L. González-Manzano, José M. de Fuentes, Sergio Pastrana, Pedro Peris-Lopez, Luis Hernández-Encinas. PAgIoT – Privacy-preserving Aggregation protocol for Internet of Things. In Journal of Network and Computer Applications, Volume 71, 2016, Pages 59-71, ISSN 1084-8045

DOI: https://doi.org/10.1016/j.jnca.2016.06.001.

PAgIoT – Privacy-preserving Aggregation protocol for Internet of Things

L. González-Manzano [a,*], José M. de Fuentes [a], Sergio Pastrana [a], Pedro Peris-Lopez [a], Luis Hernández-Encinas [b]

[a] *Computer Security Lab (COSEC), Computer Science Department, Universidad Carlos III de Madrid, Avda, Universidad, 30, 28911 Leganes, Madrid, Spain*
[b] *Institute of Physical and Information Technologies, Spanish National Research Council (CSIC), C/Serrano, 144, 28006 Madrid, Spain*

A B S T R A C T

Modern society highly relies on the use of cyberspace to perform a huge variety of activities, such as social networking or e-commerce, and new technologies are continuously emerging. As such, computer systems may store a huge amount of information, which makes data analysis and storage a challenge. Information aggregation and correlation are two basic mechanisms to reduce the problem size, for ex-ample by filtering out redundant data or grouping similar one. These processes require high processing capabilities, and thus their application in Internet of Things (IoT) scenarios is not straightforward due to resource constraints. Furthermore, privacy issues may arise when the data at stake is personal. In this paper we propose PAgIoT, a Privacy-preserving Aggregation protocol suitable for IoT settings. It enables multi-attribute aggregation for groups of entities while allowing for privacy-preserving value correlation. Results show that PAgIoT is resistant to security attacks, it outperforms existing proposals that provide with the same security features, and it is feasible in resource-constrained devices and for aggregation of up to 10 attributes in big networks.

Keywords:
Aggregation Privacy preservation Cryptography Internet-of-things (IoT)

1. Introduction

Nowadays, information technologies are rapidly evolving and the amount of electronic information shared in networks is increasing. Moreover, large volumes of data are continuously being exchanged. In this sense, data analysis and data aggregation are valuable activities, but their implementation in distributed and untrusted networks is challenging (Rajagopalan and Varshney, 2006).

In many scenarios the aggregation of information is not straightforward. Consider for example a football stadium with a capacity of 120,000 people. How to determine how many people are more likely to receive medical attention? This would allow a better calculation of the required emergency personnel. One straightforward approach would be to retrieve information (say heart rate) from attendees' personal devices (such as smart bracelets/watches or even Implantable Medical Devices). However, privacy issues may arise since personal information is at stake. Data should be exclusively accessible to authorized users preventing eavesdropping (He et al., 2007; Wenbo et al., 2008; Chen and Lou, 2015). Moreover, data should remain unlinkable to owners (He et al., 2007; Wenbo et al., 2008; Applebaum et al., 2010). It is needed to have a privacy-preserving mechanism that aggregates the information to provide with a general view of the situation.

Even more, an aggregate view cannot be enough. For example, a fast heart rate may be less dangerous in younger people than in elderly ones. It is then necessary to enable an aggregation that allows answering questions such as *How many people have fast heart rate? And how many of them are older than 50?* We coin the term *correlatable aggregation* to refer to this need, as it involves not only having the global value, but also identifying how the different variables relate each other. Previous attempts have focused on data aggregation (Xiaoying et al., 2014; Buttyán and Holczer, 2010; Kumar et al., 2015; Ren et al., 2013; Madden et al., 2002) among collected data. Data security (mainly integrity) and privacy has also been considered, e.g. He et al. (2007) and Applebaum et al. (2010). However, to the best of the authors' knowledge, no proposal addresses the need of correlatable aggregation.

The Internet of Things (IoT) in a broad sense is a network of physical objects with the capacity to collect and exchange data. The involved objects are very diverse; cars, fridges and buildings are common examples but opportunities are immense, including food, clothes, and all the variety of living things (i.e., plants, animals and even ourselves). The objects are embedded with

* Corresponding author.
E-mail addresses: lgmanzan@inf.uc3m.es (L. González-Manzano) jfuentes@inf.uc3m.es (Jose M. de Fuentes), spastran@inf.uc3m.es (S. Pastrana), pperis@inf.uc3m.es (P. Peris-Lopez), luis@iec.csic.es (L. Hernández-Encinas).

electronic, software and network connectivity. Wireless Sensor Networks (WSNs) is only a part of the IoT (Alcaraz et al., 2010) and the two principal differences with other IoT objects (e.g., smartphones or smart meters) are the computing capabilities and the ability to install applications of third-parties. On the other hand, at the beginning, WSNs were considered as isolated networks and the Internet connectivity has been recently added, which brings some challenges (Christin et al., 2009).

Having considered the distinction between WSNs and IoT, in IoT aggregation mechanisms are an issue due to resource limitations (Gubbi et al., 2013) and scalability concerns (Hai et al., 2015). Furthermore, several attacks (such as fabrication or manipulation of data) may be carried out to frustrate this action (Xiaoying et al., 2014). Considering these issues, in this paper we propose PAgIoT, a Privacy-preserving Aggregation protocol for IoT. In PAgIoT, data is decomposed into a set of attributes which are aggregated sepa-rately. A central node (sink) queries for the value of certain attri-butes, and remainder nodes respond depending on whether they possess or not these attributes. As many nodes may have the same attribute, intermediate nodes perform aggregation to save net-work resources. In this way, PAgIoT enables gathering data con-cerning several attributes of each entity in a single operation en-suring data authenticity and privacy. PAgIoT leverages on cluster-ing to deal with large-scale scenarios. Going back to the stadium setting, it allows zoning responses so that a better personnel placement could be achieved. Furthermore, it also leverages on the Paillier cryptosystem, which enables collecting information in a privacy-preserving way. PAgIoT also enables detecting malicious manipulation of aggregated information.

The remainder of this paper is as follows. Section 2 gives the background of the Paillier cryptosystem. Section 3 introduces the model considered in PAgIoT. Section 4 describes PAgIoT, whereas Section 5 shows its evaluation. Related works are described in Section 6. Finally, Section 7 outlines conclusions and future re-search directions.

2. Background: Paillier cryptosystem

In order to deal with privacy-preserving aggregation, PAgIoT uses the Paillier cryptosystem (Paillier, 1999). Note that there are other homomorphic encryption schemes which could be similarly applied but Paillier's is chosen as a successful alternative. In particular, Benaloh scheme (Benaloh, 1986) is a feasible choice. However, it is appropriate for long blocks of input data, which is not the case herein, and it involves higher computation costs in comparison with Paillier cryptosystem (Singh and Dutta, 2014). Therefore, this section gives the reader a brief description of Paillier cryptosystem.

Paillier system is a public key cryptosystem based on the decisional composite residuosity assumption. Let p and q be prime numbers, $n = p \cdot q$, and the Euler's totient function $\phi(n) = (p-1)\cdot(q-1)$. Given $\gcd(n, \phi(n)) = 1$, the hardness of this problem is to decide whether an element in $\mathbb{Z}^*_{n^2}$ is an nth power of an element in $\mathbb{Z}^*_{n^2}$, that is, if there exists a number $y \in \mathbb{Z}^*_{n^2}$ such that $z = y^n \pmod{n^2}$ (Stern, 2003).

Paillier system is an *additive homomorphic cryptosystem*, that is, the addition of two plain text messages, m_1 and m_2, can be obtained as the decryption of the multiplication of the corresponding encrypted messages, $\mathcal{E}(m_1)$ and $\mathcal{E}(m_2)$. In other words, it is verified that

$$\mathcal{D}\big(\mathcal{E}(m_1) \times \mathcal{E}(m_2)\big) = m_1 + m_2.$$

This homomorphic property is paramount in PAgIoT, as it allows intermediate nodes to aggregate data without having access to the actual information.

The following subsections describe the encryption/decryption procedures and the digital signature scheme defined by Paillier cryptosystem.

2.1. Encryption/decryption procedures

The encryption and decryption operations involve the following steps:

- *Key generation*: Let k be a security parameter. Choose k-bit prime numbers p and q, compute $n = p \cdot q$, and determine Carmichael's function $\lambda(n) = \mathrm{lcm}(p-1, q-1)$. Moreover, let $B_\alpha \subset \mathbb{Z}^*_{n^2}$ be the set of elements of order $n \cdot \alpha$, $B \subset \mathbb{Z}^*_{n^2}$ the disjoint union of the sets B_α for $\alpha = 1, \ldots \lambda$, and generate $g \in B$, at random. This can be done efficiently by checking whether $\mathrm{gc}\big(\mathrm{d}\, L(g^\lambda \pmod{n^2}), n\big) = 1$, where the function $L(\cdot)$ is defined as $L(u) = \frac{u-1}{n}$ (Paillier, 1999).
  Then the public key is the pair (n, g), whereas the private key is the pair $p, q)$, or equivalently λ.
- *Encryption*, $\mathcal{E}(m)$: To encrypt a message $m < n$, choose at random $w < n$, and determine the encrypted message c as follows:
  $$\mathcal{E}(m) = c = g^m \cdot w^n \pmod{n^2}$$
- *Decryption*, $\mathcal{D}(c)$: To decrypt c, verifying that $c < n^2$, the plain text message is computed as follows:
  $$\mathcal{D}(c) = \frac{L(c^\lambda \pmod{n^2})}{L(g^\lambda \pmod{n^2})} \pmod{n} = m.$$

2.2. Digital signature scheme

The digital signature scheme defined by Paillier system has two steps: generation of the signature and its corresponding verification. Note that key generation is done as described for encryption/decryption.

