variable size). It carries the digital signature of the token represented in Base64. The length will depend on the signature algorithm used to sign the token.
- Access Rights (AR). This field represents the set of rights that the issuer has granted to the subject.
  (1) Action (AC) (*variable size*). Its purpose is to identify a specific granted action. Its value could be any CoAP method (GET, POST, PUT, DELETE).
  (2) Resource (RE) (*variable size*). It represents the resource in the device for which the action is granted.
  (3) Condition flag (F) (*1 byte*). Following the notation in [76], it states howthe set of conditions in the next field should be combined.A value of 0 meansAND and a value of 1 means OR.
  (4) Conditions (CO). Set of conditions which have to be fulfilled locally on the device to grantthe corresponding action.
    a) Condition Type (T) (*1 byte*). The type of condition to be verified as stated by Li *et al*.
    b) Condition value (V) (*variable size*). It represents the value of the condition.
    c) Condition Unit (U) (*variable size*). It indicates the unit of measure that the valuerepresents. Its value could be any of those stated by Jennings *et al. [77]*.
- Not Before (NB) (*10 bytes*). The time before which the token must not be accepted. Its valuecannot be earlier than the II field and it implies the current time must be after or equal than NB.
- Not After (NA) (*10 bytes*). It represents the time after which the token must not be accepted.

3.5 Attribute Based Access Control (XACML) and DCapBAC

Up to now we have been introducing the steps that will lead to a complete framework for achieving secure end-to-end communication between the user (a person or a thing) with other things.But for simplicity we will refer to it as thing-to-thing communication. How the user in this case obtains the capability token is another step in this process.

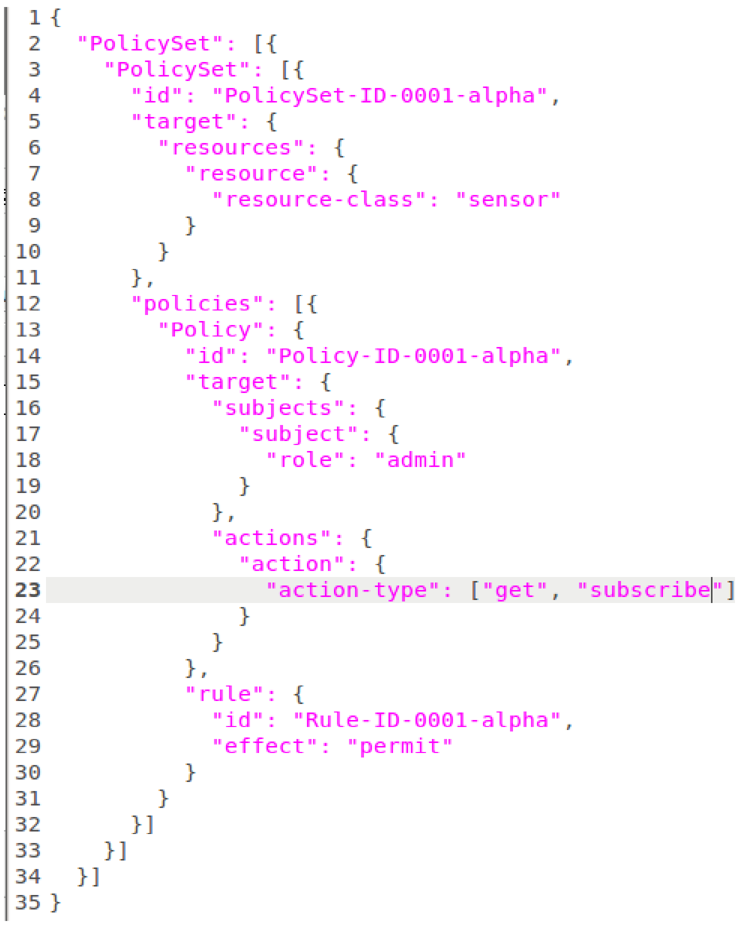
The authorization process by which the user obtains the capability token will be addressed with XACML which is a compliant ABAC framework enabling fine grained access control. Moreover, XACML provides enriched policies that can require information that is called “environmental information” to add flexibility to the access control decision.

XACML currently uses XML as a Mark-up language to define the policies and requests and while it is a versatile option, when dealing with constrained devices and high demanding environments (all things might require capability tokens to perform certain queries) we need to use a language that will not impose a toll to the overall process. Here is where JSON comes to the XACML scenario.



```
{
  "subject": {
    "category": "access-subject",
    "role": "admin"
  },
  "resource": {
    "resource-class": "sensor"
  },
  "action": {
    "action-type": "get"
  }
}
```

Figure 11: Example of XACML Request in JSON.



```
{
  "PolicySet": [{
    "PolicySet": [{
      "id": "PolicySet-ID-0001-alpha",
      "target": {
        "resources": {
          "resource": {
            "resource-class": "sensor"
          }
        }
      },
      "policies": [{
        "Policy": {
          "id": "Policy-ID-0001-alpha",
          "target": {
            "subjects": {
              "subject": {
                "role": "admin"
              }
            },
            "actions": {
              "action": {
                "action-type": ["get", "subscribe"]
              }
            }
          },
          "rule": {
            "id": "Rule-ID-0001-alpha",
            "effect": "permit"
          }
        }
      }]
    }]
  }]
}
```

Figure 12: Example of XACML Policy in JSON.

As seen in the previous figures, using this approach we reduce the length and complexity of the requests and even the policies. So, this approach reduces the process time needed to resolve the request against the policies in order to provide a quick a response for emitting a capability token. This proposal [74] confirms that JSON is a viable approach to enhance the performance of the access control decision.

4 Framework for Privacy

As protocols for enabling the communications with constrained devices are being proposed, developed, and standardized, as well as protocols for access control in these environments, we do not have to forget the importance of privacy and identity management.

Privacy and identity management solutions have to cope with new challenges due to inherent nature and requirements of IoT which is being extended as part of our personal sphere. In this direction, several remarkable efforts, such as Daidalos[1], SWIFT[2], or Primelife[3] European projects, have been undertaken in recent years regarding the application of novel user-centric privacy-preserving identity management schemas. However, currently there is a lack related to definition of an integral architecture able to tackle these issues for IoT scenarios. Therefore, the proposed solutions need to be adapted to the envisioned scenarios allowing more flexible sharing models (beyond the classic request/response approach), as well as fleeting and dynamic associations between entities, while privacy is still preserved.

In order to address the main challenges of privacy-preserving data sharing solutions, here the design of a framework to be applied on IoT scenariosis presented. In the IoT paradigm, a smart object may be represented by any entity which is able to generate (producer) and get (consumer) information. We assume that each smart object possess a partial identity [57] in the context in which the sharing transaction is being performed. Based on this, we intend to create a secure and privacy-preserving framework for data sharing among smart objects. The proposed design allows the application of recent proposals in this field. Specifically, we analyse the use of privacy-enhancing techniques such as anonymous credential systems [58], as well as CP-ABE [59]for secure data dissemination. The development of this framework represents a step forward to achieving an integral, flexible and privacy-preserving sharing model to be used in the next generation of the digital era.

4.1 Secure Data Dissemination: A cryptographic Approach

The requirements presented by common IoT scenarios require more flexible data sharing models between entities while the privacy of smart objects involved is still preserved. Unlike the current Internet, IoT interaction patterns are often based on short and volatile associations between entities without a previously established trust link. In this section, we review existing cryptographic schemes that can be potentially used to implement a secure information sharing based on the push model. These mechanisms should be applied on the smart objects themselves in order to provide an end-to-end secure data dissemination.

Because of the usage of resource-constrained devices, Symmetric Key Cryptography (SKC) has been considered as cryptographic solution on emerging IoT scenarios [31]. However, due to its cumbersome key management and distribution, it is unable to provide a suitable level of scalability and interoperability in a future with billions of heterogeneous smart objects. These issues are tackled by Public Key Cryptography (PKC), but presenting significant requirements on computing and memory resources. This has fostered the development of recent research initiatives in order to make the application of PKC on IoT scenarios feasible through several optimizations on Elliptic Curve Cryptography [56]. A common feature with SKC is that PKC allows a producer to encrypt information to be accessed only by a specific consumer. However, given the pervasive, dynamic and distributed nature of IoT[11], it is necessary to consider more flexible data-sharing models in which some information can be shared with a group of consumers or a set of unknown receivers and, therefore, not addressable a priori. In addition, the use of certificates in scenarios with a potentially large number of interacting entities can be cumbersome because of the need for validation and revocation mechanisms.

This issue has led to the emergence of Identity-Based Encryption (IBE) [66] as an alternative without certificates for PKC, in which the identity of a consumer is not determined by a public key, but a string. This represents a highly relevant feature to be leveraged by more advanced sharing schemes in IoT environments, since a data producer could share data with a set of consumers whose identity is described by a specific string. In this direction, Attribute-Based Encryption (ABE) [62] represents the generalization of IBE, in which the identity of the participants is not represented by a single string, but by a set of attributes related to their identity. Just as IBE, it does not use certificates, while cryptographic credentials are managed by an entity usually called Attribute Authority (AA).

ABE is gaining attention because of its high level of flexibility and expressiveness, compared to previous schemes. In ABE, a piece of information can be made accessible to a set of entities whose real, probably unknown identity, is based on a certain set of attributes. This represents a step forward in order to realize a privacy-preserving and secure data sharing scheme in pervasive and ubiquitous scenarios, since consumers do not need to reveal their true identity to obtain information, while producers can be sure that their data are accessed only by authorized entities.

Based on ABE, two alternative approaches were proposed. In KP-ABE [67], a cipher text is encrypted under a set or list of attributes, while private keys of participants are associated with combinations or policies of attributes. In this case, a data producer has limited control over which entities can decrypt the content, being forced to rely on the AA entity issues appropriate keys for getting access to disseminated information. In contrast, in a CP-ABE scheme [59], a cipher-text is encrypted under a policy of attributes, while keys of participants are associated with sets of attributes. Thus, CP-ABE could be seen as a more intuitive way to apply the concepts of ABE; on the one hand, a producer can exert greater control over how the information is disseminated to other entities, On the other hand, a user's identity is intuitively reflected by a certain private key. In addition, CP-ABE is secure to collusion attacks, that is, different keys from their corresponding entities cannot be combined to create a more powerful decryption key. This feature is due to the use of individual random factors for each key generation. Moreover, in order to enable the application of CP-ABE on constrained environments, the scheme could be used in combination with SKC. Thus, a message would be protected with a symmetric key, which would be encrypted with CP-ABE under a specific policy.

