This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

Context-driven Granular Disclosure Control for Internet of Things Applications

Arezou Soltani Panah, Ali Yavari, Ron van Schyndel, Dimitrios Georgakopoulos, and Xun Yi

Abstract—The Internet of Things (IoT) represents a technology revolution transforming the current environment into a ubiquitous world, whereby everything that benefits from being connected will be connected. Despite the benefits, the privacy of these *things* becomes a great concern and therefore it is imperative to apply privacy preservation techniques to IoT data collection. One such technique is called data obfuscation in which data is deliberately modified to blur the sensitive information, while preserving the data utility. The current obfuscation techniques, however, focus on the privacy of published datasets shared with untrusted parties. The high connectivity and distributed nature of IoT, opens up the possibility of privacy compromise *before* obfuscation can take effect, and therefore privacy enforcement should be deployed at earlier stages. Additionally, classical privacy treatments are too restrictive for IoT, where coarser/finer data details should be revealed for different applications. Motivated by these challenges, we propose a framework for privacy preservation in IoT environments that is capable of multi-granular obfuscation by enforcing context-driven disclosure policies. Then, we customize our framework for a smart vehicle system and make use of data stream watermarking techniques to protect privacy at different stages of the data lifecycle. To address possible concerns about additional performance overhead, we show the burden to be very lightweight, thus validating the suitability of ubiquitous use of our framework for IoT settings.

Index Terms—privacy preservation, data obfuscation, digital watermark, Internet of Things, context, pseudorandom numbers.

1 Introduction

The proliferation of Radio Frequency Identification (RFID), sensors, mobile devices and ubiquitous Internet access has sparked a new era of ''Internet of Things'' (IoT). In the IoT vision, connectivity extends to the point, where almost anything can connect to the Internet [1]. In spite of such remarkable connectivity, the privacy of individual associates with these things is a grant challenge for a global IoT deployment. Many IoT applications such as remote healthcare monitoring, energy grids, navigation systems, homeland security and defense gather sensitive information that might violate individual privacy [2], [3], [4].

One of the conventional approaches for privacy preservation is using an obfuscation function in order to reduce the granularity (e.g., accuracy, fidelity and specificity) of information. Such obfuscation techniques include generalization and suppression, data masking and perturbation methods such as random noise addition and data swapping. These methods are useful for privacy preservation of published datasets where the data is distilled based on the trust level of the data consumer, preferably in an irreversible manner to maximise data protection [5], [6]. Directly applying of these obfuscation techniques for IoT data is not feasible. To explain why, we must first understand the main privacy requirements of collected data in an IoT setting.

First, IoT data is collected from a myriad of distributed things that maybe owned by different organisations or people. This establishes a federated environment whereby the ownerships and control of things and their data is distributed. Therefore, the possibility of privacy breaches increases compared to a centralised data store, where data is collected and stored in a seemingly isolated device and is controlled by a single person or organisation. For instance, if an adversary targets an Intranet of things in an IoT-enabled hospital setting, he might be able to access medical sensor data before they are stored in the hospital database [7], [8]. In such circumstances, even applying the strongest privacy preservation method at the time of data storage may not be effective as the privacy has already been violated. Hence, it is imperative to protect the privacy of the data during the earliest phases of its lifecycle. The essence of a comprehensive solution to protect the privacy of IoT data through the whole data life-cycle is also addressed recently by Bertino [9], in particular during data collection and data sharing phases.

Second, protection of large volumes of data generated by IoT devices is very complex, in particular when IoT data is used by unknown applications without their owner's consent [7]. This privacy challenge directly related for one of the main benefits of IoT, i.e., allowing third-party applications to use the IoT devices and data produced by others. To protect privacy, the owner of any IoT data needs to control the disclosure of his or her data. Traditional authorisation models lead to a binary decision, that is whether the access to data is granted or denied. However, IoT applications can benefit by having access to different granular information. For instance, a taxi driver may be willing to reveal his precise whereabouts to the taxi company he is working with for safety reasons. In contrast, he may prefer to send only his cloaked location (for instance the current suburb) to traffic authorities. This way, the data owner is not only able to control *whom* to share data with, but also is able to determine *how much* he is willing to share.

Third, providing the aforementioned privacy preservation introduces additional complexities in terms of provisioning and management of extra information required for

• *A.Soltani Panah, A. Yavari, R. van Schyndel and X. Yi are with School of Science, RMIT University. D. Georgakopoulos is with Swinburne University.Corresponding author email: arezou.soltanipanah@rmit.edu.au*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

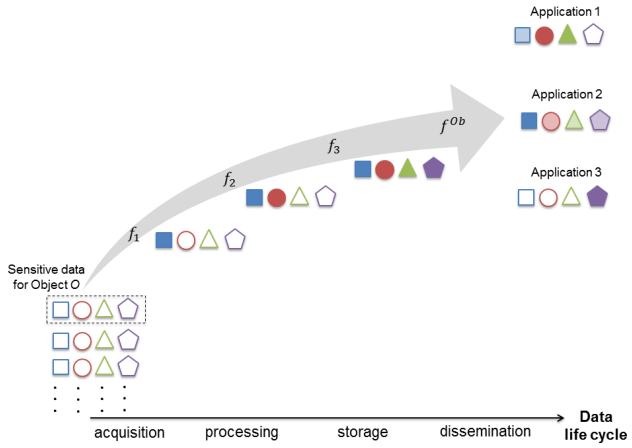

Fig. 1. Multi-stage privacy protection. The color intensity represents varying noise granularity.

determining the granularity of information. For example, in the case of the taxi driver, he might even restrict his exact whereabouts to the taxi company during the night trips in specific suburbs or a patient might allow his physician to access his health record only during the examination. Therefore, we will need to maintain the driver or patient's preferences for sharing their information and this needs to be provided efficiently in real-time for practical usages. Such information in IoT literature is called *context* and is often used to make more flexible and intelligent decisions.

In this work, we propose a novel framework for privacy preservation of IoT data. The general idea behind our framework is depicted in Fig. 1. Assume O is a privacy conscious object (or a thing). This object has four sensitive data items that are shown with different shapes (a square, circle, triangle, and pentagon). Depending on the contextual information, the *granularity* of accessed data varies. Granularity, in this context, refers to the 'blurred' precision of the data. This 'blurred' precision is dependent on access, application, user or usage properties. We call this term granularity to distinguish it from the data 'precision', which is an absolute measure based on sensor properties. For this purpose, we use an obfuscation function that returns a masked version of O, i.e. $f^{Ob} : O \rightarrow O'$ depending on the desired granularity level gl. One could consider O and O' as sample and spatiotemporal precision variation respectively.