- *Signature generation*, $\mathcal{S}(m)$: Let $h: \mathbb{N} \to \{0, 1\}^* \subset \mathbb{Z}^*_{n^2}$ be a hash function seen as a random oracle. The signature of the message $m < n$ is composed of two parts, $\mathcal{S}(m) = \{s_1, s_2\}$, where
  $$s_1 = \frac{L\big(h(m)^\lambda \pmod{n^2}\big)}{L\big(g^\lambda \pmod{n^2}\big)} \pmod{n},$$
  $$s_2 = \big(h(m) g^{-s_1}\big)^{\frac{1}{n} \pmod{\lambda}} \pmod{n}.$$
- *Signature verification*, $\mathcal{V}(m)$: To verify the signature, (s_1, s_2), of the message m, the following formula has to be checked:
  $$h(m) \stackrel{?}{=} g^{s_1} \cdot s_2^n \pmod{n^2}.$$

3. Model

This section introduces the underlying model of the proposed protocol. Particularly, Section 3.1 describes the participant entities, whereas Sections 3.2 and 3.3 introduce the trust and adversarial models, respectively. Section 3.4 introduces the working assumptions and Section 3.5 presents the goals of PAgIoT.

3.1. Entities

PAgIoT is an aggregation protocol that preserves authenticity of the data at stake, as well as the privacy of the participants. A central node S, referred to as sink, is responsible for gathering and analyzing all the aggregated data. PAgIoT is performed on the assumption that nodes are grouped into clusters. In each cluster, we identify two type of nodes: Master nodes, denoted as M^j, which act as cluster heads and aggregate data from General nodes, $G_i^j \in M^j$, for each j, which are the actual data sources. The information managed in PAgIoT is related to one or more of G_i^j attributes (e.g. height, heart rate). Next, we explain these types of entities and give a high-level overview of their functions within the protocol:

- *Sink node*, S: It gathers all data in order to perform a posterior analysis. Thus, S broadcasts queries and receives responses from the G_i^j nodes in the network. As a result, S gets an aggregated vision of the values of G_i^j attributes. Furthermore, S is able to correlate aggregated values. As an example, consider attributes *height* and *heart rate*. By querying on these attributes (we detail the querying process in Section 4.3) S may know that there are 47 people being [150,160] cm tall (aggregation), and also that 23 of them have a heart rate among [51,90] beats per minute (correlation). S has one public/private key pair, $\{K_S, k_S\}$, for both encryption and signature purposes.
- *Master nodes*, M^j: We assume that the network where PAgIoT operates is divided into several clusters. In each cluster j, there is one master node M^j that receives queries Q and forwards them to the remainder nodes of the cluster G_i^j. Then, it receives responses from these nodes, aggregates them and sends the result to S.
- *General nodes*, G_i^j: These are the actual sources of information. They receive the queries from their cluster head, M^j. After processing the query, they generate the corresponding response and send it to their master node M^j. They are assumed to know S public key K_S.

3.2. Trust model

We consider that S is a trusted party, whereas both G_i^j and M^j are partially trusted. Particularly G_i^j are honest-but-curious, which means that they can eavesdrop others data, but they cannot lie nor refuse to give information to S when asked. Regarding M^j, besides eavesdropping, we assume that they may pollute the results by omitting responses, creating new ones or modifying their content.

3.3. Adversarial model

The proposed adversarial model considers the following attacks:

- *Eavesdropping*: Any node may intercept communications trying to know attributes of another.
- *Collusion attack*: M^j nodes may collude to discover attributes of G_i^j nodes.
  It must be noted that the collusion of S and M^j is out of scope since it would unavoidably lead to a privacy violation – it could be possible for S to get *each* response of *every* node G_i^j.
- *Pollution attack*: M^j may create artificial responses to deceive S. Likewise, M^j may alter queries to request G_i^j chosen attributes and may alter the amount of received responses, deleting some of them.

3.4. Working assumptions

Following we enumerate the assumptions of the working scenario of PAgIoT:

1. *Resilient communications*: We assume that transmission control mechanisms (such as Chouikhi et al., 2015; Kusy et al., 2014) are implemented by all participants. Thanks to these techniques, the communication channel becomes resilient, meaning that data between nodes (i.e. between G_i^j and M^j or between M^j and S) is never lost. In practical terms, it means that the channel is available and that transmission control mechanisms are implemented by all participants.
2. *Node resources*: We assume that S has enough computational resources to decrypt and process the received responses. Regarding M^j and G_i^j, we consider that they can be resource-constrained – they can be implemented in modern smart devices (such as smartwatches or smartphones). This way, it allows the protocol to be run in open and wireless scenarios. Moreover, G_i^j nodes are assumed to have a pseudo-random number generator.
3. *Clustered network*: We assume that nodes are divided into clusters before the execution of PAgIoT. This could be the case of static scenarios or dynamic scenarios where a clustering algorithm is previously run. While research on clustering algorithms for dynamic and mobile scenarios is large, the insights of such a clustering process are out of scope of this paper.

3.5. Goals

The main goal of PAgIoT is to provide a secure aggregation protocol for IoT scenarios, where a large amount of nodes want to share information with a central sink. Specifically, the goals of the protocol PAgIoT are:

- *Privacy preservation*: G_i^j nodes deliver requested data preserving their anonymity. Confidentiality has to be guaranteed as well, such that S is the only node which accesses to all delivered data after being aggregated.
- *Collusion resistance*: Nodes M^j and G_i^j may collude but data from other nodes must remain inaccessible.
- *Verifiable aggregation*: Each M^j aggregates data and S verifies the proper aggregation. S checks that aggregated data belongs to nodes G_i^j involved in the protocol, thus it simultaneously tests the existence of malicious M^j which corrupt aggregated data.
- *Correlatable aggregation*: S must be able to get not only the aggregated results for several attributes, but also to know how attribute values correlate each other.

4. Protocol description

The main idea of PAgIoT is that it allows for aggregation of pieces of information (attributes of a set of entities), enabling the correlation between attribute values. A key feature is that no node is able to reveal information of the aggregated data from any other node. Indeed, even the sink S, which recovers aggregated data and may correlate the attribute values, cannot link this information to particular nodes, thus preserving their privacy. The notation in use throughout the paper is shown in Table 1.

The protocol overview is presented in Section 4.1. Afterwards, all protocol phases are described. PAgIoT consists of five phases depicted in Fig. 1: the system setup (Section 4.2), query broadcasting (Section 4.3), response creation (Section 4.4), response processing and final computation (Section 4.5). Moreover, the algorithm to detect malicious alterations of responses by M^j is presented in Section 4.6.

4.1. Protocol overview

PAgIoT works in a query-response way (Fig. 1). When the sink node S wants to get information from nodes G_i^j, it sends them a signed query $\mathcal{S}_{k_S}(Q)$. As nodes are organized in clusters, this

Table 1
Notation.

Element	Meaning
S	Sink node
M^j	Head of cluster j
G_i^j	Node i belonging to cluster j
K_S / k_S	Paillier public/private key of S
$\mathcal{E}_K(\cdot)$ / $\mathcal{D}_K(\cdot)$	Encryption/decryption procedures using key K
$\mathcal{S}_K(\cdot)$ / $\mathcal{V}_K(\cdot)$	Generation/verification procedures of a signature using key K
Q_i	Query i broadcasted by S
a_i	Attribute i
I_i	Interval of values for attribute a_i
R_i^j	Response of G_i^j node
$\mathcal{E}_K(R_i^j) = \overline{R_i^j}$	Encrypted response of G_i^j node using key K
l	Bitlength required to express the possession of an attribute
$\lvert x \rvert$	Bitlength of the value x
$\lvert A \rvert$	Number of elements of set A

communication is performed through cluster heads M^j which directly forwards the query to their cluster nodes. Each G_i^j constructs its response R_i^j and encrypts it applying Paillier cryptosystem, $\mathcal{E}_{K_S}(R_i^j)$. This response is again sent through M^j, which performs aggregation prior to forwarding back to S. Accordingly, once M^j has gathered all responses to the query from its cluster members, it aggregates all of them leveraging on the homomorphic feature of Paillier and generates the aggregated response $\overline{R^j}$. The result is sent to S, which again aggregates all the responses from received from master nodes. Then, S can decrypt the overall response and get the aggregated vision of the required data.

One important aspect is that M^j could attempt to counterfeit the aggregation. For example, it could try to inject an artificial response to deceive S. To address this issue, each node G_i^j introduces a unique, incremental random number into each response. Based on this number, S can determine if any M^j has polluted its results by applying a *Misbehaving aggregator detection algorithm*, which is further explained in Section 4.6.

4.1.1. Supporting example

Let us consider that S is the emergency manager of a football stadium, which is divided into zones or clusters as in Fig. 2 (without loss of generality, in the example we use 2 clusters for simplicity). S defines three heart rate intervals (e.g. [0,50], [51,90] and [91,200]) and two gender types (male, female) and launches the first query (Q_1) to discover the heart rate and gender of each person (each one carrying a node G_i^j).

Once the aggregated response has been retrieved, S launches a second query which is only devoted to females as the prevalence of heart disease is higher on them.[1] Particularly, this second query (Q_2) consists of discovering the age of women. Three intervals are defined concerning age, namely, [0,30], [31,60] and [61,100].

Based on these settings, the following subsections will show how PAgIoT works to address this need.

4.2. Setup

At the beginning of the protocol, each node G_i^j that takes part in the protocol receives the public key from S, K_S. Besides, each node G_i^j generates an unique random number r_i^j. This number may be much bigger than the amount of G_i^j nodes to minimize collisions. The number is encrypted and sent to S through M^j. Thus, preventing S from linking a given r_i^j to any particular node.

4.3. Query broadcasting

Whenever S desires, it broadcasts queries Q to all cluster heads M^j. The query asks for a set of attribute values from the nodes G_i^j that S wants to know. A query Q is formally defined as:

$$Q = \mathcal{S}_{k_S}(\{A, I, l\}),$$

where the set $A = \{a_1, \ldots, a_n\}$ contains all attributes a_i that are at stake in the current query, the set $I = \{I_1, \ldots, I_n\}$ contains the intervals of values for each attribute, and l is the amount of bits required to express the possession of an attribute.