4.2 CP-ABE algorithms

Following our terminology and the proposed framework, a basic CP-ABE scenario consists of three main entities: a data producer, one or more data consumers, and an attribute authority (AA), responsible for generating the appropriate keys for all participants. This authority could be also instantiated by data producers if full control over their resources is required. The role of these components is determined by their participation in the algorithms of a CP-ABE scheme [59]:

4.2.1 The CP-ABE Algorithms

Let G_0 be a bilinear group of prime order p, and let g be a generator of G_0. In addition, let $e : G_0 \times G_0 \rightarrow G1$ denote the bilinear map. A security parameter, κ, will determine the size of the groups. Also define the Lagrange coefficient $\Delta_{i,S}$ for $i \in Z_p$ and a set, S, of elements in Z_p: $\Delta_{i,s}(x) = \prod_{j\in S,\ j\neq i}\frac{x-j}{i-j}$. We will additionally employ a hash function $H: \{0, 1\}^* \rightarrow G_0$ that we will model as a random oracle. The function will map any attribute described as a binary string to a random group element. The construction follows.

• Setup (AA) ($\lambda \rightarrow \{PP, MSK\}$). It takes an implicit security parameter λ as input. The algorithm generates the public parameters PP which are common to all users of the system (for example, the universe of attributes U) as well as a master secret key MSK which are used by AA to generate secret keys for participants. If AA functionality is performed by producers, the Identity Manager Module is in charge of this algorithm.

The setup algorithm will choose a bilinear group G_0 of prime order p with generator g. Next it will choose two random exponents $\alpha,\beta \in Z_p$. The public key is published as:

$$PK = G_0, g, h = g^{\beta}, f = g^{1/\beta}, e(g,g)^{\alpha}$$

and the master key MK is (β, g^{α}). (Note that f is used only for delegation.)

• Key Generation (AA) ({MSK, A}→ SKA). After a consumer proves it has a certain set of attributes A, the algorithm takes as input the master key MSK and the set A. The result is a private key SKA for DC. Similarly, in the case AA functionality is performed by producer, Identity Manager Module is responsible to generate these keys.

The key generation algorithm will take as input a set of attributes S and output a key that identifies with that set. The algorithm first chooses a random r $\in Z_p$, and then random $r_j \in Z_p$ for each attribute j ∈ S. Then it computes the key as

$$SK = (D = g^{\frac{\alpha+r}{\beta}},$$
$$\forall_j \in S : g^r \cdot H(j)^{rj}, D'_j = g^{rj})$$

• Encrypt (Producer) ({PP, M, PT}→ CT). It takes the message M, public parameters PP and a decryption policy P T representing subsets of attributes which are allowed to decrypt M. The result of this algorithm is a cipher text CT containing PT. The functionality to select a proper sharing policy under a specific context will be carried out by the Group Manager of the proposed framework.

The encryption algorithm encrypts a message M under the tree access structure T. The algorithm first chooses a polynomial q_x for each node x (including the leaves) in the tree T. These polynomials are chosen in the following way in a top-down manner, starting from the root node R. For each node x in the tree, set the degree dx of the polynomial qx to be one less than the threshold value k_x of that node, that is, dx = $k_x - 1$. Starting with the root node R the algorithm chooses a random s $\in Z_p$ and sets $q_R(0) = s$. Then, it chooses d_R other points of the polynomial q_R randomly to define it completely. For any other node x, it sets $q_x(0) = q_{parent(x)}(index(x))$ and chooses d_x other points randomly to completely define q_x. Let, Y be the set of leaf nodes in T. The cipher text is then constructed by giving the tree access structure T and computing

$$CT = (T, \tilde{C} = Me(g,g)^{\alpha s}, C = h^s,$$
$$\forall_y \in Y: \ C_y = g^{qy(0)}, C'_y = H\big(att(y)\big)^{qy(0)})$$

• Decrypt (Consumer) ({P P , C T , S KA }→M). The decryption algorithm takes as input the public parameters PP, the cipher text CT with a PT associated, and a private key SKA. If the set A satisfies the policy PT, consumer will be able to decrypt CT with SKA.

The decryption procedure is specified as a recursive algorithm. First is defined recursive algorithm DecryptNode(CT, SK,x) that takes as input a cipher text CT = (T , $\tilde{C}$,C, $\forall_y \in$ Y : C_y,C'_y), a private key SK, which is associated with a set S of attributes, and a node x from T. If the node x is a leaf node then we let i = att(x) and defined as follows: If i ∈ S, then

$$DecryptNode(CT, SK, x) = \frac{e(D_i, C_x)}{e(D'_i, C'_x)}$$
$$= \frac{e(g^r \cdot H(i)^{ri}, h^{qx(0)})}{(g^r \cdot H(i)^{qx(0)})}$$
$$= e(g,g)^{rq_x(0)}$$

If i ∈ S is not the case, then we define DecryptNode(CT, SK,x) = ⊥. Now consider the recursive case when x is a non-leaf node. The algorithm DecryptNode(CT, SK,x) then proceeds as follows: For all nodes z that are children of x, it calls DecryptNode(CT, SK,z) and stores the output as F_z. Let S_x be an arbitrary k_x-sized set of child nodes z such that $F_z \neq \perp$. If no such set exists then the node was not satisfied and the function returns $\perp$. Otherwise, compute

$$F_x = \prod_{z \in S_x} F_z^{\Delta_{i,s'x}(0)},\ where\ i = index(z);\ \tilde{S} = \{index(z): z \in \widetilde{S_x}\}$$

$$= \prod_{z \in S_x} (e(g,g)^{r \cdot q_z(0)})^{\Delta_{i,s'x}}$$

$$= \prod_{z \in S_x} (e(g,g)^{r \cdot q_{parent(z)}(index(z))})^{\Delta_{i,S'x}(0)}$$

$$= \prod_{z \in S_x} (e(g,g)^{r \cdot q_x(i) \cdot \Delta_{i,s'x}}$$

$$= e(g,g)^{r \cdot q_x(0)}$$

Now that the function DecryptNode is defined, is possible to define the decryption algorithm. The algorithm begins by simply calling the function on the root node R of the tree T. If the tree is satisfied by S then set A = DecryptNode(CT, SK, r) = $e(g,g)^{rqR(0)} = e(g,g)^{rs}$. The algorithm now decrypts by computing

$$\tilde{C} = (e(C,D)/A) = \tilde{C}(\ e\ (h^s, g^{(\alpha+r)/\beta})/e(g,g)^{rs} = M$$

4.3 Secure Data Dissemination with CP-ABE for IoT

Following the previous description, the next figure shows the application of a CP-ABE scheme for the well-known publish/subscribe model. In addition to smart objects acting as producers and consumers, as well as the attribute authority which is needed by CP-ABE, this model requires a special node known as Broker. This entity is in charge of receiving data from producers to be published to consumers who have expressed interest over certain information. Depending on the scenario to be considered, the broker functionality may be provided by another smart object, or a central platform provided by the city council in the context of smart cities.

Before data sharing among smart objects can take place, it is necessary that the system participants are in possession of the cryptographic material which is required for a secure information exchange based on CP-ABE. For this purpose, the attribute authority (AA) distributes the public parameters (common for all participants) as well as private keys (each one of them associated with a set of attributes). This process could be usually performed during an offline stage, probably in the course of the bootstrapping phase. Furthermore, consumers must subscribe their interest to the broker in order to get data from producers. When a producer decides to publish certain information, it sends a message consisting of a cipher text CT and a label Tag. CT is the result of the execution of Encrypt algorithm, which was previously described. In this example, the policy ($a1 \wedge a2$) means a subscriber which receives the message can decrypt CT only in the case its private key is associated at least with the set of attributes a1 and a2. Additionally, Tag is used by the broker to disseminate CT to subscribers whose interest matches the Tag value.

An example of this kind of situation can be envisioned in the context of smart cities with a data sharing platform at the city council to be leveraged by citizens. In this scenario, consider a family in which children go to school every day. Their smartphones, equipped with different sensors such as GPS, provide their position in real-time acting as information producers. To ensure safety of children, this data should be only known by their parents or relatives but not by any other entity or user. Thus, the location information can be encrypted with the (″parent″ ∨“relative″) sharing policy and uploaded to the platform to be accessed by authorized users.

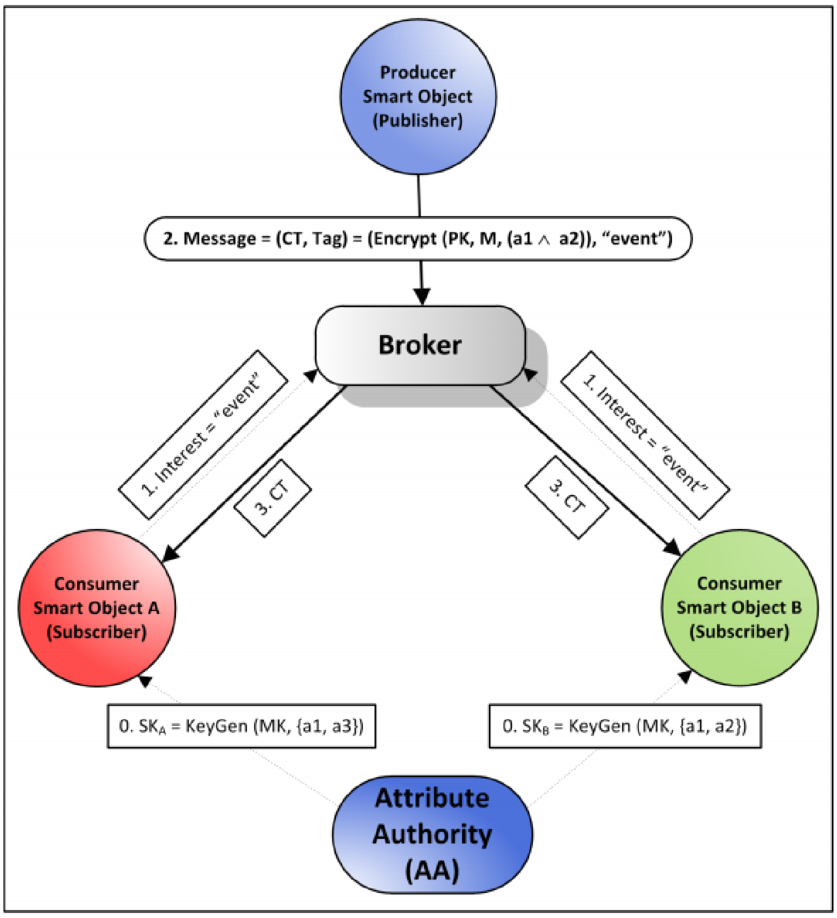

Producer Smart Object (Publisher)
2. Message = (CT, Tag) = (Encrypt (PK, M, (a1 ∧ a2)), "event")
Broker
1. Interest = "event"
3. CT
1. Interest = "event"
3. CT
Consumer Smart Object A (Subscriber)
Consumer Smart Object B (Subscriber)
0. SK$_A$ = KeyGen (MK, {a1, a3})
0. SK$_B$ = KeyGen (MK, {a1, a2})
Attribute Authority (AA)

Figure 13: Publish Subscribe Based CP-ABE.