Additionally, for end-to-end protection of sensitive data, we apply multiple privacy preservation functions, each of which is denoted by f_i with index i for the i^{th} function, on the data before the dissemination phase. This way, our context-driven framework mimimizes any potential loss of privacy, but also takes advantage of efficient contextualization techniques such as [10] and [11] that have been developed specifically for IoT to provide the scalability and efficiency required by IoT applications.

1.1 Contribution

The focus of this paper is designing a reasonably secure, fast, and lightweight privacy preserving solution to meet the requirements of IoT applications. Central to our design is the notion of *flexible privacy* –the data owner should not only be able to control the data access, but also the accuracy of his data (whom, how much). This control can be done based on contextual information of either data owner (e.g., location, time, emergency situations) or data consumer (e.g., role, physical co-location, or time of access). The more contextual the information, the finer is the disclosure granularity that can be achieved. In our suggested framework, we also advocate the privacy protection at multiple phases of data life-cycle to afford maximum data protection (this is different from data transmission security schemes). Previous data obfuscation methods are inferior to ours for IoT settings, where dynamic obfuscation is required, but also data should be protected at multiple stages before the actual delivery.

In summary, the contributions of this work are:

- Introducing a conceptual framework for end-to-end privacy preservation of IoT data through its entire life-cycle. The proposed privacy preservation framework permits a variety of alternative implementations.
- Proposing a context-driven granular obfuscation technique for IoT data.
- Describing the design and implementation of a smart vehicle system that uses the proposed framework and context-driven granular obfuscation (based on visible data watermarking techniques).
- Illustrating the efficiency and scalability of the context-driven granular obfuscation framework and related performance assessment.

The rest of this paper is structured as follows. In the next section, we review the works related to the overall problem domain. In Section 3, we describe our conceptual privacy preservation framework in detail, and then in Section 4, we customize our framework for a smart vehicle system, review the related works specific to smart vehicle data, and provide a lightweight multistage privacy preserving method based on digital watermarking. The performance results are presented in Section 5. We conclude the paper in Section 6.

2 Work Related to Privacy Preservation

There are clear parallels between our framework and two popular techniques for privacy preservation namely access control and disclosure control. The purpose of access control is to limit the actions or operations that a legitimate user of a system can perform, whereas information disclosure control aims at publishing or sharing data such that the privacy of individuals is not compromised. There has been a considerable volume of research on developing both access and disclosure control methods. We do not attempt to survey all these techniques here. However, we summarize bellow the most related methods to our framework.

The most common access control mechanisms include Discretionary Access Control (DAC), Lattice-Based Access Control (LBAC), and Role-based Access Control (RBAC) [12]. The DAC model is discretionary in the sense that the owner of the requested resource controls the access to that resource. Each access request is checked against the specified authorizations. If there exists an authorization stating that the user can access the resource in the specific mode (read or write), the access is granted, otherwise it is denied. LBAC, also known as mandatory access control, enforces one-directional information flow on the basis of a predefined lattice of security labels which are associated

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

with every resource and user in the system. A user is granted authorization to access a given resource if some relationship exists between the user and that resource in the lattice. RBAC determines access level via the role abstraction, rather than by just identity or clearance of the requester. In this model, a role is a semantic construct which is often a representation of a job in an organisation

In an IoT setting where the data and access control policies dynamically change, the above access models are not suitable to adapt these changes. In order to address such compliance requirements, the next line of research enriches access polices with contextual information. In this work, we define context as any information that characterizes the circumstances of the sensed data. In this regard, several extensions to the basic RBAC model are proposed to incorporate context variables such as Generalized RBAC [13], which introduces environmental information to *activate* roles. Likewise, a context-driven RBAC model was proposed for health-care [14]. A major deficiency of these approaches is that, data access is either granted or denied, and therefore these do not account for the specificity of information which a multi-granular data model allows.

Generally speaking, granularity represents the units of measure of the data and can be defined on any dimensions [15]. In order to provide flexibility for situations where different granularity is needed, information disclosure control, and in particular data obfuscation is advantageous. Data obfuscation is a technique to purposely degrade the sensitive information to a desired granularity.

Existing obfuscation approaches include, but are not limited to data perturbation, generalization and suppression, data anonymization and random sampling [5]. For instance, in [16] the authors compared different noise addition models to find the optimal amount of noise required to obfuscate data while preserving privacy. As discussed in Introduction, preserving privacy in an IoT setting only at the time of data dissemination does not ensure complete data protection and the whole data lifecycle needs to be considered to assure end-to-end privacy.

TABLE 1
Notations

Symbols	Description
f	Privacy preservation function
f^{Ob}	Obfuscation function
k	Number of privacy preservation functions
d	Number of granularity level
gl	Granularity level for disclosure control
C^{app}, C^{data}	Application context and Data context
CS^{app}, CS^{data}	Application context set and Data context set for a given query
n ,m	Size of CS^{app} and CS^{data}
cid^{app}, cid^{data}	Atomic context identifier of Applications and Data present in disclosure rules
CID^{app}, CID^{data}	Compound context identifier of Application and Data for a given query
Tr_{MO}	Trajectory of moving object MO
$< p_i, t_i >$	Position coordinates of the form (x_i, y_i) with time stamp t_i
(α_x, α_y)	Scale factor or watermark amplitude for (x_i, y_i)
$DLFSR$	Dynamic Linear Feedback Shift Register
$(l_1, iv_1, poly_1)$	Number of registers, initial value, and polynomial of the primary LFSR in the DLFSR generator
$(l_2, iv_2, poly_2)$	Number of registers, initial value, and polynomial of the secondary LFSR in the DLFSR generator
π	Secure permutation function
$Hash$	Hash functions, example SHA-1 or MD5
l, b	Hash output size and buffer length, respectively

3 Context-driven Multi-stage Privacy Preservation Framework

In this section, we introduce our conceptual privacy preserving framework that has two main parts: multi-stage privacy protection; and dynamic obfuscation.

For convenience in following our discussion, the summary of notations that we use in the rest of the paper is given in Table 1.

3.1 Multi-stage Privacy Protection

Our first goal is to protect data before the data dissemination stage. In Introduction, we defined O as a privacy cautious object with k sensitive data items, the privacy of which is important to be preserved. We assume the sensitive data is enriched with *pseudo-sensitive* context, meaning that the privacy of this contextual information does not need to be preserved necessarily, but affects the disclosure of sensitive data. For instance, assume a smart vehicle that periodically reports its vehicle id and GPS coordinates along with the time and speed. The vehicle id and GPS coordinates could be the sensitive data whereas, the time and speed of the vehicle can be always revealed for safety reasons. In this case a driver might decide not to share his exact GPS coordinates during the day, but may do so during night trips. Even though time is not sensitive information here, it changes the disclosure of the sensitive data.