Recalling the supporting example (Section 4.1.1), the two queries Q_1 and Q_2 are formed as follows:

$$Q_1 = \mathcal{S}_{k_S}(\{\{\text{heart rate, gender}\}, \{\{[0, 50], [51, 90], [91, 200]\}, [\text{male, female}]\}, 3\}),$$

$$Q_2 = \mathcal{S}_{k_S}(\{\{\text{age, gender}\}, \{\{[0, 30], [31, 60], [61, 100]\}, [\text{female}]\}, 3\}),$$

One parameter that deserves attention is the value $l=3$. The rationale behind this is as follows. The aggregation process counts how many nodes have an attribute value that matches each interval. Accordingly, the maximum number of matches for a given interval is the total number of nodes in the network, say $|G|$. Thus, the number of required bits to express the possession can be calculated as shown in the following equation:

$$l = \lceil \log_2(|G|) \rceil. \tag{1}$$

In the example (recall Fig. 2), $l = \lceil \log_2(6) \rceil = 3$.

Last but not least, queries are signed by S to prevent them from malicious manipulations. Particularly, as they are sent through M^j, the signature guarantees that queries are not altered by any M^j.

4.4. Response creation

Each node G_i^j verifies the signature of the received query, and constructs the response, R_i^j, based on the query. In a nutshell, R_i^j contains a set of bits for each combination of values of the queried attributes. For the sake of brevity, each one of these sets is referred to as a *property* P_i. Each node G_i^j puts $P_i=1$ if it meets the property, and $P_i=0$ otherwise.

The total amount of properties, N_p, is calculated as the amount of possible combinations of intervals I_i of the requested attributes a_i, i.e.:

$$N_p = \prod_i |I_i|$$

The node G_i^j inserts into R_i^j its random number, r_i^j, to be used in the misbehaving-aggregator-detection algorithm (Section 4.6). Thus, the final structure of responses R_i^j is as follows, $\|$being the concatenation operator:

$$R_i^j = \{P_1 \| \cdots \| P_{N_p} \| r_i^j\}.$$

The order of the properties within the response is assumed to be agreed among all nodes in a given execution. For the sake of simplicity, a *pre-order* combination of attributes is assumed in this paper. Thus, if attribute a_1 has three values and a_2 has two, property P_1 is formed by the first values of a_1 and a_2, P_2 is composed of the first value of a_1 and the second one of a_2, P_3 comes from the second value of a_1 and the first one of a_2, and so on.

Responses R_i^j are encrypted for S by G_i^j using the Paillier

[1] https://www.heart.org/idc/groups/heart-public/@wcm/@sop/@smd/documents/downloadable/ucm_449846.pdf, last accessed in October, 20, 2015.

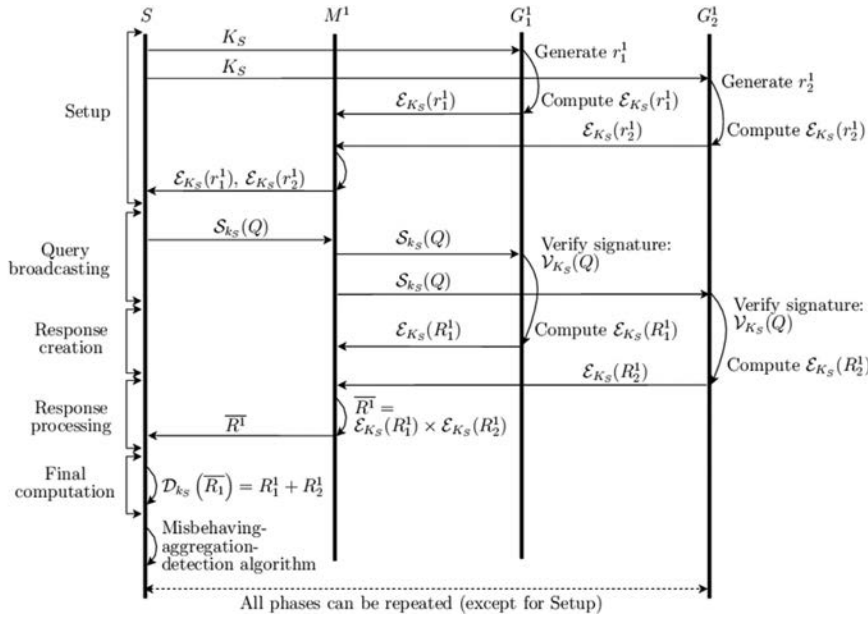

Fig. 1. Protocol phases.

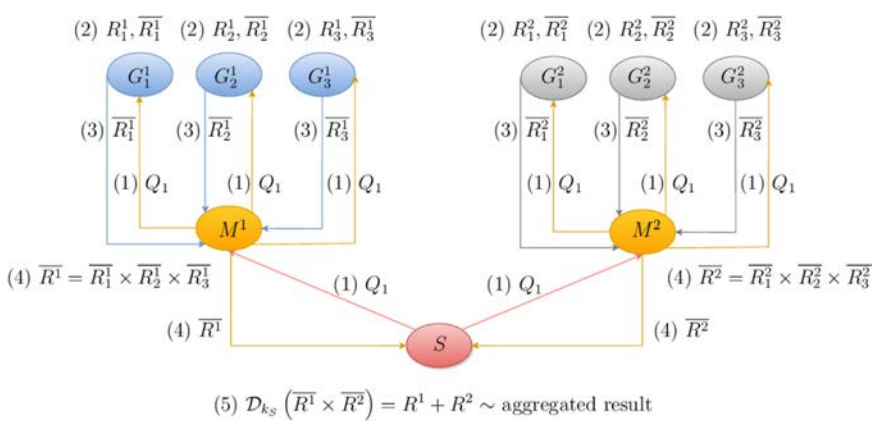

Fig. 2. Execution of PAgIoT in the football stadium example: *S* receives the heart rate and gender of each node.

cryptosystem and the encryption public key of *S*, $\mathcal{E}_{K_S}(R_i^j) = \overline{R_i^j}$. Afterwards, $\overline{R_i^j}$ is sent to M^j.

One critical aspect for performance is the length of R_i^j. This length, l_R, depends on the length of each property P_i and the number of bits of the maximum r_i^j, l_r. The length of this last parameter has to prevent collusion among nodes (studied in Section 4.6). In order to allow aggregation, each property must be long enough to allow *all* nodes having that property. For this reason, each property must be *l* bits long (recall Section 4.3). Eq. (2) shows the formal expression for l_R:

$$l_R = N_p \cdot l + l_r. \tag{2}$$

4.4.1. Response creation example

Following the supporting example (recall Section 4.1.1) and regarding Q_1, six different properties (i.e. combinations of attri-butes) are noticed: to be female with [0,50], [51,90] or [91,200] beats per minute of heart rate or to be male with [0,50], [51,90] or [91,200] beats per minutes of heart rate. Accordingly:

$P_1 = 1$ if {[0, 50], female},
$P_2 = 1$ if {[51, 90], female},
$P_3 = 1$ if {[91, 200], female},
$P_4 = 1$ if {[0, 50], male},
$P_5 = 1$ if {[51, 90], male},
$P_6 = 1$ if {[91, 200], male}.

Assuming a node G_1^1 whose owner is female and whose heart rate is 85, it would belong to P_2 from the above. Considering $l_r = 7$, the generated random number is represented as $r_1^1 = 23_{10} = 0010111_2$. Last but not least, recall that l=3. With all these elements, the response remains as follows:

$$R_1^1 = \{000\|001\|000\|000\|000\|000\|0010111\}.$$

4.4.2. Preventing eavesdroppers

Depending on the queried values, a given node G_i^j may or may not reply to them. Again applying the supporting example but concerning Q_2, node G_2^2 could be male and then, it is not forced to send an answer. However, in such a case the remaining cluster nodes (G_1^2, G_3^2 and M^2, see Fig. 2) would learn its gender. Thus, defending against eavesdroppers is critical.

To address this issue, it is important to note that G_i^j nodes are honest (recall Section 3). Thus, they cannot send false responses. However, G_i^j nodes can build an empty response (i.e. all properties P_i set to zero, using the right value of r_i^j) to defeat eavesdroppers. This decision (i.e. giving an empty response or avoiding to reply) may be based on internal G_i^j parameters, such as reducing battery consumption.

4.5. Response processing

Intermediate nodes M^j receive $\overline{R_i^j}$ from their cluster members, G_i^j. These intermediate nodes aggregate the information by computing the multiplication of all encrypted responses, thus building their response $\overline{R^j}$, that is,

$$\overline{R^j} = \prod_i \overline{R_i^j}$$

This is the main advantage of using the homomorphic features of Paillier cryptosystem – the multiplication of $\overline{R_i^j}$ leads to the addition of R_i^j (after decryption). Given the encoding proposed in the responses, each different property, P_i, will be added (i.e. aggregated) with others, but the node M^j cannot reveal information as data is encrypted.

Afterwards each M^j concatenates $\overline{R^j}$ with a number N_r, that expresses the amount of multiplied $\overline{R_i^j}$. The use of N_r is associated with the misbehaving-aggregator-detection algorithm described in Section 4.6. Once $\overline{R^j}$ is created, it is sent to S. Later, S multiplies all received $\overline{R^j}$ to have all data aggregated, and then decrypts it using k_S.