The main advantages of this approach arise from the application of the CP-ABE scheme to the publish/subscribe model. On the one hand, smart objects acting as data producers can share information with consumers which do not need to disclose their identity. Additionally, data confidentiality is preserved even in the presence of a compromised broker. On the other hand, compared to classical pull approaches, the asynchronous communication of this model can result in energy saving, a relevant feature in scenarios where resource-constrained devices are used. Moreover, this schema is gaining attention and emerging protocols such as MQTT [68] are being proposed to be used on IoT scenarios.

The main assumption of the previous model is the existence of a broker as central component responsible for data dissemination. However, given the pervasive and dynamic nature of the scenarios envisioned by IoT community, this may represent a significant limitation in situations where data sharing is required and the setup of a broker is not possible. In this direction, in recent years, an alternative model is gaining collection, the so-called opportunistic IoT[69]. Unlike the previous approach, this solution leverages opportunistic contact and ad hoc connection among devices, simulating the way people communicate in the physical world. Opportunistic communities are formed spontaneously, particularly based on physical proximity, and using short-range communications technologies without infrastructure. Due to the inherently mobile nature of smart objects (such as mobile phones or vehicles), this model has an important interest to be exploited in IoT. For example, Bob can create an opportunistic network of mobile phones when he goes into a restaurant to share useful information with others, or a combination of vehicles, geographically close at a certain point of a highway, can exchange information about an incidence on the road.

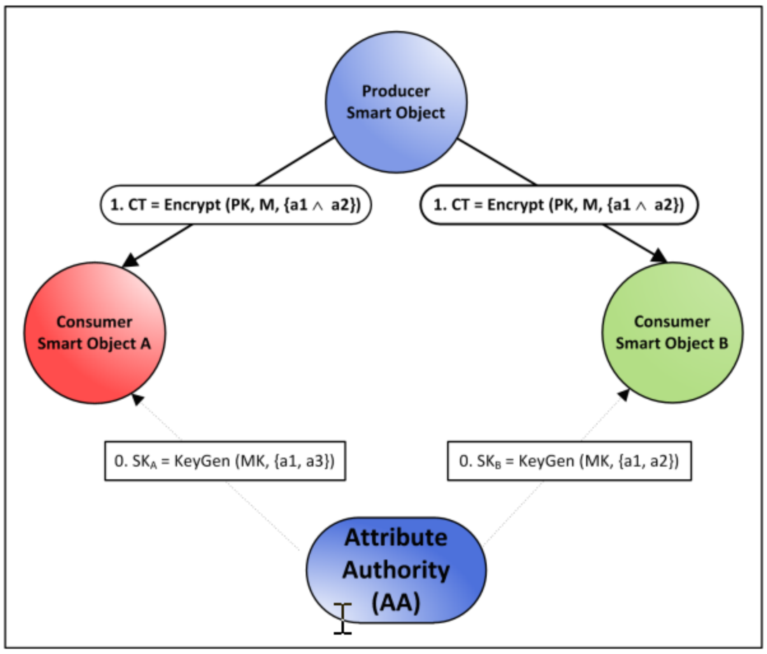

Producer Smart Object
1. CT = Encrypt (PK, M, {a1 ∧ a2})
1. CT = Encrypt (PK, M, {a1 ∧ a2})
Consumer Smart Object A
Consumer Smart Object B
0. SK_A = KeyGen (MK, {a1, a3})
0. SK_B = KeyGen (MK, {a1, a2})
Attribute Authority (AA)

Figure 14: Opportunistic sharing based on CP-ABE.

The previous figure shows the application of CP -ABE for sharing information in an opportunistic IoT scenario. An additional advantage over the previous approach is that smart objects do not need to send additional messages to establish a publish/subscribe channel with the broker. Therefore, the delay necessary for the information dissemination is smaller, which may result in an added value in situations like those described above.

An additional requirement to be considered for both push-based approaches is the usage of techniques to ensure the validity of the source and quality of the information disseminated. While CP-ABE allows preserving the privacy of consumers, it is necessary to ensure that the information being produced do not proceed from malicious or compromised nodes. For this purpose, traditional digital signature schemes can be used.

However, the use of highly sensitive personal information such as user location could still compromise the privacy of producers. Therefore, privacy-enhancing techniques used on pull-based approaches may be required so that data sharing among smart objects is carried out in a secure and privacy-preserving way. For example, a producer could sign a message by using an ABS [70] schema in order to hide his real identity when sending data. In this way, data consumers could be sure that the information comes from a trusted source, while producer privacy would be preserved.

4.4 CP-ABE/IBE Sticky Policies based dissemination for IoT

The generated information in an Internet of Things (IoT) scenario may have very strict security requirements and so, may need an end-to-end encryption method. The IoT device (data producer) may want to share his data with other entities and specify explicitly who has access to it. Encrypting the information with different keys for different ends may hinder the overall performance, mainly in an IoT scenario where a sensor may need to share his data with hundreds of entities.

A mechanism for sending information, using the concept of sticky policies, will also allow the IoT device to store the information in an untrusted environment. Through the use of cryptographic techniques, a link is established between the encrypted information and their access control policies. The system ensures that any third party trying to access the information has to fulfil the set of rules defined by the owner of the information.

We will provide a scalable and granular information access control mechanism, allowing:

- End-to-end information encryption
- Information access to specific elements within a group
- Information secure storage

4.4.1 Description

Sticky policies is a policy based encryption scheme that allows an entity to encrypt data according to a policy file, in such a way that only who meets the policy file requirements has access to the information. It also enables an entity to store data in an untrusted database in a secure way. For example, a system passes a health-care record from a hospital to a research institution and to a research team. The information might be in a form in which certain attributes such as medical results are encrypted, with an associated sticky policy describing how parts of this could be used - a patient wants his information to be released only to his doctors and requests it be deleted after three years.

Encrypting the data and linking the policies to the key or to the encrypted data enables the distribution of the information to unknown entities while ensuring that only those complying with the associated policies are able to access the information.

The deployment of such system is reasonably straightforward, as it does not require any change to either existing trusted third parties except to deal with additional policy condition checks or existing storage providers. If the storage providers are used to store data, an authenticated reference is passed around instead of the data (see[28])

The policies are defined in XACML 3.0 which allows the expression of very complex rules and various types of Access Control such as: Identity Based Access Control (IBAC), Attribute Based Access Control (ABAC) or Role Based Access Control (RBAC). Proposed implementation to support ABAC only includes simple rules that allow a user to define what attributes a consumer must possess to access the data. For example, a consumer can define that only the medical staff has access to his blood pressure.As policies written in XACML are very long, there have been attempts to implement XACLM using JSON. Such implementation allows maintaining the properties of XACML while reducing the size of policy files.

As stated in[28],Trusted Authorities *(*TAs) provide assurance by keeping track of promises that the involved parties make to access data, along with controlling access to such data. The TAs' role may be integrated with other functionality, such as being a consumer organization, a certification authority (CA), or a well-known organization. The TA's role can also be performed by a client-side software component or service that is under the control of users or other parties. It can also be achieved using distributed components or a peer-to-peer mechanism.

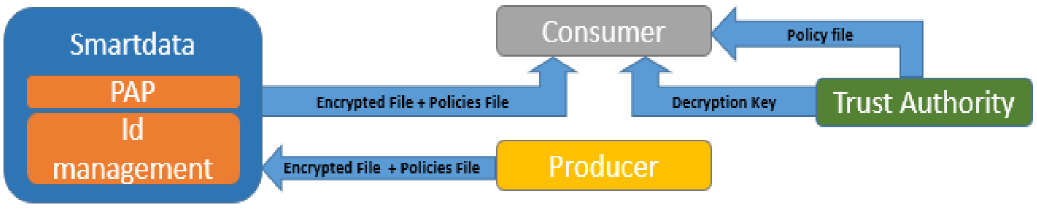
The Sticky Policies encryption scheme is implemented using pairing bases cryptography. A pairing is a function that takes as input two points on an elliptic curve and outputs an element of some multiplicative abelian group (G1 x G2 → Gt). This function satisfies some special properties, the most important of which is bilinearity. Pairing based cryptography allows entities to derive their keys from their identities (or some attributes) and the corresponding keys are generated from some trusted party (http://alexandria.tue.nl/extra2/afstversl/wsk-i/maas2004.pdf). Pairings can be classified as symmetric (G1 = G2) and asymmetric (G1 ≠ G2) and according to their properties they can be of three (or more) different types:

- Type 1: G1 = G2;
- Type 2: G1 ≠ G2 and there is an efficiently computable homomorphism;
- Type 3: G1 ≠ G2 and there is no efficiently computable homomorphism between G1 and G2.

4.4.2 Implementation

Our Sticky Policies implementation uses the type 1 pairing, constructed on the curve $y^2 = x^3 + x$ over the field $F_q$for some prime q = 3 mod 4. This pairing type is easier to implement, but it needs bigger keys in order to provide the same security level from the other types.

Currently the implementation of type 3 pairingsis being evaluated. This type offers better performance and flexibility for high security parameters.In the implementation, it is possible to identify the following entities and flows:



1. Producer: The entity that produces the data to be encrypted;
2. Consumer: It obtains the data and needs to decrypt it;
3. Sticky Policies Service: Service responsible for data encryption/decryption. This service may be:
    a. Internal: If the producer is a non- constrained device, it may be able to encrypt/decrypt the information himself;
    b. External: In the case of constrained devices that cannot encrypt/decrypt data, this service can be deployed as an external service in a more powerful device;
4. Trust Authority (TA): This entity has three main functionalities:
    a. Generate public parameters that are used by the producer to generate keys from policy files;
    b. It serves as a Policy Decision Point (PDP) to verify consumers' authorization;
    c. Generates decryption keys from policy files.
5. Policy Administration Point (PAP): Creates policies with a limited types of rules (still in development phase and slightly immature)
6. Id management: Entity used for authentication purposes; TA uses this entity to obtain consumers' information in order to evaluate authorization.

4.4.2.1 Message Encryption

Proposed message encryption flows are presented and described below as well as the associated sequence diagram.