We then define k privacy preservation functions to protect the privacy of k sensitive data items, each of which is denoted by f_i to protect the privacy of the corresponding sensitive data d_i. For the previous example, k was 2 (vehicle id and GPS coordinates) and therefore two functions are needed. The three main requirements for such functions are:

- Security to guarantee the privacy of the protected data,
- Reversible privacy, in a sense that the original data can be re-created from obfuscated data during run time access, and
- Lightweight functionality to meet the IoT constraint requirements.

The first requirement necessitates the existence of *trapdoor* information held by a legitimate entity, such as a private key or seed value. The second requirement is reversible privacy that is actually needed to achieve our dynamic obfuscation goal. Therefore methods like generalization and suppression, or random noise additions are not acceptable as they transform data into another form in an irreversible manner. The third requirement is an emphasis on low computational complexity of the functions f. For instance, relying on heavyweight cryptographic mechanisms such as homomorphic encryption [17] to obtain privacy guarantees is too expensive for devices with tight resource constraints.

Note that the privacy preserving functions can be applied in different stages of the data lifecycle based on the security

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

requirements. For example, in Fig. 1, f_1 occurs at the collection point, while f_2 and f_3 are applied at data dissemination stages. Ideally, a successful security solution should provide an end-to-end protection of data, which means that the data, should be protected from the data acquisition point to the final destination. The real-time processing and constraint computational capacity of IoT devices make the end-to-end protection an ambitious goal therefore, a trade-off must be sought to provide *reasonable security* in an IoT setting. In reference [9], the authors recommended privacy protection at data collection and data dissemination stages.

Based on the stage at which the privacy function is intended to be applied, one can decide about the security aspect of the function. For instance, if the function f resides on sensors (data collection stage), it should have low complexity. In contrast, if the protection is taking place at the storage stage by a high-power computer, a more complex function is acceptable. Apart from the stage, the 3V features of big data, namely Volume, Variety and Velocity should be accommodated in the design of privacy functions. For instance, if the variety of sensitive data is not high (such as vehicle id), an RSA key with 256 bit key length can be used, but for time and location data, even an encryption method with relatively small key and ciphertext decreases the system efficiency (in terms of query response time).

For the rest of the paper, we shall use the term data and context to mean only the *sensitive* data and *pseudo-sensitive* context respectively, unless otherwise specified.

3.2 Dynamic Obfuscation

So far, only privacy protection of data before the dissemination stage has been discussed. Now, we describe how flexible privacy can be achieved at the time of data dissemination. In our framework, an obfuscation method is used that offers coarser or finer granularity disclosure based on the information that is present in a contextual query. A contextual query is defined as a query with contextual information being passed in the search query, such as role or spatiotemporal attributes of data requester. This type of query is not issued by the data requester itself; instead we assume there is a contextualization service such as [10] that enriches the search query by the contextual information.

Before describing our dynamic obfuscation method, we first distinguished between two types of context: Data and Application Context.

Definition 1: The Data Context (C^{data}) refers to the context associated with the collected data (such as time and speed for the smart vehicle example). The Application Context (C^{app}) includes the contextual information in which the queries are issued such as the role of the data requester (physician, nurse, police, etc) or physical co-location (of a physician with the patient).

The C^{data} and C^{app} information have direct impact on data obfuscation. To this end, we introduce an obfuscation function f^{Ob}, the goal of which is to provide varying degree of content information (granularity) to the application (data consumer) based on both C^{data} and C^{app} values. In other words, $f^{Ob}(d)$ is a version of the original data with a lesser degree of information precision depending on the allowed disclosure rules. This necessitates the existence of disclosure rules to determine the level of data obfuscation that we explain next.

3.2.1 Scaling up dynamic obfuscation

The question that arises here is whether flexible privacy can be accommodated with respect to big data characteristics. Consider the following example.

Example 1. Assume, Alice wants to find the current location of Bob. Some example of disclosure policies are listed as follows:

- Rule 1: If *Alice* is a paramedic currently located at *Melbourne*, and *Bob* has an incident in *Melbourne*, the precise location of Bob is shared with Alice.
- Rule 2: If *Alice* is an employee with *Melbourne* city branch of *13cabs*, and *Bob* has a trip at *Footscray* suburb over night, he only shares his street name with Alice .
- Rule 3: If *Alice* is an employee in *VicRoads*, and *Bob* is driving in *Heidelberg Rd*, *Chandler Highway*, or *Malvern Rd* during weekdays, he shares his suburb name with *Alice* only during her working hours.

Given policies such as the above, it is clear that exhaustively testing every context of both data and application (C^{data}, C^{app}) for every request is prohibitively expensive. In the above scenario, only one data requester (Alice) and one *thing* (Bob's GPS reader) was considered. In IoT visions, millions of things are connected to the Internet, and will generate big data at unprecedented scale (Volume characteristic of big data). In such a setting, often a subset of things is queried. Apart from that, some contextual information such as time and location may frequently be updated and therefore the response latency can be dramatically high (Velocity characteristic of big data).

For traditional systems, there are many privacy enhancing techniques, among which differential privacy (DP) is a notable privacy notion for releasing aggregate statistics. While DP has rigorous privacy guarantee for static datasets, it cannot be applied for an IoT setting. The main reasons include: (1) DP assumes that data is perfectly secure (a curator is perfect) and the only concern is how to share data without violating the privacy of individuals. As explained in Introduction, privacy might be already violated during IoT data collection; (2) DP is too complex to be applied for a streaming data model. As we will explain later, one needs to calculate all possible combinations between neighbouring data points to find the amount of noise to be added to original data. That is to say, there is a privacy guarantee but with the cost of expensive calculation; (3) There is no privacy guarantee for queries about specific individuals.

To address the above issues and in making our obfuscation scheme scalable, we consider two aspects in our framework design. First, we make the flexible privacy dependent on policies, so called *disclosure rules*, rather than all the possible values for individual contexts. Once a query is issued, the existing rules are scanned to find a match. If there is a match, the data is obfuscated based on the stated granular rules, otherwise the data is not revealed at all[1]. This way the complexity is relative to the number of existing rules, not the

1. Here, the default behaviour of the system is zero disclosure. It is possible to grant full disclosure if there is no rules for the requested data in the system.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

number of all possible combination of multiple contexts (i.e both C^{data} and C^{app}). Second, we propose a rule-indexing scheme to further speed up finding the corresponding rules. This scheme is based on prime factorization to scale up our framework, which is described next.