4.5.1. Response processing example

Regarding Q_1 of the supporting example and detailed in the upper part of Fig. 3, it is supposed that M^1 and M^2 have received information from all G_i^j nodes, $1 \le j \le 2$ and $1 \le i \le 3$, in their respective clusters each of one having the following properties:

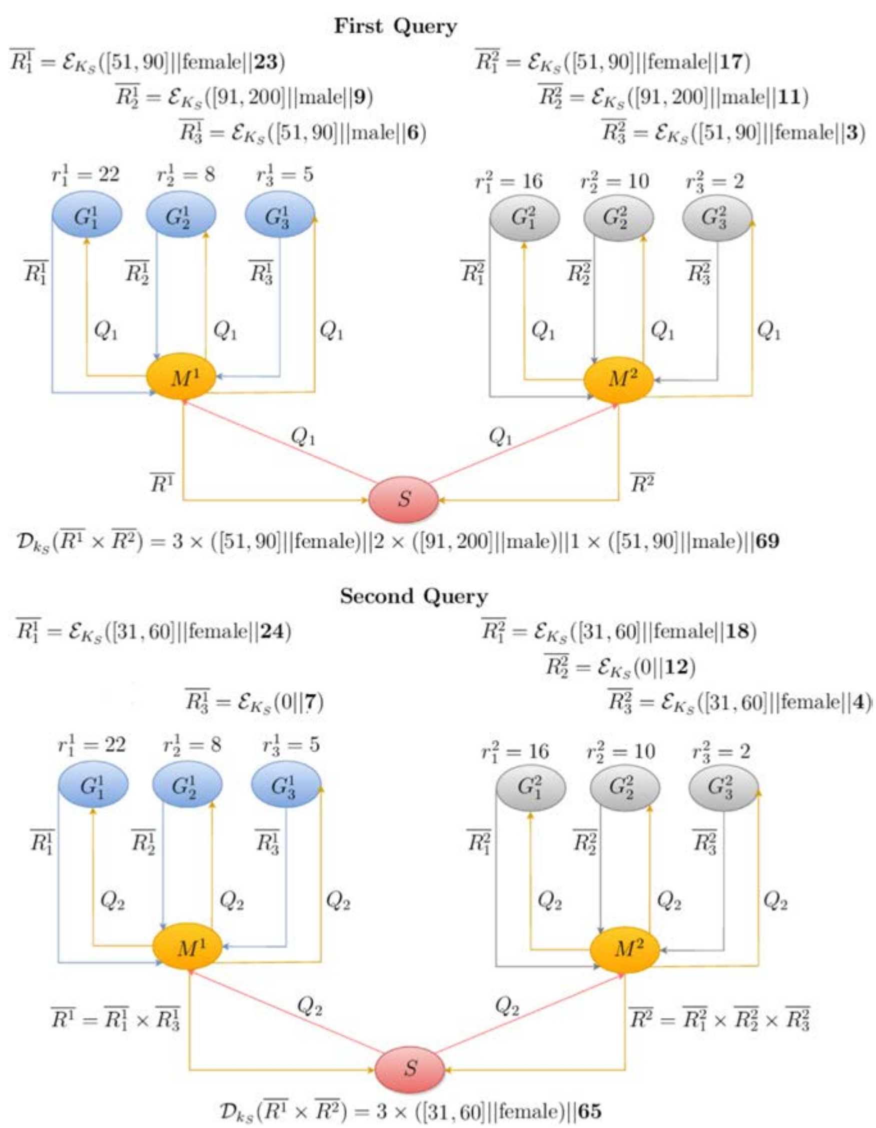

Fig. 3. Misbehaving-aggregator-detection algorithm applied over the football stadium example.

Table 2
Sample aggregation of decrypted responses R^1 and R^2 by S.

M^1	$R_1^1 = \{000\|\|001\|\|000\|\|000\|\|000\|\|000\|\|0010111\}$
	$R_2^1 = \{000\|\|000\|\|000\|\|000\|\|000\|\|001\|\|0001001\}$
	$R_3^1 = \{000\|\|000\|\|000\|\|000\|\|001\|\|000\|\|0000110\}$
M^2	$R_1^2 = \{000\|\|001\|\|000\|\|000\|\|000\|\|000\|\|0010001\}$
	$R_2^2 = \{000\|\|000\|\|000\|\|000\|\|000\|\|001\|\|0001011\}$
	$R_3^2 = \{000\|\|001\|\|000\|\|000\|\|000\|\|000\|\|0000011\}$
S	$R^1 = \{000\|\|001\|\|000\|\|000\|\|001\|\|001\|\|0100110\}$
	$R^2 = \{000\|\|010\|\|000\|\|000\|\|000\|\|001\|\|0011111\}$

$$G_1^1 \sim \{[51, 90], \text{female}\} \approx P_2, \quad \text{with } r_1^1 = 23_{10},$$
$$G_2^1 \sim \{[91, 200], \text{male}\} \approx P_6, \quad \text{with } r_2^1 = 9_{10},$$
$$G_3^1 \sim \{[51, 90], \text{male}\} \approx P_5, \quad \text{with } r_3^1 = 6_{10},$$
$$G_1^2 \sim \{[51, 90], \text{female}\} \approx P_2, \quad \text{with } r_1^2 = 17_{10},$$
$$G_2^2 \sim \{[91, 200], \text{male}\} \approx P_6, \quad \text{with } r_2^2 = 11_{10},$$
$$G_3^2 \sim \{[51, 90], \text{female}\} \approx P_2, \quad \text{with } r_3^2 = 3_{10}.$$

When S decrypts $\overline{R}^1$ and $\overline{R}^2$, the aggregation of all responses of each cluster would be as shown in Table 2. It can be concluded that there are three nodes female with heart rate between [51,90] (one of them belonging to the cluster 1 and the other two to the cluster 2), two nodes male with a heart rate between [91,200] (one of each node) and a node male with a heart rate between [51,90] (from the cluster 1).

Finally, S has to verify if any M^j has altered or dropped responses. To do so, the algorithm presented in the following section is developed.

4.6. Misbehaving aggregator detection algorithm

This algorithm helps to detect malicious M^j which lie about the amount of received responses. It leverages on the random number r_i^j introduced in each response. The algorithm works as follows:

1. Each G_i^j creates a random number r_i^j and sends it, encrypted, to M^j, thus avoiding S linking r_i^j with G_i^j. Each of such values should be $r_i^j < L_r + N_Q$, where L_r is an upper limit of r_i^j and N_Q an upper limit of the expected number of queries. Accordingly, the number of bits to define r_i^j, denoted as l_r, is calculated from the number of nodes G_i^j involved in the protocol, $|G|$, and $(L_r + N_Q)$, see the following equation:
$$l_r = \lceil \log_2(L_r + N_Q) \rceil \cdot \lceil \log_2(|G|) \rceil. \tag{3}$$
2. Response creation and Response processing (recall Sections 4.4 and 4.5) are executed.
3. Once S decrypts multiplied responses, it verifies that the last concatenated value, N_r, is the sum of received $r_i^j + 1$ in the setup phase of all received responses. The amount of operations to perform is equivalent to the calculus of combinations of G_i^j nodes involved in the protocol, $|G|$, choosing N_r of them, see the following equation:
$$\binom{|G|}{N_r} = \frac{|G|!}{N_r!\,(|G| - N_r)!} \tag{4}$$
Then, S just has to calculate the sum of all $r_i^j + 1$ because all nodes have answered. Thus, S checks if
$$\sum_{i,j} \left(r_i^j + 1 \right)$$
is equivalent to the last part of the concatenated values of
$$\mathcal{D}\left(\prod_j \overline{R^j} \right) = \mathcal{D}\left(\prod_j \mathcal{E}\left(R^j \right) \right) = \sum_j R^j. \tag{5}$$
When this process finishes, S updates values of all r_i^j to be ready for the following query.
According to Q_1 of the supporting example (upper part of Fig. 3), S stores the values of all r_i^j: $r_1^1 = 22$, $r_2^1 = 8$, $r_3^1 = 5$, $r_1^2 = 16$, $r_2^2 = 10$, and $r_3^2 = 2$. Later, S computes the sum of all such values, obtaining
$$\sum_{1 \le i \le 3, 1 \le j \le 2} \left(r_i^j + 1 \right) = 23 + 9 + 6 + 17 + 11 + 3 = 69,$$
which coincides with the last part of the decrypted value obtained by S (see Eq. (5)). Finally, S updates the values of r_i^j, increasing each of them in 1: $r_1^1 = 23$, $r_2^1 = 9$, $r_3^1 = 6$, $r_1^2 = 17$, $r_2^2 = 11$, and $r_3^2 = 3$.
4. Assuming that S sends a second query, the process is repeated with the exception that each G_i^j increments r_i^j by one. It must be noted that *all* nodes update their value of r_i^j, no matter if they answer to a query or not.
Again on the bases of the supporting example (lower part of Fig. 3), assuming that G_1^1, G_1^2 and G_3^2 are women and G_1^3 and G_2^2 answer to preserve their anonymity, S performs
$$\binom{6}{5} = \frac{6!}{5!(6-5)!} = 6$$
operations to verify the correct reception of responses. In particular, it calculates all possible combinations and if
$$r_1^1 + 1 + r_3^1 + 1 + r_1^2 + 1 + r_2^2 + 1 + r_3^2 + 1 = 24 + 7 + 18 + 12 + 4 = 65$$
is reached, it means that all M^j have behaved properly. Finally, S updates r_i^j.

It is noteworthy that if a set of x additions get the same value, the verification of the next query would multiply by a factor of x the amount of operations. However, operations do not exponentially increase with the number of queries because after each query, S knows values of r_i^j applied for the following verification. Moreover, note that r_i^j are random and kept secret (through the use of encryption), to avoid malicious nodes from the inference or injection of well-formed responses.

5. Evaluation

This section presents a theoretical and an empirical evaluation. Firstly, the assessment of established goals is studied (Section 5.1). Secondly, privacy preservation and, particularly, the data that may be inadvertently guessed by M^j is analyzed (Section 5.2). Thirdly, a performance analysis is presented (Section 5.3). Lastly, PAgIoT is compared against the two works that provide with the same set of security properties, as it is discussed in Section 6. This analytical comparison is shown (Section 5.4).