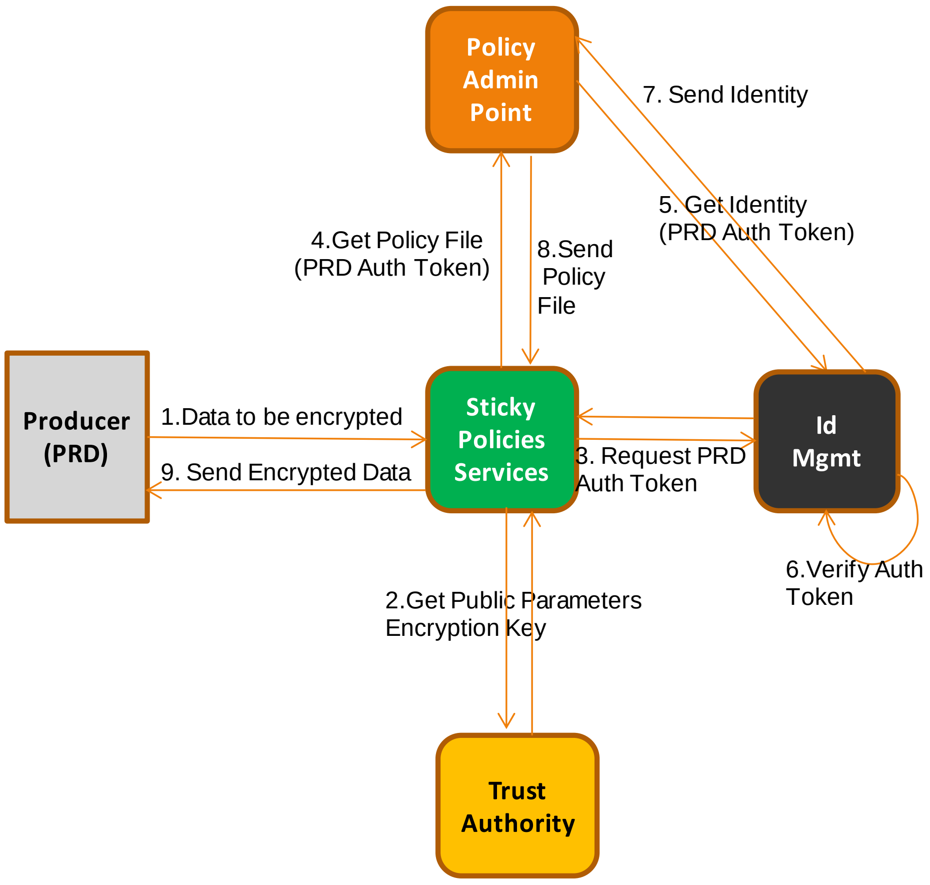


Figure 15: Message Encryption flows with sticky policies.

1. The Producer (PRD) sends the unencrypted data to the Sticky Policies Service;
2. The Sticky Policies Service gets the Public Parameters from the TA;
3. The Sticky Policies Service requests the Auth Token from the Id Management;
4. The Sticky Policies Service sends the PRD Auth Token to the PAP;
5. The PAP sends the PRD Auth Token to the Id Management;
6. The Id Management verifies the token validity;
7. The Id Management sends the PRD identity to the PAP;
8. The PAP sends the policy file to the Sticky Policies Service;
9. The Sticky Policies Service sends the encrypted data to the PRD:
   a. Generates a symmetric cipher;
   b. Encrypts the data with the symmetric key;
   c. Encrypts the symmetric key with the sticky policies key;

Notes:

- Proposed implementation uses AES as a symmetric cipher, this is one of the most popular symmetric ciphers among security providers.
- The sticky policies key is obtained using the public parameters and the policy file using pairing based cryptography (Java library – jPBC.)
- The Auth token is created using Oauth v2.0
- The sticky policies service is a Java library that can be (preferably) installed in the producer's device or, if it is a computational constrained device, available as a private network service.
- The entities requiring policy files and keys for encryption should register in the Id Management.
- All API requests must have an Auth token.

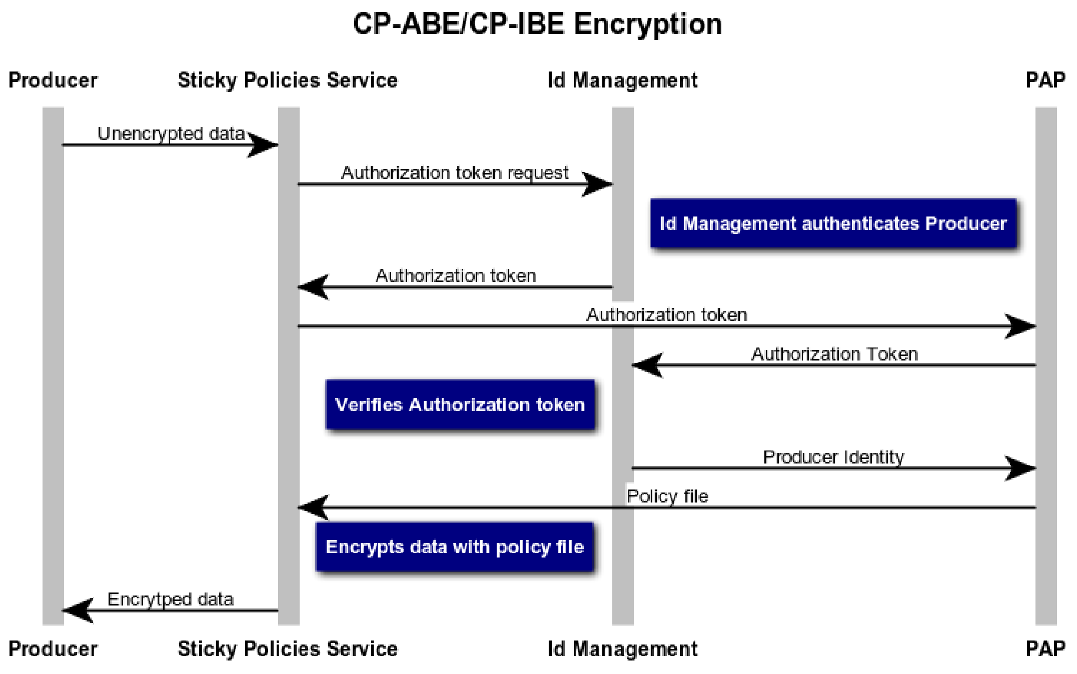


Figure 16: Sticky policies Sequence Diagram.

4.4.2.2 Message Decryption

Proposed message decryption flows are presented and described below as well as the associated sequence diagram:

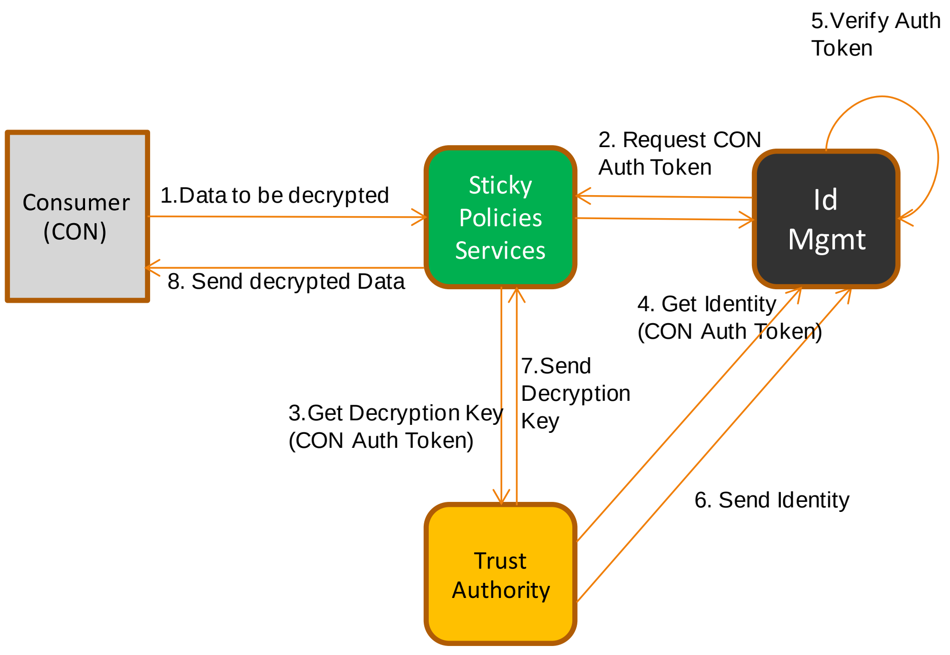


Figure 17: Message decryption flow with sticky policies.

1. The Consumer (CON) sends the encrypted data to the Sticky Policies Service;
2. The Sticky Policies Service obtains the CON Authorization token from the Id Management;
3. The Sticky Policies Service sends the Auth token and the policy file to the Trusted Authority (TA);
4. The TA sends the Auth Token to the Id Management;
5. The Id Management verifies the Auth Token Validity;
6. The Id Management sends the CON's identity to the TA;

7. The TA sends the Decryption Key to the Sticky Policies Service;
    a. The TA will verify if the consumer meets the requirements specified in the policy file;
    b. The TA generates a decryption key based on the policy file received;
8. The Sticky Policies Service sends the decrypted data to the CON:
    a. Extracts the symmetric key from the received data;
    b. Decrypts the symmetric key using the key sent from the TA;
    c. Decrypts the data using the decrypted symmetric key;

Note:

- During the decryption process, the TA also acts as a Policy Decision Point (PDP). Java library Balana is responsible for making the decisions based on the policy file received and the attributes of the consumers.

4.5 Encryption scheme for data minimization

As an instance of remotely managed IoT or M2M systems, building management systems monitor and control several systems in a building such as heating, air condition, lighting, mechanical systems and various kind of alarms. To be able to manage the building smart and mostly automated, the system collects data from many sensors throughout the building. A modern and efficient building management system makes generally use of the "Cloud" which offers cheap data storage, but also makes it possible that access is shared and the management services are booked on demand. Thus, it is very likely, that building data is kept outside the building and even outside the building management organization at cloud storage or cloud service providers.

Several security and privacy problems arise from such a system. The data collected is strongly correlated with the actions of people, so privacy issues arise and data protection law becomes relevant. This is not only the case if the building is a home for people, but also in office environments, data collected from the building reveals personal preferences and habits of employees. From the perspective of a company, the collected data may allow to derive confidential information, e.g. the work load of groups and departments.

It is also important to note that when tackling data analysis problems, the typical approach is to collect the data initially as a whole, and later focus on the relevant subset of data. We consider this approach a source of threats, in particular considering outsourcing of data analysis to third parties. Therefore we make the case that while data on e.g. facility / room conditions in certain events should be analysed, only the data relevant for the event occurrence should be available to minimize data exposure and to protect stakeholders' interests.

4.5.1 Encryption Techniques:

This section describes encryption techniques based on symmetric encryption primitives, thus in particular suitable for encryption on restricted sensors or for high-throughput. Three variants are described: time and range encryption add different features compared to attribute-based encryption based on the use of a hash-chain to model a scale. The generic context-based encryption realises a subset of attribute-based encryption.