3.2.2 Disclosure Rule-Indexing Scheme

In this work, we have made the assumption that the cost of answering a query in terms of response time is proportional to the number of disclosure rules that needs to be examined. To reduce this cost, this section presents a rule-indexing scheme, the idea of which is borrowed from ConTaaS, an Internet-scale contextualization tool described in [10]. Following this approach, instead of scanning the entire table consisting of disclosure rules, all these rules are indexed in such a way that the query itself determines only a subset of disclosure rules to be examined. Because the scheme only uses simple arithmetic operations like multiplication and unique factorization (of composite integers), we can reap dramatic benefits for accelerating access to the relevant rules and subsequently decreasing query response time.

Before explaining our scheme, recall that a disclosure rule maps a set of conditions to a granularity level ($conditions \rightarrow gl$). Our framework, does not put any limitation for expressing gl values in terms of logical or numeral granular data representation. We assume there are d possible granularity levels $gl_1, gl_2, ..., gl_d$ for a particular data item.

Additionally, we need to reach a consensus on defining the disclosure rules. Generally, regular expressions can facilitate rule representations to support even complex rules. There are also more specific options such as Platform for Privacy Preferences (P3P) [18] or Semantic Web Rule Language (SWRL) [19] that can be used. In this work, we have chosen, SWRL to express disclosure rules, but other languages can be used interchangeably. For instance the first rule of the Example 1, can be expressed using SWRL as

```
(paramedic(?requester) ∧ hasCurrent-
Location(?requester, ?Melbourne) ∧
hasIncident("Bob") → shareLocation
(?maximumPrecison).
```

The disclosure rules need to be defined by a security manager or equivalent who is in charge of preserving privacy of things. Although these rules are determined before the system implementation, the security manager can dynamically add new rules or delete some of them when needed. Once the disclosure rules are defined, the next step is indexing them as follows.

Our proposed indexing scheme consists of two steps: context labeling and rule translation. First, an identifier is assigned to every context that is present in the disclosure rules, whose value is the next available prime number. We denote the prime identifiers for application contexts and data contexts with cid^{app} and cid^{data}, respectively. The assigned identifiers will be stored in two separate tables, namely application context identifiers (ACI) and data context identifiers (DCI) tables. Second, every existing disclosure rule is translated into two compound identifiers, one for application and one for data. For this purpose, atomic and compound context identifiers are defined as follows.

Definition 2. Atomic and Compound context Identifier. An atomic context identifier is a prime number that is assigned to the context present in a disclosure rule. A compound context identifier is constructed by multiplying several atomic context identifiers. In contrast to atomic identifiers that are represented by small letters cid^{app} and cid^{data}, compound identifiers are shown by capital letters, i.e. CID^{app} and CID^{data}.

In fact, the compound context identifiers can greatly help in finding the relevant disclosure rules without going through all of the rules. For an existing disclosure rule, the two mentioned compound identifiers are calculated and the resulting identifiers are associated with each other according to that rule. Finally, the translated rules are stored in a table, we refer to this as the mapped rule index (MRI) table.

The compound identifiers are used as keys to retrieve the disclosure rules. Once a query is issued, the corresponding context identifiers are determined by means of a unique factorization. If the prime numbers or any of their partial products exist in the MRI table, the relevant disclosure rules will be retrieved. This way, rules are stored in a compact way and therefore, the query efficiency will be improved.

Example 2. For the Example 1, if we assume Bob's vehicle id is ''car1234'', the existing contexts and the associated atomic context identifiers for the first rule can be as follows:

<Alice, Role, Paramedic>, cid_1^{app}=5
<Alice, Location, Melbourne>, cid_2^{app}=11

, and

<car1234,Location, Melbourne>, cid_1^{data}=7
<car1234, Incident, yes>, cid_2^{data}=11.

Likewise, for the second rule, we will have:

<Alice, Employee, 13cab>, cid_3^{app}=19

, and[2]

<car1234, Location, Footscray>, cid_3^{data}=13
<car1234, Time, Night>, cid_4^{data}=17.

The above rules are stored in MRI Table. Now assume the contextual identifiers of an issued query is ($CID^{app} = 429, CID^{data} = 221$). Since prime factorization of $429 = 3 \times 11 \times 13$, one can infer that all the contextual information related to this application is also relevant to the compound identifier 33 ($= 3 \times 11$), 143 ($= 11 \times 13$), and 39 ($= 3 \times 13$).

The scalability of our rule-indexing approach depends on the number of prime numbers. Although there are infinitely many prime numbers, the number of known primes less than 10^{23} is evaluated to be $\approx 1.9 \times 10^{21}$ [20]. Additionally, in our scheme, the context identifiers of application and data are independent, so the practical upper bound of our indexing scheme is twice i.e. $\approx 3.8 \times 10^{21}$. We assume, this number is enough for indexing contexts existing in disclosure rules of the proposed framework. In any practical implementation, the number of rules will be considerably less than this.

In this work,we assumed that context is acquired either directly from users or devices they own and trust. If this assumption is restrictive, a separate secure channel such as SSL/TLS (in addition to the data channel) can be used to obtain context as secure side-channel information.

2. The second rule of the Example 1 has <Alice, Location, Melbourne> that is already assigned context id of 11, i.e . cid_2^{app}=11.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

Alternatively, for IoT devices with stringent memory and computation constraints, a lightweight stream cipher such as Self-Shrinking Generator (SSG) [21] or Trivium [22] could be employed. Next, we review the main components of our architecture.

3.3 Security Service

In our framework, the dynamic obfuscation is done by a Security Service that follows the principle of Security-as-a-Service model [23]. However, our goal is to achieve *content* security as opposed to *transport* security to govern disclosure control. We have logically decoupled several tasks of this service into three main components of Application Context Engine, Data Context Engine, and Disclosure Decision Point.

Application Context Engine

The main task of the Application Context Engine (ACE) is finding contextual information of the incoming data request. Upon receiving a query, ACE, first authenticates the data requester. If the authentication is successful, then the next step is forming the context set i.e. CS^{app} and CS^{data}. The ACE delegates the formation of the latter set to the Data Context Engine and is itself responsible for forming the CS^{app}.