5.1. Goals assessment

- *Privacy preservation*: Data sent from every G_i^j is encrypted with

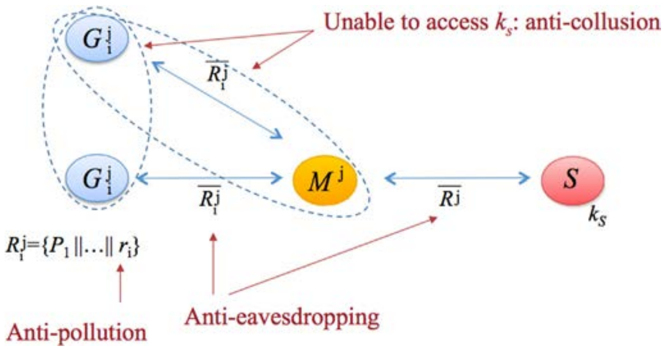

Fig. 4. Prevented attacks.

the public key K_S, which belongs to S. It ensures confidentiality and avoids malicious eavesdroppers to access the private data. Besides, since S only receives aggregated data, individual anonymity of nodes G_i^j is preserved. In particular, S is unable to explicitly identify G_i^j by analyzing received attributes. In Section 5.2 we conduct a further analysis of the capabilities of M^j nodes to infer data from the received responses within their clusters.

- *Collusion resistance*: Each node G_i^j encrypts data and sends it to M^j. Neither M^j nor G_i^j know the decryption key and then even colluding, no data of other nodes is accessible. A force brute attack can be also considered. However, as Paillier cryptosystem produces different cyphertext from the same clear text, the attack would be impractical. Due to these considerations the collusion among multiple M^j or an M^j and several G_i^j do not compromise the system.
- *Verifiable aggregation*: The inclusion of incremental random numbers r_i^j in each response (applied in the misbehaving-aggregator-detection algorithm) helps to verify the correctness of the aggregation guaranteeing that M^j properly aggregate all received responses.
- *Correlatable aggregation*: The design of responses, based on combining attribute values, allows S to correlate such values without additional computation.

Summarized in Fig. 4, encryption prevents from eavesdropping as data is inaccessible. Likewise, encryption together with the fact that S is the only one which owns the decryption key, avoids collusion attacks (recall that or proposal is based on asymmetric cryptography). Conversely, the introduction of random numbers within responses allows the identification of malicious behaviors. S can detect, based on the received sum of random numbers, if the aggregation is or not polluted.

5.2. Characterizing the privacy preservation

One of the key aspects of PAgIoT is that it prevents S from linking the answers to particular G_i^j. This statement is also true for M^j, since answers are encrypted for S. There is, however, a room for M^j to gain information about a given G_i^j. This section focuses on how this undesired result may happen.

In a nutshell, information gain by M^j depends on two main factors: the set of queries and the context knowledge of M^j. To illustrate this issue, the supporting example is considered (recall Section 4.1). As the first query Q_1 is general, all nodes will give an answer. However, query Q_2 is only focused on women. As a result, a given male node (say G_2^1) is not forced to give an answer.

Considering this issue, the information gain that M^1 achieves from G_2^1 is clear: his gender. The capacity of giving a void response for these cases is critical for privacy preservation. In such a case, thanks to the use of encryption, M^1 cannot discover that G_2^1 is giving a void response and thus, gender leakage is prevented.

One obvious consequence of this is that G_2^1 must keep a coherent participation pattern. Let us consider that S sends a third query, Q_3 that asks all women for their heart rate, using all possible intervals. G_2^1 must also give a void answer because otherwise M^1 would discover that the answer to Q_2 was useless and that, in the end, G_2^1 is a male.

Apart from this issue, context knowledge of M^j also plays an important role. If the example was done in a football-related supermarket (say an official store for a given football club), M^j could predict in advance the most probable profile of any G_i^j. Particularly, the *expected* gender of G_i^j would be male (say 75% of them). Therefore, if Q_2 is answered by 40%, a given M^j would know that a portion of answers (an estimated amount of 15% of them) are void. Although it could not lead to a precise identification, this could be combined with the remaining queries.

Considering the previous facts, privacy preservation in PAgIoT is achieved as long as S carefully designs its queries in such a way that they do not lead to a data leakage, either individually or by combination. Furthermore, how G_i^j determines when to participate in unforced queries may also contribute to this issue.

5.3. Performance analysis

The performance analysis focuses on the study of the computation cost to test the feasibility of developing PAgIoT, as well as the size of responses to analyze channel bandwidth.

5.3.1. Computation time

In order to measure consumed resources by the protocol in a real setting, the functionality of both G_i^j nodes and S has been implemented. M^j nodes are less challenging, since they only forward queries and responses, and perform linear additions of the received encrypted responses. Thus, they are assumed to have enough resources to cope with the amount of nodes in their cluster.

G_i^j is emulated by an Android application running in smartphones with constrained resources (i.e., battery and power). Regarding S, it is implemented as an Ubuntu server. We conduct the experiments using a Samsung Galaxy S3 (processor ARMv7 and 832 MB of RAM) and an Intel Core 2 Processor (3.00 GHz) with 4 GB of RAM.

In the experiments the size of the Paillier security parameter k is 256 (i.e., $|n| = 512$). We have experimented with constant values

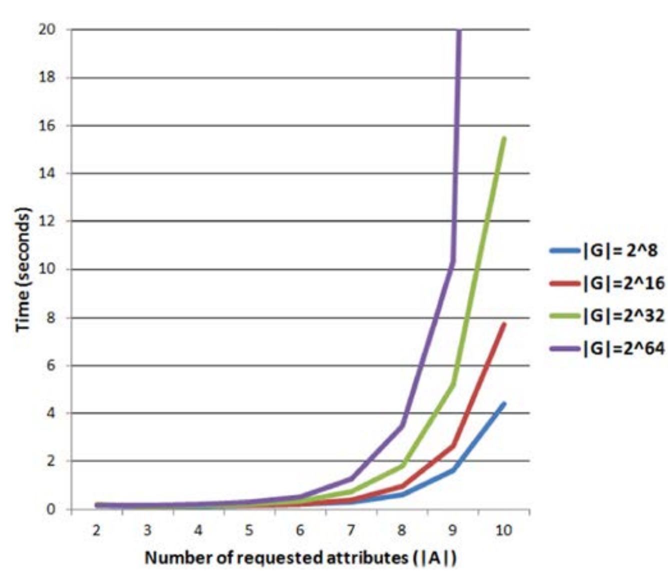

Fig. 5. Response encryption time measured in a smartphone.

for requested attributes (note that in a real scenario, these values will be either stored in the node or requested to wearable or other devices from the user). Moreover, different network sizes are tested, namely $|G| \in \{2^8, 2^{16}, 2^{32}, 2^{64}\}$, as well as $|A|$ from 2 to 10 and $|I_i| = 3$ for all attributes. Under these assumptions the computation time of each protocol phase and operation have been measured. Nonetheless, findings show that the encryption of responses is by far the most costly operation and it is the only one which is worth studying. The computation cost of remaining operations can be considered negligible in the proposed scenario.

Under the previous setting, Fig. 5 shows encryption time. It is noticed that requesting up to 6 attributes can be performed in less than half of a second with any size of the network. Indeed, in a network with, e.g., 2^{16} nodes which can be considered a large scale network, encrypting a response when 7 attributes are requested still requires less than a second of computation. Indeed, in a worst case scenario when 9 attributes are requested, for $|G| \in \{2^8, 2^{16}, 2^{32}, 2^{64}\}$, the encryption time is 1.6; 2.6; 5.2 and 10.3 s., respectively. These results show that, even with such amount of requested attributes, the protocol is particularly feasible for networks with 2^8 and 2^{16} nodes.

The experimental results show that the protocol PAgIoT is suitable to be implemented in a large scale network composed by one central server, running in a non-constrained system (like a high performance computing server) and a large number of nodes running an application which consumes few resources.

5.3.2. *Responses size analysis*

This analysis focuses on studying the size of the responses, l_R, which is a critical parameter for the overall performance of the network and particularly related to the channel bandwidth. The aim of this analysis is to determine the most appropriate values for the size of the network, $|G|$, the size of intervals of each attribute, $|I_i|$, and the amount of requested attributes, $|A|$, in a query. Besides, three thresholds are considered, namely, the maximum amount of bytes that can be sent in a long text message, TH1, that is 480 bytes (Sheets et al., 2012); the maximum size of a TCP packet, TH2, that is 64 kbytes; and the average size of a WhatsApp message, TH3, that is 225 kbytes (Fiadino et al., 2015). Note that TH1 and TH2 are chosen for being units of information transmission in networks and TH3 for illustrative purposes as WhatsApp is a widespread application.

In the proposed experiment different values of network sizes and attributes are applied, namely $|G| \in \{2^8, 2^{16}, 2^{32}, 2^{64}\}$ and $|A_i| \in \{2, 6, 10\}$. For simplicity, it is assumed that all requested attributes have the same interval size, that is, $|I_i| \in \{3, 5, 10\}$. The size of the random number l_r is established in regard to $L_r = 10^6$ and the number of expected queries N_Q is set to 5000. Note that l_r, identified from Eq. (3), does not significantly affect l_R and then, a high value for l_r is chosen as a representative example. Results are depicted in Table 3 where values under **TH1**, *TH2* and <u>TH3</u> are bold, italic and underline style, respectively.

The worst results (and unacceptable under the established thresholds) occur when $|I_i| = 10$ and $|A| > 2$. However, challenging results are achieved in the remaining cases. Under TH1 for $|A| = 2$ and $|I_i| = \{3, 5\}$, all l_R are accepted except for $|G| = \{2^{32}, 2^{64}\}$ when $|A| = 6$. In the case of TH2, the only rejected results are those in which $|A| = 10$ and $|I_i| = \{5, 10\}$. Similarly, for TH3 all values are accepted except for those in which $|A| = 10$, $|I_i| = 5$ and $|G| = \{2^{32}, 2^{64}\}$ and in which $|A| = 10$ and $|I_i| = 10$. Moreover, $|G|$ also affects l_R but not as much as the remaining elements. Indeed, $|G| = \{2^{32}, 2^{64}\}$ are the only values which affect l_R negatively. For instance, for $|I_i| = 5$ and $|A| = 6$, l_R=124,735 when $|G| = 2^{16}$ and l_R=249,470 when $|G| = 2^{32}$. It means that l_R increments 50% when $|G|$ increases 2^{16}%.