4.5.1.1 Encryption based on time

In time-based encryption, the data collector encrypts a message dependent of the current time t. An epoch $e = [t_0, \dots, t_n]$ and within the epoch e it is possible to create a decryption key k_T for any time interval $T = [t_i, \dots, t_j] \subseteq e$ such that k_T can exactly decrypt only the ciphertexts$c$ that were created at a time $t \in T$. Practically, an epoch might be one day, but could also be longer or shorter. As only one interval key per epoch can be created, the price for the flexibility in requesting multiple intervals that comes with short epochs is the additional key overhead.

$$\begin{tikzcd}
t_0 \arrow[rrr, mapsto] & & & t_n = t_0 + n \cdot \delta t \\
\mathrm{MK}_{t_0} = \mathrm{KDF}(\mathrm{MK}, t_0, \text{‘time’}) \arrow[d, equal] & & & \\
g_0 \arrow[r, "H"] & g_1 \arrow[r, "H"] & \cdots \arrow[r, "H"] & g_n \\
& & & \mathrm{MK}_{t_n} = \mathrm{KDF}(\mathrm{MK}, t_n, \text{‘time’}) \arrow[d, equal] \\
h_0 & h_1 \arrow[l, "H"'] & \cdots \arrow[l, "H"'] & h_n \arrow[l, "H"']
\end{tikzcd}$$

Figure 18: Generation of the Hash Chains g and h given the master key MK and an epoch from t_0 to t_n.

The specification of the agorithms Setup, Encrypt, Decrypt is as follows:

Setup:

To setup the scheme, an epoch e is defined, given by its start date t_0, a time period δt and a length $n \in \mathbb{Z}$. The end time of the epoch is then given as $t_n = t_0 + n \cdot \delta t$.

Given the master key MK, two keys for the scheme are derived, denoted by $MK_{t_0} = KDF(MK, t_0,' time')$ and $MK_{t_n} = KDF(MK, t_n,' time')$.

Two hash chains of length n are computed. The elements are computed as follows:

$$g_i = H^i(MK_{t_0}) \text{and} h_{n-i} = H^i(MK_{t_n}) \text{ for } i = 0, \dots, n,$$

where$H^i(\cdot)$ denotes$i$-fold application of the hash function H, thus the first elements are given as $g_0 = MK_{t_0}$ and $h_n = MK_{t_n}$.

For each time t_i in the epoch, the encryption key is set as $k_i = g_i \oplus h_i$.This guarantees that the relevant element for the time needs to be known from both hash chains. This is the case for encryption, as the party who encrypts will know the master key that is used to compute the hash chains.

For decryption, the party decrypting is only given one element from each hash chain. A decryption key is given by $k_T = (g_i, h_j)$with $i \leq j$. This provides the ability to compute the hash chain g starting from time i and the hash chain h starting with j backwards. This gives the party the ability to decrypt ciphertexts in the interval $T = [t_i, \dots, t_j]$.

Encrypt:

Given the key MK the plaintext m is at time t_i is encrypted as

$$Enc_{MK}^{t_i}(m) = E_{k_i}(m),$$

for a standard secure encryption scheme such as AES in a suitable, preferably authenticated mode, e.g. GCM.

Decrypt:

Given the ciphertext$c$ from time t_c a key $k_T = (g_i, h_j)$ is needed, with $i \leq c \leq j$.

From g_i and h_j it is possible to compute g_c and h_c by completing the hash chain as described in the setup protocol. With the obtained key $k_c = g_c \oplus h_c$it is possible to decrypt the ciphertext c to obtain m.

$$Dec_{k_T}^{t_C}(c) = D_{k_c}(c) = m.$$

Again, D is the decryption of the standard cipher used, e.g. AES.

4.5.1.2 Encryption based on the range of the plaintext

This variant builds on the same principles as time-based encryption, but instead of time, the keys are computed dependent on the data to be encrypted itself. The purpose is to release partial data based on the encrypted values, e.g. to release only outliers. The plaintext space is modelled as a totally ordered finite set X, typically a discrete numerical scale. Let n denote the size of X and $m_0, \dots, m_{n-1}$the elements of in ascending order. The encryption offers the possibility to produce a key pair k_{in}, k_{out} for an interval $[a, b] \subseteq X$. The keys have the property, that k_{in} only decrypts the ciphertexts where $c = E(m)$ and $m \in [a, b]$.

In addition, the encryption method offers the possibility to decrypt the outliers only by releasing only k_{out} i.e. k_{out} only decrypts $c = E(m)$ and $m \notin [a, b]$.

Setup:

Given the master key MK and the plaintext space (m_0, ..., m_{n-1}) four keys for the scheme are derived, denoted by

$$\begin{aligned}
\mathrm{MK}^I_{m_0} &= \mathrm{KDF}(\mathrm{MK}, m_0, `in', `range'),\\
\mathrm{MK}^I_{m_{n-1}} &= \mathrm{KDF}(\mathrm{MK}, m_{n-1}, `in', `range'),\\
\mathrm{MK}^O_{m_0} &= \mathrm{KDF}(\mathrm{MK}, m_0, `out', `range'),\\
\mathrm{MK}^O_{m_{n-1}} &= \mathrm{KDF}(\mathrm{MK}, m_{n-1}, `out', `range').
\end{aligned}$$

Starting with each key, a hash chain is computed. Encrypting values in order to release only values within an interval is done in a similar way to time-based encryption by computing $g_i^I = H^i(MK^I_{m_0})$ and $h^I_{n-1} = H^{n-1}(MK^I_{m_{n-1}})$.

New key parts are required for releasing values outside of an interval. For this reason, two more hash chains are computed as

$$g_i^O = H^i(MK^O_{m_0})\text{and}h^O_{n-i} = H^i(MK^O_{m_{n-1}}). \backslash\backslash$$

For all hash chains$H^i(\cdot)$ denotes i-fold application of the hash function H, thus the first elements are given as $g_0^O = H(MK^O_{m_0})$ and $h_n^O = H(MK^O_{m_{n-1}})$.

For each $m_i \in X$ three derived keys are computed by $k_i^I = g_i \oplus h_i$, $k_i^{O>} = g_i\$$ and $k_i^{O<} = h_i$.

Encrypt:

Given the key *MK*, from which all of the keys listed in setup can be bootstrapped, the value *m* is encrypted as

$$Enc_{MK}(m) = \left(E_{k_i^I}(m), E_{k_i^{O<}}(m), E_{k_i^{O>}}(m)\right),$$

where$i$ is the index of the bucket where the elment m is in.

Decrypt:

As the encryption key dependents on the message, any information about the key used would also leak information about the message. Thus, given ciphertext$(c^I, c^{O<}, c^{O<})$ consisting of three components, a decryption attempt has to be made with all possible keys without knowing if it will be successful. Depending if values within an interval or outliers should be decrypted, the decryption procedure is as follows:

To decrypt within an interval, a key $k_T = (g_i, h_j)$ is required. From $g_i$and h_j, it is possible to compute the required keys g_ℓ and h_ℓ by completing the hash chain as described in the setup protocol for all $i \leq \ell \leq j$. As the encryption scheme is authenticated, decryption will only succeed, if c^I is indeed encrpyted with the used key. Note that if all decryption attempts based on $k_T = (g_i, h_j)$ fail, the decrypting party learns that the encrypted message is not contained within the interval T.

Outliers are decrypted in a similar way, by completing the hash chain and do a trial decryption with every element. This has to be done for both hash chains separately, where one is for smaller and one is for larger values. Again, if all decryption attempts fail, the encrypted element is not an outlier.

To increase the performance of the scheme, in particular if a large range of values should be covered, the values need to be distributed to buckets.

4.5.2 Context-based encryption

This encryption modes ties the encrypted message to context variables at the time of encryption. Unlike the other two modes considered in this paper, this encryption mode does not imply an order in the set of possible contexts. The context is given by a list of elements Y.

Setup:

Given is a finite list Y. For each element $y \in Y$, a key k_y is computed by $k_y = KDF(MK, y)$.

Encrypt:

When an element m is to be encrypted, the context y is queried. Then k_y is used to encrypt the message as

$$E^y_{MK(m)} = E_{k_y}(m).$$

Decrypt:

Given a ciphertext$c^y = E_{k_y}(m)$ with context y, and the key k_y, the message can simply be obtained by

$$m = D_{k_y}(m).$$

4.5.3 Combined encryption modes

It is possible to combine the technologies presented in the previous sections, i.e. to produce a ciphertext where a value-range and a time-range need to be specified. This is easily achieved by XORing the resulting encryption keys.

5 Proposed Scenarios

This section describes the 3 proposed frameworks for Event Detection, Access Control and Privacy in 3 different application scenarios. Each scenario shows the entities involved, the protocols employed and the sequence of data communication.

5.1 Proposed Scenario for Event Detection

The event detection scenario shows how a user is able to subscribe to a topic (i.e. moving vehicles), and get the information when vehicles enter or leave a defined area.

5.1.1 Entities involved

User: In this scenario the users want to monitor moving objects, e.g. moving vehicles in a city, and get notified in case of an event that an object enters or leaves an area defined by the user.

PrivLoc encryption proxy: A trusted server scrambles the location reports of vehicles before sending and storing them in the database.

SmartData platform: The scrambled areas of interest and scrambled location reports are stored in the database of the platform. A service enabler runs the geofencing algorithm of PrivLoc.

Objects: Objects move around the city and send location beacons to the proxy or platform

5.1.2 Protocols and technologies

PrivLoc: PrivLoc provides the algorithm for scrambled location reports, and the service enabler of performing geofencing on the scrambled data.

SmartDataPlatform: The SmartDataPlatform receives the location reports from the objects, i.e. vehicles. The Platform notifies subscribers of events.

5.1.3 Description of sequence diagram

In this scenario, a user is interested in the events that vehicles enter or leave a certain area. The user subscribes to that area using the scrambling function of the PrivLoc proxy. The proxy submits the subscription to the geofencing service enabler in the platform. From this moment, the user will be notified of events for his subscription.

During the scenario, objects which might be vehicles driving through the city constantly update their location to the PrivLoc proxy. The proxy scrambles the location report and forwards it to the service enabler of the platform. The location update consists of the most recent movement vector of the object. If the object crossed any subscribed area by any user, that user is notified by the service enabler of that event.