As already stated, we assume the contextual information for the application is either stated in the query (contextual query), or extracted by the ACE. For instance, the provided credential could reveal the role of data requester or the IP address can be used to find the requester location. Apart from that some of the contextual information such as time might need to be translated to other high level contexts (such as "working hours" or "night") and therefore the application context engine should interpret this context. The details of such interpretations are out of scope of this work.

Data Context Engine

The Data Context Engine (DCE) has two main tasks: forming the context set for the queried data (CS^{data}), and obfuscating the data before its dissemination to the data requester.

For both tasks, the DCE might need to decode some information using the relevant privacy preservation function, being the multi-stage privacy protection that we achieved by applying k different functions to protect sensitive information. For instance, if a physician wants to have access to heartbeat data of patients who have a family history with cardiovascular disease, this information might be stored in the patient's health-record that is already encrypted (say by function f_1). Therefore, the DCE needs to reverse the transformation process prior to retrieving contextual information. Additionally, once the granularity rule for the data requester is found, the data itself might be protected in a database, if it has been defined as sensitive data (say by function f_2 for heartbeat data). Therefore, the data is first decoded and then obfuscated prior to its dissemination to the destination. That is why we emphasised the low complexity and reversible privacy requirements of function f.

Disclosure Decision Point

The Disclosure Decision Point (DDP) is where the tables ACI, DCI and MRI have been stored. Once, the CS^{app} and CS^{data} are obtained from application and data context engine, the DDP, translates each individual context to the prime identifier using table ACI for application context and DCI for data context. Then, the compound identifiers CID^{app} and CID^{data} are calculated by multiplying the relevant identifiers. Finally, the MRI table is scanned for a match. If so, the corresponding granularity level (gl) is given to the Data Context Engine that obfuscates the data accordingly, and it is sent back to the data requester. Otherwise, the access disclosure is denied (as we consider denial for non-existing rules to be the default state).

4 Case Study: Smart Vehicles

In this section, we customize our proposed framework for a smart city scenario. Smart cities rely on current advances in technologies, such as IoT, networking, data analytics, recommendations, and decision support, to deliver better quality of life to citizens. The indispensable building blocks towards smart city vision include smart healthcare, smart vehicle, smart grid, etc. Although, we described a general purpose framework in Section 3, we focus, for this case study, on smart vehicle deployment.

Smart vehicles are the key to the revolution that is forging the modern intelligent transportation system. The connected vehicles are already on the market, and it is estimated that, by 2020, 75% of globally shipped cars will be connected to the Internet [24], and this obviously opens up privacy issues. The reason that we choose smart vehicles is that the types of data generated by smart vehicles include spatiotemporal streams (i.e. trajectory data) that are changing frequently, and therefore context-driven privacy preservation is inherently a difficult task. More specifically, there are types of smart vehicles in terms of Internet-connectivity namely *brought-in* or *built-in* connectivity. The former option caters connectivity through a Smartphone of the driver/passenger, whereas the latter option relies on cellular service in the on-board infotainment system. Since the built-in smart vehicles have more stringent requirements on latency and reliability for control or monitoring purposes [25], a lightweight privacy protection is preferred.

4.1 System Overview

Our smart vehicle system is shown in Fig. 2. At the bottom of this figure, there are smart vehicles that with the aid of sensors transmit their data to a cloud storage periodically. We have considered five main services/applications in our system. These include Paramedic service; Road Safety service; Parking Locator; Fuel Station locator; and a Diagnostic Health service. The Paramedic service is an emergency service that is available to smart vehicles in the case of accidents. The Road Safety service provides traffic information such as road congestion, recommending paths and also driving offenses. The Parking Locator and Fuel Station Locator are essentially location-based services for finding available parking spaces and nearby fuel stations, respectively. Finally the Diagnostic Health service gathers information about people who might be in the presence of contagious diseases. In the subsection 4.3, we explain how this could be related to the trajectory data.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

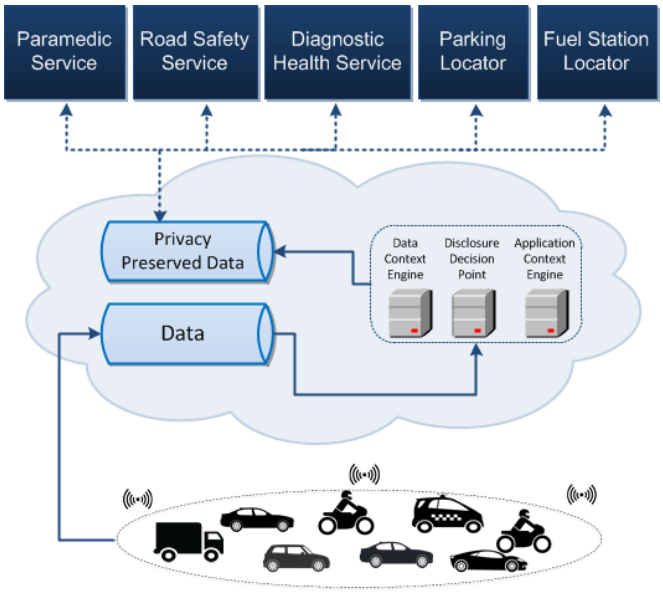

Fig. 2. An overview of a smart vehicle system.

4.2 Data Model

A smart vehicle is a moving object that is equipped with one or more sensors. In our data model, streaming trajectories obtained by these sensors are treated as sensitive information that we want to protect. For a moving object MO, we represent a trajectory data stream Tr_{MO} as a sequence of pairs $Tr_{MO} = \{< p_1, t_1 >, < p_2, t_2 >, ..., < p_{now}, t_{now} >\}$, where position p_i is a Cartesian point coordinates shown as x_i and y_i with ordered timestamps t_i. The (x_i, y_i) values could be easily obtained by mapping GPS coordinates i.e. longitude and latitude using a Universal Transverse Mercator (UTM) transformation.

Apart from the trajectory stream, every individual data point is enriched with a set of contextual information that we denoted by CS^{data}. For the particular dataset that we used for the implementation, the context set includes vehicle id (vid), and current speed sp_i. Therefore, the information for the object MO at time t_i includes $(< p_i, t_i >, vid, sp_i)$. As the dynamic data obfuscation depends on contextual information of queried data and data requester, we also need to define the application context (CS^{app}) for our system as described in the following sections.

4.3 Spatiotemporal Granularities and Disclosure Rules

As stated in Section 3.2, there are different ways to define granularities. Without loss of generality, for spatial granularity ($gl^{spatial}$), we consider location precision (in terms of kilometers, meters, etc) and for the temporal granularity ($gl^{temporal}$), a binary granularity is assumed, indicating whether the time information should be revealed or not.