In sum, l_R remains between established thresholds once choosing many $|A|$ (about 10) with small amount of $|I_i|$ (about 5) or few $|A|$ (about 2) with high amount of $|I_i|$ (about 10). On the contrary, the total number of nodes in the network, $|G|$, is not a distinguishing factor because l_R is much lesser affected by this element than by the number of requested attributes, $|A|$, or the requested intervals for each attribute, $|I_i|$.

The above results show that PAgIoT is suitable for large scale networks, composed by a great number of nodes. However, a pair of issues should be considered concerning attributes and intervals. On the one hand, these elements have to be chosen in regard to each particular scenario to prevent from performance problems. On the other hand, attributes and intervals particularly affect the channel bandwidth. It is worth recalling that due to the query/response approach used in PAgIoT, as well as the use of clustering, the amount of traffic is limited and distributed. Each query must arrive to every G_i^j node through M^j nodes, and at least $|G|$ packets are transmitted. However, nodes can decide to respond to a query, and thus the response process generate less than $|G|$ packets or $|G|$ packets at most. While this may be object of further research when implementing PAgIoT in a real scenario, current mobile technologies such as 802.11 or 3G, can deal with such large scale settings provided that enough access points are placed within the scenario (Riiser et al., 2013; Chen et al., 2012).

Table 3
l_R for different values of $|G|$, $|I_i|$, and $|A|$.

$\|A_i\|$	$\|G\| = 2^8$	$\|G\| = 2^{16}$	$\|G\| = 2^{32}$	$\|G\| = 2^{64}$
$\|I_i\| = 3$				
$\|A_i\| = 2$	**223.5**	**447.0**	**894.0**	**1788.1**
$\|A_i\| = 6$	**1887.5**	**3775.0**	*7550.0*	*15,100.1*
$\|A_i\| = 10$	*8159.5*	*16,000.0*	*32,000.0*	*65,276.1*
$\|I_i\| = 5$				
$\|A_i\| = 2$	**415.51**	**831.02**	**1662.04**	**3324.08**
$\|A_i\| = 6$	*62,367.51*	*124,735.02*	*249,470.04*	*498,940.08*
$\|A_i\| = 10$	<u>800,159.51</u>	<u>1,600,000.00</u>	3,200,000.00	6,401,276.08
$\|I_i\| = 10$				
$\|A_i\| = 2$	*8351.51*	*16,703.02*	*33,406.04*	*66,812.08*
$\|A_i\| = 6$	483,729,567.51	967,459,135.02	1,934,918,270.04	3869,836,540.08
$\|A_i\| = 10$	80,000,000,159.51	160,000,000,000.00	320,000,000,000.00	640,000,001,276.08
TH1: 3840				
TH2: 524,280				
<u>TH3: 1,800,000</u>				

Table 4
Comparison HDA (Kim et al., 2013), iCPDA (He et al., 2009) and PAgIoT , where n is the amount of nodes.

Proposals	Number of keys	Number of interchanged messages		
		Setup	Aggregation	Aggregation verification
HDA	$n + 1$	$2n + 2$	$n + 1$	$2n + 2$
iCPDA	$\frac{n(n+1)}{2}$	$n^2 + 2$	$5n + 5$	1
PagIoT	1 *pair*	$2n + 1$	$2n + 2$	0

5.4. Analytical comparison

An analytical comparison between PAgIoT, HDA (Kim et al., 2013) and iCPDA (He et al., 2009) is performed herein. The last two proposals have been chosen for being the ones whose security requirements are similar to those set for PAgIoT (see Section 6). For the sake of clarity, we refer the reader to Section 6 to get a brief description of these works.

The comparison is performed based on two aspects that are relevant for measuring the performance of the proposal, namely the amount of keys at stake and the amount of transmitted messages in the setup, aggregation and aggregation verification phases within a single cluster. Thus, it can be easily extrapolated to any number of clusters. To perform a fair comparison it is assumed that clusters are already created, that each cluster is composed of n nodes G_i^j and that aggregated data is sent from the head of the cluster M^j to the sink S.

Table 4 presents results of the comparison. The number of keys is particularly noticeable in HDA and iCPDA. In the former proposal each G_i^j shares a key with M^j and this latter with S. In iCPDA a protocol is enforced in which each G_i^j shares a key with all other nodes in the cluster as well as with M^j. Conversely, in PAgIoT a public key cryptosystem is applied in which just a keypair (public/private) is used for encryption and aggregation purposes.

Concerning sent messages, we analyze each phase separately. In the Setup phase results are quite similar in PAgIoT and HDA. In HDA, nodes, including M^j with S, create the key to apply in the remaining parts of the protocol. The process is compared to a Diffie–Hellman protocol but in the context of elliptic curves. In PAgIoT a message composed of the sink public key K_S is sent to each G_i^j. Subsequently, each G_i^j creates a random number and sends it encrypted to M^j which redirects them to S. A more costly process is performed in iCPDA. A key establishment protocol is executed such that each G_i^j performs a broadcast communication with all the remaining G and lastly M^j shares keys with S.

Significant differences are identified comparing the Aggregation phase. HDA requires computing Hilbert curve points and a sum of them to be sent the result to M^j and redirected to S subsequently. In iCPDA the aggregation can be divided in three parts: firstly each G_i^j broadcasts a random number to all G in the cluster ($2(n + 1)$ messages); secondly each G_i^j computes a polynomial per other G and sends results to each of them ($2(n + 1)$ messages); and finally, each G_i^j performs a final addition of polynomials and sends the result to M^j which redirects it to S ($n + 1$ messages). By contrast, in PAgIoT S sends a query and each G_i^j responses accordingly to M^j and finally the result is received by S.

The amount of messages interchanged for aggregation verification is specially significant in HDA. A Privacy Information Retrieval (PIR) technique is applied. Each G_i^j, also M^j from S, receives an integrity request message and G response accordingly. The protocol proposed in iCPDA consists of sending a message from M^j to other M disclosing received information to detect malicious behaviors. A different approach proposes PAgIoT, the verification is performed using data within the received aggregated message and thus, no extra messages are needed.

Results drawn from this comparison show that PAgIoT is specially appropriate in terms of key management and aggregation verification. Moreover, the amount of messages in the setup phase is analogous to HDA and outperforms iCPDA. Concerning aggregation, HDA involves less amount of messages than PAgIoT because an initial request query is not required. However, iCPDA is by far the proposal with worse results producing a significant workload in the network. In sum, considering that PAIoT outperforms iCPDA in all cases, it can also be considered a better alternative than HDA because no messages are required for aggregation verification and just one key is managed.

6. Related work

Aggregation has been a challenging topic so far (Rajagopalan and Varshney, 2006; Heidemann et al., 2001). Multiple proposals have been developed, from the aggregation of values (Applebaum et al., 2010; Xiaoying et al., 2014; Buttyán and Holczer, 2010; Li et al., 2014; Horey et al., 2007; Chan et al., 2007; Othman et al., 2015; Ma et al., 2015; Manni et al., 2015; Villas et al., 2013; Conti et al., 2009; Fu et al., 2015), to additions (He et al., 2007; Wenbo et al., 2008; He et al., 2009; Bae et al., 2016; Rieffel et al., 2014; Rastogi and Nath, 2010; Raj et al., 2016; He et al., 2016; Fan et al., 2014; Grining et al., 2016), calculus of minimums or maximums (Madden et al., 2002; Groat et al., 2011; Rodero et al., 2010) or even more complex tasks such as averages (Ren et al., 2013; Madden et al., 2002; Chan et al., 2007; Castelluccia et al., 2005), variances (Ren et al., 2013; Castelluccia et al., 2005), range queries (Xiaoying et al., 2014) or logic operations (Kumar et al., 2015).

Over the years security requirements have been introduced in this process (Cheikhrouhou, 2015). Anonymity is specially remarkable because aggregators should not learn particular data. Some proposals focus on the anonymity of messages passed through aggregator nodes (Xiaoying et al., 2014; Horey et al., 2007; Groat et al., 2011) or on identity preservation of such nodes (Buttyán and Holczer, 2010) or nodes providing information (He et al., 2007; Wenbo et al., 2008; Applebaum et al., 2010; He et al., 2009; Rieffel et al., 2014). Anonymity is generally achieved by the use of cryptographic schemes, e.g. symmetric encryption (He et al., 2007; Xiaoying et al., 2014; Buttyán and Holczer, 2010; Chan et al., 2007) or more sophisticated schemes like homomorphic encryp-tion (Kumar et al., 2015; Bae et al., 2016; Rieffel et al., 2014; Rastogi and Nath, 2010; Othman et al., 2015; Raj et al., 2016; He et al., 2016; Fan et al., 2014; Grining et al., 2016; Castelluccia et al., 2005), homomorphic aggregation of signatures (Fu et al., 2015) and homomorphic encryption with re-randomization applying a proxy (Applebaum et al., 2010). By contrast, Groat et al. (2011) propose the use of randomized arrays instead of a cryptographic system. Each node sends an array of values to the root of a subtree until reaching the root of the tree (i.e. the overall aggregator). Values are in plain text but the position of the real values in the array is only known by the node which is root of the tree. Likewise, Horey et al.(2007) do not use cryptography but negative surveys. Negative values are transmitted from nodes providing information to the overall aggregator, which reconstructs the final histogram of data. Much simpler aggregation techniques are proposed by Ma et al.(2015), Manni et al. (2015), and Villas et al. (2013).They just mention the aggregation of plain text data by a central entity.