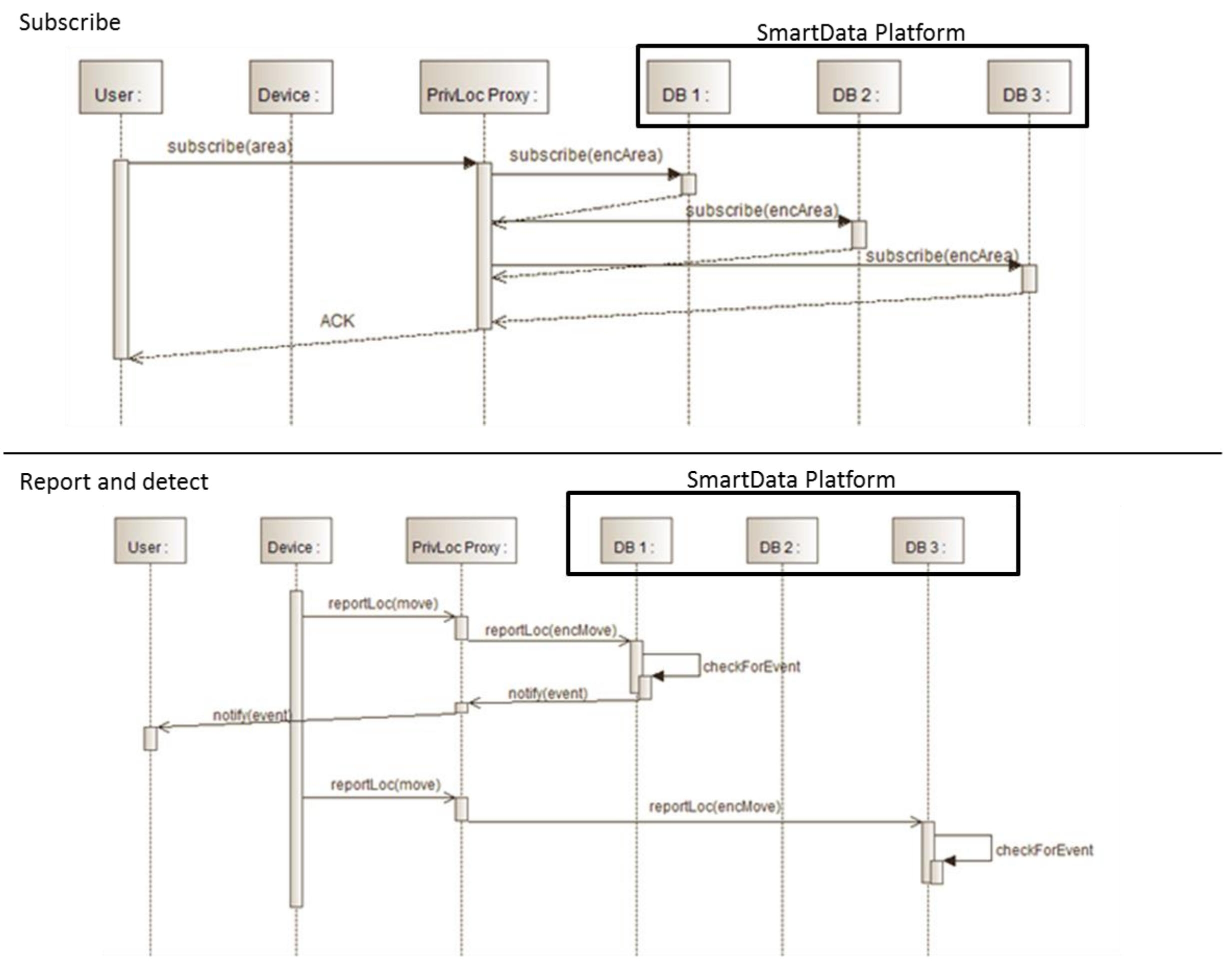


Figure 19: Scenario for Privacy-Preserving Event Detection.

5.2 Proposed Scenario for Access Control

This access control scenario shows how a user is able to retrieve information from a sensor by means of anaccess control technique. This process assumes that the phaseof bootstrapping was already done.

5.2.1 Entities involved

User: In this scenario the user intends to retrieve information from a sensor. To do that, he requests to the Capability Manager to obtain the capability token.

Capability Manager: This is the entity that processes the query from a user in order to provide a token for allowing the access to the requested information from an IoT source.

PDP: The Policy Decision Point implements XACML and gives a response to the authorization query for a capability token.

Sensor: This is the source of information and receives the capability requested by the user. The sensor verifies the sign of the token and responds the queried information.

5.2.2 Protocols and technologies

*DTLS*is used to establish a secure communication channel between all the entities in the scenario.

*CoAP*is employed to exchange the information at the application level. The CoAP messages are??secured with DTLS.

XACML is used for resolving the authorization decisionof a capability token requested by the user towards the Capability Manager.

*DCapBAC*is the distributed capability technique to generate and validate the tokens requested by user in order to access a resource of aIoTsensor.

5.2.3 Description of sequence diagram

The user sends a query (1) to the Capability Manager for a capability token. This query contains the target sensor as well as the action that is requested to be performed. For instance, the user can request the temperature of a sensor.

To do that, the user initiates a DTLS exchange (2) to establish a secure communication channel with the Capability Manager. The Capability Manger authenticates (3) the user, if the authentication is successful than the manager sends a XACML query (4) to the PDP to check the authorization of the action requested by the user. The Capability Manager receives a XACML Response (5) indicating if the query is Permitted or Denied. When the action is permitted, the Capability Manager generates the capability token (6) and sends it back to the user (8).

With the capability token, the user is able to establish a secure end-to-end communication with the sensor. If the user sendsa CoAP query without the capability token (9), thenthe sensor will deny the information access(10). To perform a secure communication, the user starts a DTLS exchange (11) with the sensor and sends a CoAPquery containing the capability token (12). Then the sensor is able to verify the sign of the capability token (13). If the token is valid, the sensor sends back the requested information (14).

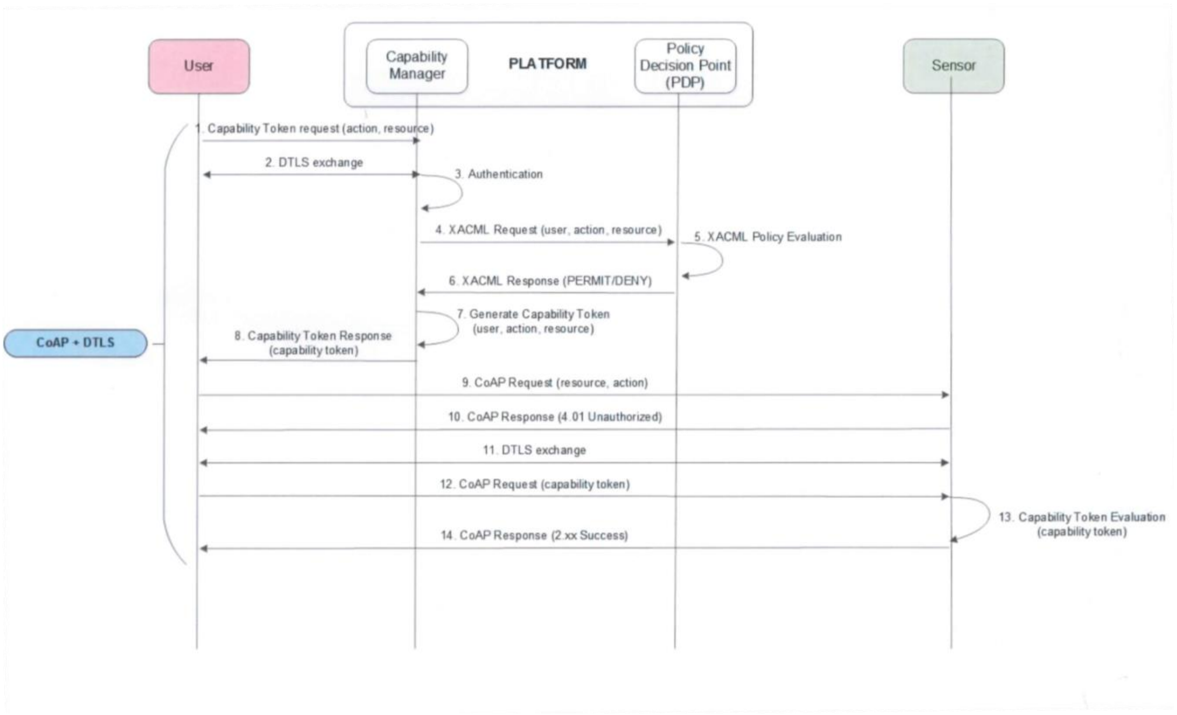

User
Capability Manager
PLATFORM
Policy Decision Point (PDP)
Sensor
1. Capability Token request (action, resource)
2. DTLS exchange
3. Authentication
4. XACML Request (user, action, resource)
5. XACML Policy Evaluation
6. XACML Response (PERMIT/DENY)
7. Generate Capability Token (user, action, resource)
8. Capability Token Response (capability token)
CoAP + DTLS
9. CoAP Request (resource, action)
10. CoAP Response (4.01 Unauthorized)
11. DTLS exchange
12. CoAP Request (capability token)
13. Capability Token Evaluation (capability token)
14. CoAP Response (2.xx Success)

Figure 20:Scenario for Access Control.

5.3 Proposed Scenario for Privacy

This subsection presents a scenario in which a user is able to subscribe to a topic and get the information from a sensor keeping a privacy preserving communication. This process assumes that the phase of bootstrapping was already done.

5.3.1 Entities involved

User: In this scenario the user intends to retrieve information from a sensor. To do that, he requests the Capability Manager to obtain the capability token.

Capability Manager: This is the entity that processes the query from a userin order to provide a token for allowing the access to the requested information from an IoT source.

PDP: The Policy Decision Point implements XACML and gives a response to the authorization query for a capability token.

Topic Manager: This entity is in charge of gathering the information from the sources of information. The topic manager acts as a broker in the Publish Subscribe scenario with CP-ABE.

Sensor: This is the source of information and receives the capability requested bythe user. The sensor verifies the sign of the token and responds the queried information.

5.3.2 Protocols and technologies

DTLSis used for securing the communication between all the entities in the scenario.

XACMLisemployed for resolving the authorization decision during the request of capability tokens from the user to the Capability Manager.

DCapBAC is the distributed capability technique to generate and validate the tokens requested by user in order to access a resource of anIoT sensor.

CP-ABE is used as a ciphering tool enabling the user to receive the information ciphered according to some policies. The user is able to decipher the information with his keys.

5.3.3 Description of sequence diagram

In this scenario, the Topic Manager is acting as a broker between the sensor and the user. First, the Topic Manager will request the Capability Manager (0) for a capability token, in order to homogenize the communication from the sensors perspective. With this capability token, the Topic Manager will send a query to the sensor for subscribing to a given topic. The generated information of the sensor for this topic is transmitted to the Topic Manager. This information is ciphered according to the appropriate policies and is sent to the users that are subscribed to the topic.

At this point the user is able to subscribe to a topic provided by the Topic Manager (1). To do that, a DTLS (2) connection is established towards the Topic Manager. The Topic Manager authenticates the user (3) and asks the PDP if the user is authorized to perform the mentioned subscription (4). If the PDP responds (5) an acceptation message, then the Topic Manager sends a confirmation to the user with the requested token.

If the subscription was successful, each time the sensor generates new information; such data is transmitted to the Topic Manager. Then the manager will cipher the information with CP-ABE according to some policies in order to send the data ciphered towards the user. The user will decipher the mentioned information with its keys.