To make it clear, let us review some of the discourse rules for different services. If a vehicle is involved in an incident, the precise location and time is shared with the Paramedic services. The Diagnostic Health service only has access to the spatial data with granularity of 1 meter, (i.e., $gl^{spatial} = 1m$ and $gl^{temporal}$=0). This disclosure rule for our case study has been motivated by a scenario where a driver is suspected for a super-contagious disease such as (say) Ebola, and therefore the Diagnostic Health needs to know the places where the driver has visited in order to track the possible spread of the disease. However, the order of visiting places may not matter to this service and thus they do not need to be revealed. For a general scenario, having d granularity levels, we set the $(d-1)$ least significant bits of location coordinates to zero for the worst case (applications with least discourse granularity) in order to mask the precise coordinates.

It is important to note that for a particular service, the granularity could change based on the contextual information. For instance, the Road Safety service might be authorised to have access to the location information with ($gl^{spatial} = 100feet$ and $gl^{temporal}$=0), but if the vehicle is travelling beyond the street limit, the exact location and time might be revealed to this service.

4.4 Privacy Protection at a Glance

We consider a trajectory stream as sensitive data the privacy of which needs to be protected. Therefore, there are two main data attributes, spatial and temporal. In this regard, our system resembles the privacy protection against location-based services that shall be reviewed in Section 4.5.3. The majority of those proposed methods, only focused on location privacy and did not consider the importance of revealing timeliness of spatial information that could result in breach of privacy [26].

For our system, we propose a two-tier privacy preservation (k=2), one for privacy preservation of spatial data at the data collection stage, and the other for temporal data that is applied at data storage stage. Therefore, we achieve spatiotemporal privacy while data is at rest. Additionally we use an obfuscation function f^{Ob} that obfuscates data according to the desired granularity level at the time of data dissemination. Fig. 3 illustrates an example of our spatiotemporal privacy preservation approach, where the original trajectory of a moving object is not only replaced with the cloaked locations, but also the sequence of these locations are perturbed.

In order to respect the low complexity of IoT devices, both suggested privacy preserving functions f_1 and f_2, and also the f^{Ob} function are lightweight, while we have tried to make them as secure as possible, given IoT device constraints. For this purpose we make use of digital watermarking methods and pseudo-random constructions because of their hardware-friendly nature. In fact, f_1, f_2, and f^{Ob} are respectively: pseudonoise addition; (optionally hashed-based) scrambling; and data masking (which can be coupled with one-time pad partial encryption, if a more secure transmission is needed).

4.5 Preliminaries

In this section, we briefly introduce some preliminaries that are necessary to follow the rest of Section 4.

4.5.1 *Digital Watermarking*

Digital watermarking is a proven technique usually in multimedia domain for copyright protection. The watermark constitutes a piece of secret information to be hidden within the digital content in such a way that it is not visible to the consumer. This requirement is called watermark *invisibility*. A digital watermark can be either distortion-based or distortion-free depending on whether the embedded

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

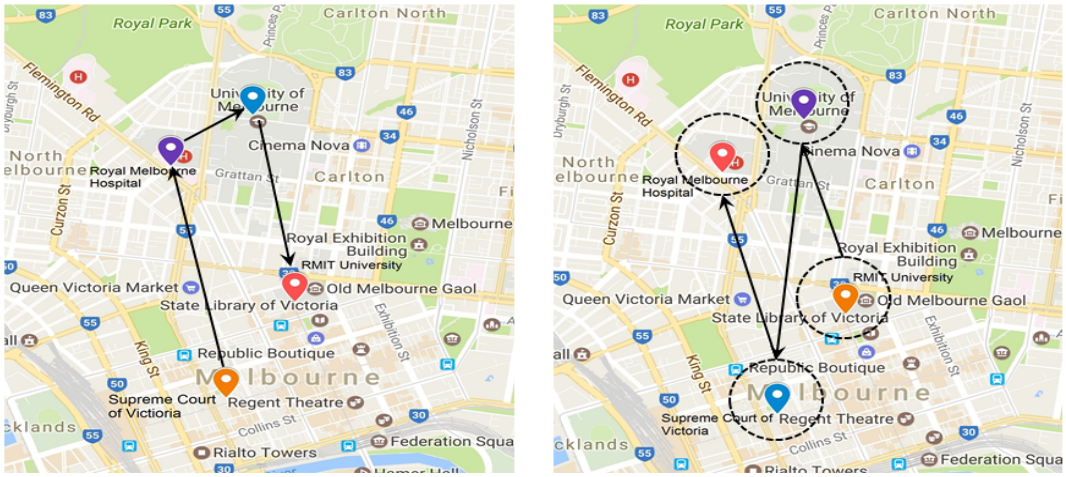

Fig. 3. Spatiotemporal privacy preservation. (a) Original trajectory: Supreme Court of Victoria → Royal Melbourne Hospital → University of Melbourne → RMIT University, (b) modified trajectory: *cloaked*(RMIT University) → *cloaked*(University of Melbourne) → *cloaked*(Supreme Court of Victoria) → *cloaked*(Royal Melbourne Hospital).

marks introduce any distortion to the underlying data [27]. For example, adding random numbers to data samples results in changing the original values (distortion-based watermarks), whereas re-arranging data samples according to a secret watermark do not introduce any change into data values (distortion-free watermarks).

One of the recent driving forces in digital watermarking research is data obfuscation [28]. Because embedding watermarks introduces a tunable distortion in host data, it is possible to mask the original data for the applications where privacy is of great concern. Contrary to conventional watermarking, the visibility constraint can be relaxed and the reversible distortion introduced by the watermark is used to reduce data precision to below levels where privacy can be compromised. These levels are tunable to application-dependent granularity.

In this work, we have used a distortion-based watermark (noise addition) and a distortion-free watermark (data scrambling). For the former, we take advantage of so called *Linear feedback shift registers* to construct the noise/watermark that we describe briefly in the following.

4.5.2 Linear Feedback Shift Registers

Generating random numbers has been studied thoroughly for many applications such as stream cipher design, watermarking codes, spread spectrum communications [29]. Unfortunately, generating truly random numbers (TRNs) is an expensive task due the complexity of required hardware (such as thermal noise of zener diodes or radioactive decay). In this regard, Linear Feedback Shift Registers (LFSRs) are favourite primitives due to their desirable statistical properties and hardware-friendly nature [29].