Privacy preservation against observers is considered in He et al. (2007), Wenbo et al. (2008), Applebaum et al. (2010), Buttyán and Holczer (2010), Kumar et al. (2015), Horey et al. (2007), He et al. (2009), Groat et al. (2011), Othman et al. (2015), He et al. (2016), Fan et al. (2014), Castelluccia et al. (2005), Fu et al. (2015), Kim

Table 5
Related work analysis.

Proposals	Security properties				Correlatable aggregation	Aggregation purpose	Encryption scheme
	Anonymity of sending nodes	Privacy preservation against observers	Collusion resistance	Verifiable aggregation			
Groat et al. (2011)	✓	√	✓	✗	✗	Min, max	✗
He et al. (2007)	√	√	√	✗	✗	Sum	Symmetric
Bae et al. (2016)	√	√	✗	✗	✗	Sum	Homomorphic
Li et al. (2014)	✗	✗	✗	√	✗	Sum	Asymmetric
Wenbo et al. (2008)	√	√	✗	✗	✗	Sum	not defined
Madden et al. (2002)	✗	✗	✗	✗	✗	Count, min, max, sum, average	✗
Rodero et al. (2010)	✗	✗	✗	✗	✗	Count, min, max, sum	✗
Ren et al. (2013)	✗	✗	✗	✗	✗	Count, min, max, sum, average, stdev, variance	✗
Applebaum et al. (2010)	√	√	√	✗	✗	Sum	Asymmetric
Xiaoying et al. (2014)	✗	✗	✗	√	✗	Count	Symmetric
Buttyán and Holczer (2010)	✗	√	✗	✗	✗	Generic aggregation	Symmetric
Horey et al. (2007)	✗	√	✗	✗	✗	Count	✗
Rieffel et al. (2014)	√	√	√	✗	✗	Sum	Homomorphic
Chan et al. (2007)	✗	✗	✗	√	✗	Count, min, max, average, median	Symmetric
Kumar et al. (2015)	✗	√	✗	✗	✗	Not defined	Homomorphic
Rastogi and Nath (2010)	√	✗	√	✗	✗	Sum	Homomorphic
Ma et al. (2015)	✗	✗	✗	✗	✗	Not defined	✗
Othman et al. (2015)	✗	√	✗	√	✗	Not defined	Homomorphic
Raj et al. (2016)	✗	✗	✗	✗	✗	Sum	Homomorphic
He et al. (2016)	✗	√	✗	√	✗	Sum	Homomorphic
Manni et al. (2015)	✗	✗	✗	✗	✗	Not defined	✗
Fan et al. (2014)	✗	√	✗	✗	✗	Sum	Homomorphic
Grining et al. (2016)	√	✗	✗	√	✗	Sum	Homomorphic
Castelluccia et al. (2005)	√	√	✗	✗	✗	Sum,average,variance	Homomorphic
Conti et al. (2009)	√	✗	✗	√	✗	Not defined	Asymmetric
Villas et al. (2013)	✗	✗	✗	✗	✗	Not defined	✗
Fu et al. (2015)	✗	√	✗	√	✗	Not defined	Not defined
Yoon et al. (2014)	✗	√	✗	√	✗	Sum	Symmetric
Kim et al. (2013)	√	√	√	√	✗	Sum	Symmetric
He et al. (2009)	√	√	√	√	✗	Sum	Symmetric
PAgIoT	√	√	√	√	√	Count	Homomorphic

et al. (2013), and Yoon et al. (2014). The use of cryptography is demanding to manage this issue (He et al., 2007; Wenbo et al., 2008; Applebaum et al., 2010; Buttyán and Holczer, 2010; Kumar et al., 2015; Li et al., 2014; Horey et al., 2007; He et al., 2009; Rieffel et al., 2014; Castelluccia et al., 2005; Fu et al., 2015; Kim et al., 2013; Yoon et al., 2014), as well as randomized arrays or negative surveys (Groat et al., 2011).

Collusion resistant is another security requirement (He et al., 2007; Applebaum et al., 2010; He et al., 2009; Rieffel et al., 2014; Rastogi and Nath, 2010; Kim et al., 2013). Particularly, He et al. (2007, 2009) present collusion resistant proposals as long as clusters are big enough. In Applebaum et al. (2010) an aggregation scheme is proposed in which nodes, a proxy and a data base (DB) are involved. Messages are sent through the proxy to finally reach the DB. Collusion resistant is guaranteed when the proxy and the DB do not collude. Rieffel et al. (2014) apply multiple keys for decrypting values such that colluding parties are not able to infer all required keys. A different approach is taken in Rastogi and Nath (2010), collusion is avoided by trusting users whose values are aggregated. Conversely, in Kim et al. (2013) authors mention the protection against collusion but it is not particularly described how it is managed.

Verifying that aggregation is correctly performed without polluted results is also addressed by several proposals (Xiaoying et al., 2014; Chan et al., 2007; He et al., 2016, 2009; Li et al., 2014; Othman et al., 2015; Conti et al., 2009; Fu et al., 2015; Grining et al., 2016; Kim et al., 2013; Yoon et al., 2014). A monitoring neighbor scheme is proposed in He et al. (2009), where neighbors of intermediate aggregator nodes monitor their behavior. Paper (Xiaoying et al., 2014) presents an integrity-verification algorithm based on identifying unauthentic, superfluous or incomplete data, as well as empty results. In Wenbo et al. (2008) redundancy checks are applied and in Chan et al. (2007) a proof of correctness based on hash functions is presented. On the other hand, Li et al. (2014) focuses on data pollution against malicious intermediate aggregator nodes. Authors propose that each leaf node verifies its parent's aggregation, considering that nodes are organized as a tree. He et al. (2016) apply digital signatures to provide message integrity apart from confidentiality. With the same purpose (Fu et al., 2015; Othman et al., 2015) apply elliptic curve digital sig-natures. Conti et al. (2009) address this issue from the point of view of data lost, twin-keys are applied to obfuscate real values sent to the sink. In Yoon et al. (2014) the imaginary part of a complex number is used to verify integrity by aggregators and the sink node. A query-response technique called PIR proposes (Kim et al., 2013) to verify messages integrity.

Table 5 presents a summary of related works identifying the management of anonymity of nodes, privacy preservation against observers, collusion resistant, verifiable aggregation, correlatable aggregation, aggregation purpose and encryption scheme. Indeed, these features are the goals addressed by PAgIoT. Here, √ means that the property is fulfilled, whereas represents ✗ that the property is not satisfied. For the sake of comparison, PAgIoT is included as well.

HDA (Kim et al., 2013) and iCPDA (He et al., 2009) are the proposals whose security requirements are analogous to the ones proposed in PAgIoT and then, they are briefly described.

In particular, iCPDA proposes a clustering algorithm to enforce data integrity and privacy. It bases on PDA (He et al., 2007) to perform the clustering and the aggregation. After clustering and the enforcement of a key establishment protocol the aggregation starts. Aggregation focuses on addictive properties of polynomials. Nodes within a cluster compute a value of a polynomial for the remaining nodes in the cluster and send values to each appropriate node. Once values from all nodes are received, each of them sums received values and the result is sent to the cluster leader. This latter node computes a final polynomial addition to create an aggregated message. iCPDA proposes other phase in which each cluster leader when receiving messages (of aggregated data) from downstream leaders, makes results accessible to all of them. If some leader identifies that the received result is incorrect, a misbehavior is detected.

HDA proposes an aggregation scheme to also enforce data privacy and integrity. Hilbert-curve algorithm is applied to transform one-dimensional data into two-dimensional data. Once clusters are created and pairs of nodes have computed and shared keys based on an elliptic curve Diffie–Hellman algorithm (ECDH), the aggregation is performed. Data sent from one node to another is encrypted through a Hilbert-curve algorithm and sent to the aggregator which adds received values to create the aggregated result. Then, another phase starts to verify data integrity. Parent nodes send to all child nodes a PIR message, that is a message to request children the computation of a value of their Hilbert curve. Nodes response and parents verify if the computation is correct and thus integrity holds.

In the light of this study some weaknesses, addressed by PAgIoT, are not considered by current proposals. In particular, PAgIoT proposes the privacy-preserving aggregation of data enabling correlation of original values while also managing anonymity of sending nodes, collusion resistant and verifiable aggregation.

7. Conclusions

The huge amount of data spread worldwide provides the need for data analysis, being aggregation a challenging activity in this regard. Security cannot be taken for granted, being data authenticity and users privacy major issues. This paper presents PAgIoT, a privacy-preserving aggregation protocol for Internet of Things (IoT) scenarios. It contributes to the aggregation of data based on attribute-based queries and the homomorphic Paillier cryptosystem. In fact, PAgIoT achieves correlatable aggregation, enabling the sink to get not only the overall aggregated value, but also the correlation between attribute values. A misbehaving aggregator detection algorithm is also introduced to avoid malicious aggregators. Results of the evaluation show that PAgIoT is feasible for resource-constrained environments (such as IoT). Furthermore, it is resistant to eavesdropping, collusion and pollution attacks, and it is practical using a wide set of attributes and in relative large networks (e.g. using queries composed of 10 attributes in networks of 2^{16} nodes). Moreover, PAgIoT has been compared against two previous works (referred to as HDA and iCPDA) that provide with the same security properties. Results show that PAgIoT outperforms both proposals considering the amount of keys at stake and the number of messages sent.

Future work will be focused on the use of the proposed protocol in the cybersecurity context which involves, among other issues, risks, threats and vulnerabilities management. The use of other cryptographic schemes (particularly lightweight ones) will be assessed as well.

Acknowledgments

This work was partially supported by the MINECO grant TIN2013-46469-R (SPINY: Security and Privacy in the Internet of You) and the CAM grant S2013/ICE-3095 CIBERDINE-CM (CIBERDINE: Cybersecurity, Data, and Risks) funded by the Autonomous Community of Madrid and co-funded by European funds.

References

Alcaraz, C., Najera, P., Lopez, J., Roman, R., 2010. Wireless sensor networks and the internet of things: Do we need a complete integration? In: 1st International Workshop on the Security of the Internet of Things (SecIoT'10).