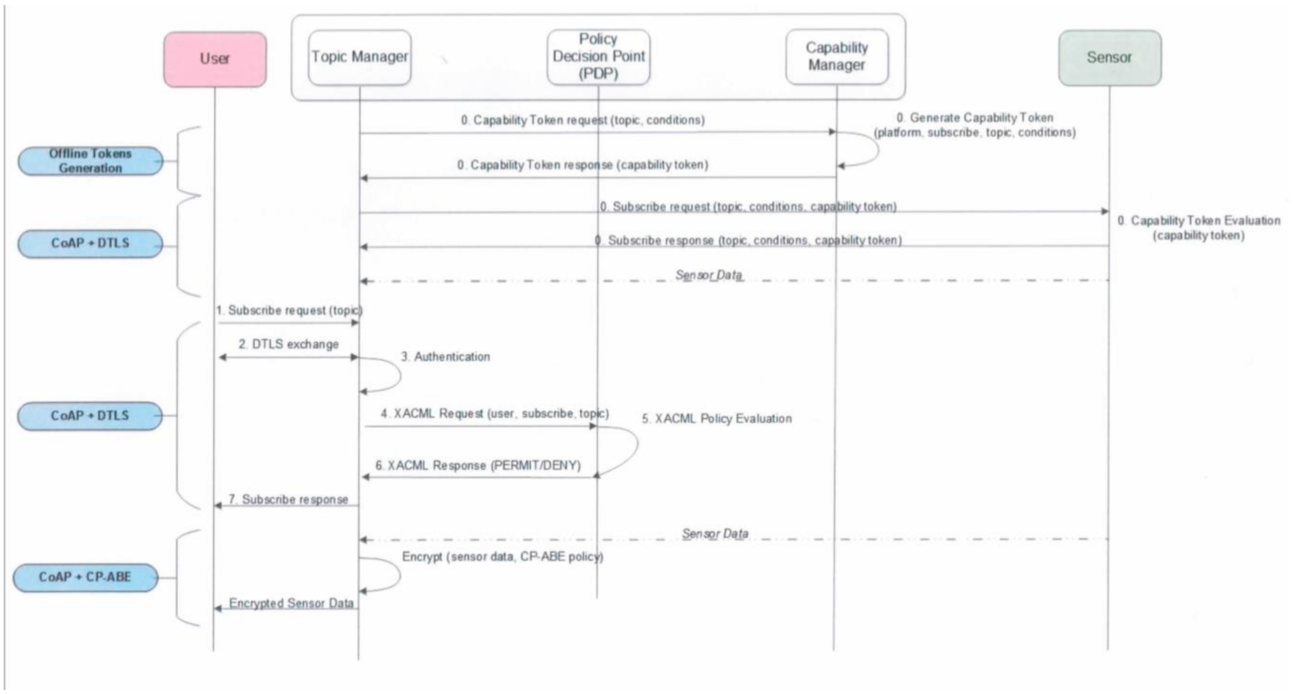

User
Topic Manager
Policy Decision Point (PDP)
Capability Manager
Sensor
Offline Tokens Generation
0. Capability Token request (topic, conditions)
0. Generate Capability Token (platform, subscribe, topic, conditions)
0. Capability Token response (capability token)
CoAP + DTLS
0. Subscribe request (topic, conditions, capability token)
0. Capability Token Evaluation (capability token)
0. Subscribe response (topic, conditions, capability token)
Sensor Data
1. Subscribe request (topic)
2. DTLS exchange
3. Authentication
CoAP + DTLS
4. XACML Request (user, subscribe, topic)
5. XACML Policy Evaluation
6. XACML Response (PERMIT/DENY)
7. Subscribe response
Sensor Data
CoAP + CP-ABE
Encrypt (sensor data, CP-ABE policy)
Encrypted Sensor Data

Figure 21: Scenario for Privacy.

6 Conclusions

In this deliverable wepresent 3 frameworks of SMARTIE project to enable Event Processing, Access Control and Privacy functionalities. First, Event Processing framework explores techniques on how to process and detect certain events from encrypted data. Second, Access Control framework covers the lifecycle of the things that are incorporated into the IoT environment and provides a way of exchanging information in a secure manner from the things, as well as from the user perspective. Third, Privacyframework proposes how to enable the users to express interest on certain kinds of information and obtain that information without the user or the producer of that information compromising their privacy. Finally we present an application scenario for each framework that exemplifies the proposed solutions in concrete use cases. These secure solutions wereextendedstandardIoT protocols and technologies (i.e. COAP, DTLS, etc.) in order to enable their use in real world scenarios.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," Elsevier Computer Networks, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] V. Mayer-Scho ̈nberger and K. Cukier, Big data: A revolution that will transform how we live, work, and think. Houghton Mifflin Harcourt, 2013.

[3] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, and A. Oliveira, Smart cities and the future internet: Towards cooperation frameworks for open innovation. Springer, 2011.

[4] N. Kushalnagar, G. Montenegro, and C. Schumacher, "Ipv6 over low- power wireless personal area networks (6lowpans): overview, assump- tions, problem statement, and goals," RFC4919, August, vol. 10, 2007.

[5] A. J. Jara, M. A. Zamora, and A. Skarmeta, "Glowbalip: An adaptive and transparent ipv6 integration in the internet of things," Mobile Information Systems, vol. 8, no. 3, pp. 177–197, 2012.

[6] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (coap)," IETF RFC 7252, vol. 10, June 2014.

[7] A. J. Jara, A. C. Olivieri, Y. Bocchi, M. Jung, W. Kastner, and A. F. Skarmeta, "Semantic web of things: an analysis of the application semantics for the iot moving towards the iot convergence," International Journal of Web and Grid Services, vol. 10, no. 2, pp. 244–272, 2014.

[8] C. P. Mayer, "Security and privacy challenges in the internet of things," Electronic Communications of the EASST, vol. 17, 2009.

[9] C. M. Medaglia and A. Serbanati, "An overview of privacy and security issues in the internet of things," in The Internet of Things. Springer, 2010, pp. 389–395.

[10] D. Miorandi, S. Sicari, F. Pellegrini, and I. Chlamtac, "Internet of Things: Vision, Applications & Research Challenges," Ad Hoc Networks, vol. 10, no. 7, pp. 1497–1516, September 2012.

[11] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," Computer Networks, vol. 57, no. 10, pp. 2266–2279, 2013.

[12] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security challenges in the ip-based internet of things," Wireless Personal Communications, vol. 61, no. 3, pp. 527–542, 2011.

[13] H. Yu, J. He, T. Zhang, P. Xiao, and Y. Zhang, "Enabling end-to- end secure communication between wireless sensor networks and the internet," World Wide Web, pp. 1–26, 2013.

[14] L. S. Committee et al., "Part 11: wireless lan medium access control (mac) and physical layer (phy) specifications. ieeestd 802.11 i-2004," IEEE ComputSoc, 2004.

[15] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz et al., "Extensible authentication protocol (eap)," RFC 3748, June, Tech. Rep., 2004.

[16] C. Rigney, S. Willens, A. Rubens, and W. Simpson, "Rfc 2865: Remote authentication dial in user service," Internet Society (Jun. 2000), 2000.

[17] E.Rissanen,"extensibleaccesscontrolmarkuplanguage(xacml)version3.0 oasis standard," 2012.

[18] J. L. Hernandez-Ramos, A. J. Jara, L. Mar ́ın, and A. F. Skarmeta,"Dcapbac: Embedding authorization logic into smart things through ecc optimizations," International Journal of Computer Mathematics, no. just-accepted, pp. 1–22, 2014.

[19] A.Bassi,M.Bauer,M.Fiedler,T.Kramp,R.vanKranenburg,S.Lange, and S. Meissner, "Enabling things to talk," 2013.

[20] S. Babar, P. Mahalle, A. Stango, N. Prasad, and R. Prasad, "Proposed security model and threat taxonomy for the internet of things (iot)," in Recent Trends in Network Security and Applications. Springer, 2010, pp. 420–429.

[21] A. F. Skarmeta, J. L. Hernandez-Ramos, and M. Moreno, “A decen- tralized approach for security and privacy challenges in the internet of things,” in Internet of Things (WF-IoT), 2014 IEEE World Forum on. IEEE, 2014, pp. 67–72.

[22] R. Roman, P. Najera, and J. Lopez, “Securing the internet of things,” Computer, vol. 44, no. 9, pp. 51–58, 2011.

[23] R. H. Weber, “Internet of things–new security and privacy challenges,” Computer Law & Security Review, vol. 26, no. 1, pp. 23–30, 2010.

[24] P. Porambage, C. Schmitt, P. Kumar, A. Gurtov, and M. Ylianttila, “Pauthkey: A pervasive authentication protocol and key establishment scheme for wireless sensor networks in distributed iot applications,” vol. 14, 2014.

[25] E. Rescola and N. Modadugu, “Rfc 4347: Datagram transport layer security (dtls),” Request for Comments, IETF, 2006.

[26] A. Liu and P. Ning, “Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks,” in Information Processing in Sensor Networks, 2008. IPSN’08. International Conference on. IEEE, 2008, pp. 245–256.

[27] B. Sarikaya, Y. Ohba, R. Moskowitz, Z. Cao, and R. Cragie, “Security bootstrapping solution for resource-constrained devices,” IETF Internet Draft, draft-sarikaya-core-sbootstrapping-05, 2012.

[28] R. Hummen, J. Hiller, M. Henze, and K. Wehrle, “Slimfit—a hip dex compression layer for the ip-based internet of things,” in Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on. IEEE, 2013, pp. 259–266.

[29] D. Forsberg, Y. Ohba, B. Patil, H. Tschofenig, and A. Yegin, “Rfc 5191 protocol for carrying authentication for network access (pana),” Network Working Group, 2008.

[30] “8802-1x-2013,” 2013. [Online]. Available: http://dx.doi.org/10.1109/ ieeestd.2013.6679204

[31] O.Garcia-Morchon,S.Keoh,S.Kumar,P.Moreno-Sa ́nchez,F.Vidal- Meca, and J. Ziegeldorf, “Securing the IP-based internet of things with HIP and DTLS,” in Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks. ACM, 2013, pp. 119–124.

[32] P. M. Sanchez, R. M. Lopez, and A. F. G. Skarmeta, “Panatiki: A network access control implementation based on pana for iot devices,” Sensors, vol. 13, no. 11, pp. 14 888–14 917, 2013.

[33] F. Bersani and H. Tschofenig, “Rfc 4764-the eap-psk protocol: A pre-shared key extensible authentication protocol (eap) method,” IETF Network Working Group, Tech. Rep., 2007.

[34] J. Liu, Y. Xiao, and C. L. P. Chen, “Authentication and Access Control in the Internet of Things,” in Proc. of the 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW), Macau, China. IEEE, June 2012, pp. 588–592.

[35] D. Ferraiolo, J. Cugini, and R. Kuhn, “Role-based access control (RBAC): Features and motivations,” in Proc. of 11th Annual Computer Security Application Conference, 1995, pp. 241–48.

[36] B. Ndibanje, H.-J. Lee, and S.-G. Lee, “Security analysis and improve- ments of authentication and access control in the internet of things,” Sensors, vol. 14, no. 8, pp. 14 786–14 805, 2014.