LFSRs are shift registers, generating new bits using a linear feedback polynomial. Fig. 4 shows an example of an LFSR of order 4. In this example, a new bit is generated for each shift, based on a linear combination of the bit values of (in this case 3 and 4) previous shifts. Certain feedback combinations produce a pseudorandom pattern of bits equal to 2^{order}. So in the case of Fig. 4, a pattern of maximal period of 15 is produced (16-1 as the all-zeros case is excluded). For this reason, the produced sequence is called a *maximal-* or m-sequence.

A LFSR can be specified by means of its *characteristic polynomial* or the positions of the *taps*. For Fig. 4, the characteristic polynomial is $g(x) = x^4+x^3+1$. Once $g(x)$ and non-zero initial value of registers (1101 for Fig. 4) are known, the rest of the sequence is uniquely identified. In fact, for our usage, the deterministic behaviour is desirable because of the reversibility requirement of the privacy preserving operation (function f). On the other hand, the security requirement of the privacy functions calls for unpredictability of PN sequences that is measured by *linear complexity*.

Linear Complexity (LC) is the length of the shortest LFSR that is able to generate a given sequence. An ideal binary PN sequence of length p, is one whose linear complexity is also p. In other words, you need the entire sequence in order to predict future bit values. For very long sequences, this is impractical, and thus is sufficiently secure. Unfortunately, the LC of an LFSR itself is poor ($\log_2 p$). In the literature, there are many proposals to generate PN sequences with higher LC such as: adding a source of truly random number generator; combining multiple LFSR; or decimating generators irregularly. For this work, we use a method called Dynamic Linear Feedback Shift (DLFSR) that achieve high LC by frequently changing initial values and characteristic polynomial. Because of the space limit, we cannot provide the details of this construction, but we briefly explain the logic of Dynamic LFSR (DLFSR) in the following. Readers are referred to [30] for this particular construction.

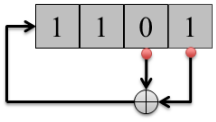

Fig. 4. An LFSR of order 4 with characteristic polynomial $x^4 + x^3 + 1$.

4.5.3 Work Related to Spatiotemporal Obfuscation

Because our general privacy preservation framework is studied for trajectory data streams, related works to this

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

domain is discussed here as well. Our proposed solution for the envisioned smart vehicle system is semantically coherent with the privacy protection against location-based services (LBSs) like Google Maps or Foursquare.

A line of research for preserving privacy when the identity of the users is not relevant for the provision of the LBS, adopts the K-anonymity principle [31] to transform a point location into a cloaked region such that the user is indistinguishable from other $(K-1)$ users. Another example is a user-anonymizer-LBS named Casper [32] that maintains the locations of clients using a pyramid data structure, similar to a Quad-tree. Let user u ask a query and c be the lowest-level cell of the Quad-tree where u resides. If c contains enough users (i.e. $|c| \geqslant K$) , c becomes a cloaking region; otherwise, the horizontal c_h and vertical c_v neighbors of c are retrieved. If $|c \cup c_h| \geqslant K$ or $|c \cup c_v| \geqslant K$, the corresponding union of cells becomes the cloaking region; otherwise, the anonymizer retrieves the parent of c and repeats this process recursively. A more generalized approach is Hilbert Cloak [33] based on a Hilbert space filling curve that guarantees privacy for any distribution of user locations.

Another line of research closest to ours is authenticated LBSs, whereby a user cannot remain anonymous but still wishes to retain location privacy. In this regard, Duckham and Kulik proposed a few rules as the key principles of research on location privacy, which make it different from other privacy preserving research [34]. An example of these rules is that, privacy against LBSs cannot be protected by anonymity or replacing the real user identity with a pseudonym one because anonymity presents a barrier to authentication and personalisation. Therefore, other location-specific information such as predictable mobility of humans and the constraints of the area within which people move, should be taken into account.

In [35] various location privacy techniques including noise addition and spatial cloaking are studied. One important founding of this research is that inference attacks can be made difficult by artificially introducing a large amount of noise to the sensor locations. A dummy-based location privacy approach for mobile service is proposed by [36], in which fake queries are deliberately generated according to either a virtual grid or circle to cover a user's actual location. Hiding sensitive locations [37] is another alternative. In [38], a comprehensive framework for balancing the individuals needs for high-quality information services and the location privacy is presented. For this purpose, the author proposed to degrade the quality of the location information and provide obfuscation features by adding n points at the same probability to the real user position.

The majority of the above methods only focus on location privacy. However, revealing timeliness of spatial information, opens up the possibility of time-and-location attack [26] and results in breach of privacy. To the best of our knowledge, there are only a few privacy preservation techniques that do not disregard the temporal cloaking of trajectory data. An example of combining spatial and temporal cloaking can be found in [26].

Recently, there are some efforts that adapt traditional privacy enhancing techniques with strong privacy guarantee such as *private information retrieval* and *differential privacy* for LBS applications. Methods based on the former scheme allow a user to retrieve data from a database, without disclosing the index of the data to be retrieved to the server. For instance, if a sever S holds a database with n bits, say $X = (X_1, ..., X_n)$, user U should be able to retrieve the value of X_i, without disclosing the value of i to S. However, due to computational complexity, it is unclear at present if these approaches can be applied in a real LBS setting [39]. The latter scheme aims to extend the standard differential privacy for spatial data streams. In particular, a new generalised notion of privacy, *Geo-Indistinguishability* is proposed [40]. Intuitively, a mechanism provides Geo-Indistinguishability, if two locations that are geographically close have similar probabilities to generate a certain reported location. Therefore, the neighbouring concept in the standard differential privacy can be measured by the Euclidean distance metric. In Section 5 we implement this method to see whether it is suitable for practical use.

To the best of our knowledge, the only work that directly uses watermarking for obfuscation of trajectory data is recently presented in [41]. The proposed technique guaranties preservation of hierarchical clustering operations after watermark insertion. This method is suitable for watermarking of trajectory datasets but cannot be used for on-the-fly watermarking of a trajectory stream.

4.6 Privacy Preserving Data Collection-Spatial Cloaking

The spatial privacy in our work is achieved by adding PN values (watermarks) to the location coordinates at the sensors. First, we explain how these watermarks are generated.

Watermark Generation using DLFSR

We use the binary DLFSR, proposed by [30], that consists of two LFSRs (primary and secondary) and a counter. The primary LFSR is controlled by the counter, whose value depends on the internal state of the secondary LFSR and therefore the primary LFSR polynomial is changed in a round robin fashion. In other words, there is a secondary LFSR that cloaks regularly combined with a primary LFSR with irregular cloaking. This results in a source of PN generator with higher period and linear complexity compared to the simple LFSR.