Applebaum, B., Ringberg, H., Freedman, M., Caesar, M., Rexford, J., 2010. Collaborative, privacy-preserving data aggregation at scale. In: Privacy Enhancing Technologies, pp. 56–74.

Bae, M., Kim, K., Kim, H., 2016. Preserving privacy and efficiency in data communication and aggregation for ami network. J. Netw. Comput. Appl. Elsevier, http://www.sciencedirect.com/science/article/pii/S108480451500168X.

Benaloh, J.C., 1986. Secret sharing homomorphisms: keeping shares of a secret. In: Advances in Cryptology CRYPTO'86. Springer, Berlin Heidelberg, pp. 251–260.

Buttyán, L., Holczer, T., 2010. Perfectly anonymous data aggregation in wireless sensor networks. In: MASS 2010. IEEE 7th International Conference on Mobile Adhoc and Sensor Systems, pp. 513–518.

Castelluccia, C., Mykletun, E., Tsudik, G., 2005. Efficient aggregation of encrypted data in wireless sensor networks. In: The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2005. MobiQuitous 2005. The Second Annual International Conference, IEEE, pp. 109–117.

Chan, H., Perrig, A., Przydatek, B., Song, D., 2007. SIA: secure information aggregation in sensor networks. J. Comput. Secur. 15 (1), 69–102. Cheikhrouhou, O., 2015. Secure group communication in wireless sensor networks: a survey. J. Netw. Comput. App., Elsevier.

Chen, H., Lou, W., 2015. On protecting end-to-end location privacy against local eavesdropper in wireless sensor networks. Pervas. Mobile Comput. 16, 36–50.

Chen, Y., Nahum, E., Gibbens, R., Towsley, D., Lim, Y., 2012. Characterizing 4G and 3G

Networks: Supporting Mobility with Multi-Path TCP. Technical Report. University of Massachusetts Amherst, Technical Report: UM-CS-2012-022.

Chouikhi, S., El Korbi, I., Ghamri-Doudane, Y., Saidane, L.A., 2015. A survey on fault tolerance in small and large scale wireless sensor networks. Comput. Commun. 69, 22–37.

Christin, D., Reinhardt, A., Mogre, P.S., Steinmetz, R., 2009. Wireless sensor networks and the Internet of things: selected challenges, pp. 31–34.

Conti, M., Zhang, L., Roy, S., Di Pietro, R., Jajodia, S., Mancini, L.V., 2009. Privacy-preserving robust data aggregation in wireless sensor networks. Secur. Com-mun. Netw. 2 (2), 195–213.

Fan, C.-I., Huang, S.-Y., Lai, Y.-L., 2014. Privacy-enhanced data aggregation scheme against internal attackers in smart grid. IEEE Trans. Ind. Inform. 10 (1), 666–675.
Fiadino, P., Schiavone, M., Casas, P., 2015. Vivisecting WhatsApp in cellular networks: Servers, flows, and quality of experience. In: Traffic Monitoring and Analysis. Springer International Publishing, pp. 49–63.
Fu, S., Ma, J., Li, H., Jiang, Q., 2015. A robust and privacy-preserving aggregation scheme for secure smart grid communications in digital communities. Wiley Online Libr.
Grining, K., Klonowski, M., Syga, P., 2016. Practical Fault–Tolerant Data Aggregation.
Groat, M., W.H., Forrest, S., 2011. KIPDA: k-indistinguishable privacy-preserving data aggregation in wireless sensor networks. In: INFOCOM, 2011 Proceedings IEEE, pp. 2024–2032.
Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M., 2013. Internet of Things (IoT): a vision, architectural elements, and future directions. Future Gener. Comput. Syst. 29 (7), 1645–1660.
Hai, J., Shen, F., Chen, S., Li, K., Jeong, Y., 2015. A secure and scalable storage system for aggregate data in IoT. Future Gener. Comput. Syst. 49, 133–141.
He, W., Liu, X., Nguyen, H., Nahrstedt, K., Abdelzaher, T., 2007. PDA: privacy-preserving data aggregation in wireless sensor networks. In: INFOCOM 2007. 26th IEEE International Conference on Computer Communications, pp. 2045–2053.
He, W., Liu, X., Nguyen, H., Nahrstedt, K., 2009. A cluster-based protocol to enforce integrity and preserve privacy in data aggregation. In: ICDCS Workshops09. 29th IEEE International Conference on Distributed Computing Systems Workshops, pp. 14–19.
He, D., Kumar, N., Lee, J.-H., 2016. Privacy-preserving data aggregation scheme against internal attackers in smart grids. Wireless Netw. 22 (2), 491–502.
Heidemann, J., Silva, F., Intanagonwiwat, C., Govindan, R., Estrin, D., Ganesan, D., 2001. Building efficient wireless sensor networks with low-level naming. In: SOSP '01. Eighteenth ACM Symposium on Operating Systems Principles, pp. 146–159.
Horey, J., Groat, M.M., Forrest, S., Esponda, F., 2007. Anonymous data collection in sensor networks. In: MobiQuitous 2007. Fourth Annual International Con-ference on Mobile and Ubiquitous Systems: Networking Services, pp. 1–8.
Kim, Y.-K., Lee, H., Yoon, M., Chang, J.-W., 2013. Hilbert-curve based data aggrega-tion scheme to enforce data privacy and data integrity for wireless sensor networks. Int. J. Distrib. Sensor Netw.
Kumar, M., Verma, S., Lata, K., 2015. Secure data aggregation in wireless sensor networks using homomorphic encryption. Int. J. Electron. 102 (4), 690–702.
Kusy, B., Abbott, D., Richter, C., Huynh, C., Afanasyev, M., Hu, W., Brünig, M., Ostry, D., Jurdak, R., 2014. Radio diversity for reliable communication in sensor networks. ACM Trans. Sensor Netw. (TOSN) 10 (2), 32.
Li, H., Li, K., Qu, W., Stojmenovic, I., 2014. Secure and energy-efficient data aggregation with malicious aggregator identification in wireless sensor networks. Future Gener. Comput. Syst. 37, 108–116.
Ma, T., Liu, Y., Zhang, Z.-j., 2015. An energy-efficient reliable trust-based data aggregation protocol for wireless sensor networks. Int. J. Control Autom. 8 (3), 305–318.
Madden, S., Franklin, M., Hellerstein, J., Hong, W., 2002. TAG: a tiny aggregation service for ad-hoc sensor networks. SIGOPS Oper. Syst. Rev. 36, 131–146.
Manni, R., Rai, M., Kansal, L., 2015. Edrina: enhanced data routing for in-network aggregation algorithm for wsns. Asian J. Inf. Technol. 14 (8–12), 276–280.
Othman, S.B., Bahattab, A.A., Trad, A., Youssef, H., 2015. Confidentiality and integrity for data aggregation in wsn using homomorphic encryption. Wireless Personal Commun. 80 (2), 867–889.
Paillier, P., 1999. Public-key cryptosystems based on composite degree residuosity classes. In: Advances in Cryptology, Proceedings of the EUROCRYPT 99. Springer Berlin Heidelberg, pp. 223–238.
Raj, G., Thanjaivadivel, M., Viswanathan, M., Bindhu, N., 2016. Efficient sensing of data when aggregated with integrity and authenticity. Indian J. Sci. Technol. 9 (3).
Rajagopalan, R., Varshney, P., 2006. Data Aggregation Techniques in Sensor Networks: A Survey, no. paper 22.
Rastogi, V., Nath, S., 2010, Differentially private aggregation of distributed time-series with transformation and encryption. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. ACM, pp. 735–746.
Ren, F., Zhang, J., Wu, Y., He, T., Chen, C., Lin, C., 2013. Attribute-aware data aggregation using potential-based dynamic routing in wireless sensor networks. IEEE Trans. Parallel Distrib. Syst. 24 (5), 881–892.
Rieffel, E., Biehl, J., Lee, A., van Melle, W., 2014. Private aggregation for presence streams. Future Gener. Comput. Syst. 31, 169–181.
Riiser, H., Vigmostad, P., Griwodz, C., Halvorsen, P. 2013. Commute path bandwidth traces from 3G networks: analysis and applications. In: Proceedings of the 4th ACM Multimedia Systems Conference, MMSys 13. ACM, pp. 114–118.
Rodero, I., Guim, F., Corbalan, J., Fong, L., Sadjadi, S., 2010. Grid broker selection strategies using aggregated resource information. Future Gener. Comput. Syst. 26 (1), 72–86.
Sheets, L., Callaghan, F., Gavino, A., Liu, F., Fontelo, P., 2012. Usability of selected databases for low-resource clinical decision support. Appl. Clin. Inform. 3 (3), 326–333.
Singh, V.K., Dutta, M., 2014. Analyzing cryptographic algorithms for secure cloud network. CoRR.
Stern, J., (Ed.), 2003. Advances in cryptology. In: Proceedings of the EUROCRYPT 99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2–6, 1999, Springer.
Villas, L.A., Boukerche, A., Ramos, H.S., de Oliveira, H.A., de Araujo, R.B., Loureiro, A. A., 2013. Drina: a lightweight and reliable routing approach for in-network aggregation in wireless sensor networks. IEEE Trans. Comput. 62 (4), 676–689.
Wenbo, H., Hoang, N., Xue, L., Nahrstedt, K., Abdelzaher, T., 2008. iPDA: an integrity-protecting private data aggregation scheme for wireless sensor networks. In: MILCOM 2008. Military Communications Conference, pp. 1–7.
Xiaoying, Z., Lei, D., Hui, P., Hong, C., Deying, L., Cuiping, L., 2014. Achieving efficient and secure range query in two-tiered wireless sensor networks. In: IWQoS 2014. IEEE 22nd International Symposium of Quality of Service, pp. 380–388.
Yoon, M., Jang, M., Kim, H.-I., Chang, J.-W., 2014. A signature-based data security technique for energy-efficient data aggregation in wireless sensor networks. Int. J. Distrib. Sensor Netw.