[37] N. YE, Y. Zhu, R. chuan WANG, R. Malekian, and L. Qiao-min, “An efficient authentication and access control scheme for perception layer of internet of things,” Applied Mathematics & Information Sciences, vol. 8, no. 4, pp. 1617–1624, jul 2014. [Online]. Available: http://dx.doi.org/10.12785/amis/08041

[38] E. Yuan and J. Tong, “Attributed based access control (ABAC) for web services,” in Proc. of the 12th IEEE International Conference on Web Services (ICWS), Orlando, USA. IEEE, July 2005.

[39] P. N. Mahalle, B. Anggorojati, N. R. Prasad, and R. Prasad, “Identity Establishment and Capability based Access Control (iecac) scheme for Internet of Things,” in Proc. of the 15th International Symposium on Wireless Personal Multimedia Communications (WPMC), Taipei, China. IEEE, September 2012, pp. 187–191.

[40] S. Gusmeroli, S. Piccione, and D. Rotondi, “A capability-based security approach to manage access control in the internet of things,” Math- ematical and Computer Modelling, vol. 58, no. 5-6, pp. 1189–1205, September 2013.

[41] L. Seitz, G. Selander, and C. Gehrmann, “Authorization framework for the internet-of-things,” in 2013 IEEE 14th International Symposium on ”A World of Wireless, Mobile and Multimedia Networks” (WoWMoM). IEEE, june 2013. [Online]. Available: http://dx.doi.org/10.1109/wowmom.2013.6583465

[42] D. Crockford, “RFC 4627: The application/json Media Type for Javascript Object Notation (JSON),” IETF RFC 4627, July 2006, http://www.ietf.org/rfc/rfc4627.txt.

[43] L. Marin, A. J. Jara, and A. F. Skarmeta, “Shifting primes: Optimizing elliptic curve cryptography for 16-bit devices without hardware multi- plier,” Mathematical and Computer Modelling, 2013.

[44] A. J. Jara, “Trust extension protocol for authentication in networks oriented to management (tepanom),” in Availability, Reliability, and Security in Information Systems. Springer, 2014, pp. 155–165.

[45] M. Bauer, E. Berg, F. Carrez, S. Ebers, S. Haller, T. Jacobs, S. Krco, F. Massel, G. Woysch, and R. Egan, “The internet of things initiative (iot-i). deliverable 1.5: Iot reference model white paper,” 2011.

[46] S. Ziegler, C. Crettaz, L. Ladid, S. Krco, B. Pokric, A. F. Skarmeta, A. Jara, W. Kastner, and M. Jung, Iot6–moving to an ipv6-based futureiot. Springer, 2013.

[47] “Butler. deliverable 3.2: Integrated system architecture and initial pervasive butler proof of concept,” 2013.

[48] V. Tsiatsis, A. Gluhak, T. Bauge, F. Montagut, J. Bernat, M. Bauer, C. Villalonga, P. M. Barnaghi, and S. Krco, “The sensei real world internet architecture.” in Future Internet Assembly, 2010, pp. 247–256.

[49] S. Sotiriadis, E. G. Petrakis, S. Covaci, P. Zampognaro, E. Georga, and C. Thuemmler, “An architecture for designing future internet (fi) applications in sensitive domains: Expressing the software to data paradigm by utilizing hybrid cloud technology,” in Bioinformatics and Bioengineering (BIBE), 2013 IEEE 13th International Conference on. IEEE, 2013, pp. 1–6.

[50] J. B. Bernabe, J. L. Herna ́ndez, M. V. Moreno, and A. F. S. Gomez, “Privacy-preserving security framework for a social-aware internet of things,” in Ubiquitous Computing and Ambient Intelligence. Personali- sation and User Adapted Services. Springer, 2014, pp. 408–415.

[51] L. Seitz and G. Selander, “Problem description for authorization in constrained environments,” IETF Internet Draft, draft-seitz-ace-problem- description-01, 2014.

[52] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen, “Internet key exchange protocol version 2 (ikev2),” RFC 5996, September, Tech. Rep., 2010.

[53] A. J. Jara, D. Fernandez, P. Lopez, M. A. Zamora, and A. F. Skarmeta, “Lightweight mipv6 with ipsec support,” Mobile Information Systems, vol. 10, no. 1, pp. 37–77, 2014.

[54] S. Raza, T. Voigt, and V. Jutvik, “Lightweight ikev2: A key management solution for both the compressed ipsec and the ieee 802.15. 4 security,” in Proceedings of the IETF Workshop on Smart Object Security, 2012.

[55] J. Dennis and E. V. Horn, “Programming Semantics for Multipro- grammed Computations,” Communications of the ACM, vol. 9, no. 3, pp. 143–155, 1966.

[56] L. Marin, A. J. Jara, and A. F. Gomez-Skarmeta, “Multiplication and squaring with shifting primes on openrisc processors with hardware multiplier.” J. UCS, vol. 19, no. 16, pp. 2368–2384, 2013.

[57] M. Hansen, P. Berlich, J. Camenisch, S. Clauß, A. Pfitzmann, and M. Waidner, “Privacy-enhancing identity management,” Information Security Technical Report, vol. 9, no. 1, pp. 35–44, 2004.

[58] J. Camenisch and A. Lysyanskaya, “An efficient system for non-transferable anonymous credentials with optional anonymity revocation,” in Advances in Cryptology—EUROCRYPT 2001. Springer, 2001, pp. 93–118.

[59] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in Security and Privacy, 2007. SP’07. IEEE Symposium on. IEEE, 2007, pp. 321–334.

[60] J. Camenisch and E. Van Herreweghen, “Design and implementation of the idemix anonymous credential system,” in Proceedings of the 9th ACM conference on Computer and communications security. ACM, 2002, pp. 21–30.

[61] W. Mostowski and P. Vullers, “Efficient u-prove implementation for anonymous credentials on smart cards,” in Security and Privacy in Communication Networks. Springer, 2012, pp. 243–260.

[62] A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in Advances in Cryptology–EUROCRYPT 2005. Springer, 2005, pp. 457–473.

[63] N. Ragouzis, J. Hughes, R. Philpott, E. Maler, P. Madsen, and T. Scavo, “Security Assertion Markup Language (SAML) V2.0 Technical Overview,” 2008.

[64] T. Moses, “Extensible Access Control MarkupLanguage(XACML) Version 2.0,” 2005.

[65] J. Hernandez-Ramos, A. Jara, L. Mar´ın, and A. Skarmeta, "Distributed Capability-based Access Control for the Internet of Things," Journal of Internet Services and Information Security (JISIS), vol. 3, no. 3/4, 2013.

[66] D. Boneh and M. Franklin, “Identity-based encryption from the weil pairing,” in Advances in Cryptology—CRYPTO 2001. Springer, 2001, pp. 213–229.

[67] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute based encryption for fine-grained access control of encrypted data,” in Proceedings of the 13th ACM conference on Computer and communications security. ACM, 2006, pp. 89–98.

[68] U. Hunkeler, H. L. Truong, and A. Stanford-Clark, “Mqtts—a publish/subscribe protocol for wireless sensor networks,”in Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on. IEEE, 2008, pp. 791–798.

[69] B. Guo, Z. Yu, X. Zhou, and D. Zhang, “Opportunistic iot: exploring the social side of the internet of things,” in Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference on. IEEE, 2012, pp. 925–929.

[70] H. K. Maji, M. Prabhakaran, and M. Rosulek, “Attributebased signatures: Achieving attribute-privacy and collusionresistance.” IACR Cryptology ePrint Archive, vol. 2008, p.328, 2008.

[71] Ramos, Jose L. Hernandez, Jorge Bernal Bernabe, and Antonio F. Skarmeta. "Towards Privacy-Preserving Data Sharing in Smart Environments." Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2014 Eighth International Conference on. IEEE, 2014.

[72] Hernandez Ramos, J., et al. "Towards a Lightweight Authentication and Authorization Framework for Smart Objects."

[73] http://tools.ietf.org/id/draft-marin-ace-wg-coap-eap-01.txt

[74] Griffin, Leigh, et al. "On the performance of access control policy evaluation." Policies for Distributed Systems and Networks (POLICY), 2012 IEEE International Symposium on. IEEE, 2012.

[75] Jones, M., Tarjan, P., Goland, Y., Sakimura, N., Bradley, J., Panzer, J., & Balfanz, D. (2012). JSON Web Token (JWT).

[76] Li, S. T., J. Hoebeke, and A. J. Jara. "Conditional observe in CoAP, draft-li-core-conditional-observe-04." IETF Trust (2012).

[77] Jennings, Cullen, JariArkko, and Zach Shelby. "Media types for sensor markup language (SENML)." (2012).

[78] L. Seitz and G. Selander, “Problem description for authorization in constrained environments,” IETF Internet Draft, draft-seitz-ace-problem- description-01, 2014.

[79] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, and B. Moller, “Elliptic curve cryptography (ecc) cipher suites for transport layer

[80] Jens-Matthias Bohli, Dan Dobre, Ghassan O. Karame, Wenting Li: PrivLoc: Preventing Location Tracking in Geofencing Services. TRUST 2014: 143-160

[81] John Black and Phillip Rogaway. Ciphers with Arbitrary Finite Domains. In CT-RSA 2002, Vol. 2271 of Lecture Notes in Computer Science, pages 114–130. Springer, 2002.

[82] SaikatGuha, Mudit Jain, and Venkata N. Padmanabhan. Koi: A Location-privacy Platform for Smartphone Apps. In Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation, NSDI'12, pages 14–14, Berkeley, CA, USA, 2012. USENIX Association.

[83] Hidetoshi Kido, Yutaka, Yanagisawa, and Tetsuji Satoh. An Anonymous Communication Technique using Dummies for Location-based Services. In ICPS, pages 88–97, 2005.

[84] Latanya Sweeney. Achieving K-anonymity Privacy Protection Using Generalization and Suppression. Int. J. Uncertain. Fuzziness Knowl.-Based Syst., 10(5):571–588, October 2002.

[85] Latanya Sweeney. K-anonymity: A Model for Protecting Privacy. Int. J. Uncertain. Fuzziness Knowl.-Based Syst., 10(5):557–570, October 2002.

[86] Deliverable 4.1 SMARTIE project EU

[87] Liu, Dongxi; Bertino, Elisa; Yi, Xun. **Privacy of outsourced k-means clustering**. In: *9th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2014)*; 4-6 June, 2014; Kyoto, Japan. ACM; 2014. 123-134.

[88] Christopher Wolf and Bart Preneel, "Taxonomy of public key schemes based on the problem of multivariate quadratic equations". Cryptology ePrint Archive, Report 2005/077, 12th of May 2005

[89] http://en.wikipedia.org/wiki/Davies%E2%80%93Bouldin_index

[90] http://en.wikipedia.org/wiki/Precision_and_recall