The above DLFSR needs three values for both LFSRs in order to generate random numbers: number of LFSR stages (LFSR order), initial value of the register, and the initial selected polynomial. We denote these values by $(l_1, iv_1, poly_1)$ and $(l_2, iv_2, irrpoly_2)$ for the primary and secondary LFSRs, respectively. Note that the examples explained below assume a bipolar LFSR (i.e. alphabet of 2), for simplicity, but we can also consider LFSR's using an alphabet, A, which is a prime number, and balance it by subtracting floor $(A/2)$. The DLFSR construction works identically provided both LFSRs use the same alphabet. The sequence values are then uniformly distributed from $-A/2$ to $+A/2$.

Watermark Insertion

Assume the i^{th} generated random number obtained from the DLFSR is r_i. The watermark is then a scaled version of the r_i that is added to the location coordinates to make it inaccurate. For the location $p_i = (x_i, y_i)$, the watermarked location $p_i^{'}$ is calculated as follows:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

$p_i^{'} = (x_i^{'}, y_i^{'}) = (x_i + \alpha_x \times r_i, y_i + \alpha_y \times r_i)$

where α_x and α_y denote the obfuscation scaling factors, or watermark amplitudes.

Clearly, the higher the scale factors, the better spatial privacy can be achieved as the accuracy of the data decreases. This way, trajectories can be hidden on the fly at the point of origin. The Security Service needs to be able to obtain the original values if necessary. Therefore, for every moving object MO, 8 secret parameters need to be exchanged *a-priori* with the Security Service including $\{(l_1, iv_1, poly_1), (l_2, iv_2, poly_2), (\alpha_x, \alpha_y)\}$. This key exchange can be effectively done using pairing-free key encapsulation methods such as pCLSC-TKEM [42].

It is important to note that the lightweight operations of spatial obfuscation is mostly related to the use of DLFSR generators [30] that are tested by the authors on RFID tags of types EPC Gen2 and Gen3 and proved to be optimal to be implemented on computationally bounded chips. Therefore, we can substantially use this performance guarantee for our private data collection method.

4.7 Privacy Preserving Data Storage-Temporal Cloaking

We achieve temporal privacy by scrambling the obfuscated locations. Note that the temporal cloaking does not change the timestamps. Instead, the coordinates are shuffled positionally to obscure the timeliness of location information. For instance, if vehicle "car1234" has been at University of Melbourne at time 5pm, and then RMIT University at time 8pm, we swap the locations, but we do not change the time values themselves. At the time of data dissemination, the Security Service decides whether the correct order of trajectories needs to be revealed to the application.

In contrast to the spatial cloaking, the temporal cloaking is done by the Security Service. This means we have more freedom to choose a secure privacy preservation method (compared to the LFSR) as long as the time complexity is not high. For this purpose, we define a secure permutation $\pi : [1..b] \rightarrow [1..b]$ to scramble data in a particular order such that the i^{th} watermarked point $p_i^{'}$ is substituted by $p_{\pi(i)}^{'}$ and b is the required buffer size (that is a power of 2). For example, a trajectory of length 4, $\left\{< p_1^{'}, t_1 >, < p_2^{'}, t_2 >, < p_3^{'}, t_3 >, < p_4^{'}, t_4 >\right\}$, is replaced with $\left\{< p_4^{'}, t_1 >, < p_1^{'}, t_2 >, < p_2^{'}, t_3 >, < p_3^{'}, t_4 >\right\}$ for a certain permutation π.

There are many different ways for having secure permutation such as using block cipher, hash functions, congruential random numbers. Here, we have chosen a simple yet effective secure scrambling method based on hash functions that we describe bellow:

Assume $Hash()$ is a one-way keyed hash function with the output size of l bits and the buffer size is $b = 2^c$. We apply a non-overlapping window of size c to the hash output and encode it to a decimal value. We then advance the window by c values. This leads to l/c numbers that correspond to the permutation indexes. If some of the numbers collide, we skip the duplicate values. The beauty of this scheme is that hashed values can be customized for individual moving objects using a secret key or its combination with other contextual information such as vehicle id. The security of this permutation lies in the security of the hash function and the buffer size b. The longer the buffer, the number of possible permutations will be higher and a brute force attack will be less effective. On the contrary, a very long buffer results in delays to the reverse process and decrease the system efficiency in terms of query response time.

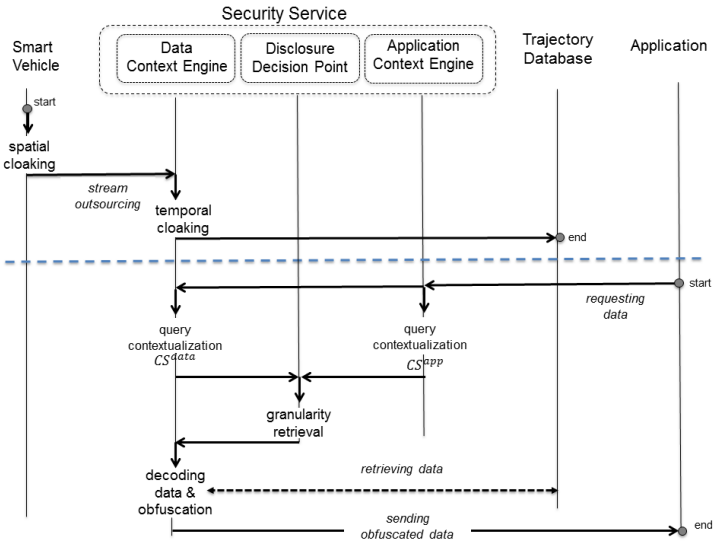

Fig. 5. Sequence diagram for spatiotemporal privacy preservation. The dashed line separates the data collection and storage stages from data dissemination (As explained in Section 3.3, the three components of the Security Service are normally part of one service and are only de-coupled here for clarification.)

By coupling temporal scrambling with spatial cloaking, we ensure data is protected at the rest. If the complexity of the $Hash()$ is still too much for the IoT device, a simpler stream cipher such as SSG or Trivium, could be employed. While potentially less secure, they have the advantage of using LFSR's for the generation. We would expect in the near future for all IoT devices to be capable of secure connections, so the above are only interim measures.

4.8 Privacy Preserving Data Dissemination-Dynamic Obfuscation

The data dissemination stage is triggered by receiving a query. At this point, we assume the query is contextualized and therefore the corresponding granularities i.e. $gl^{spatial}$ and $gl^{temporal}$ are retrieved from the DDP. Before performing data obfuscation, the Security Service first needs to obtain the exact location and time and then based on the $gl^{spatial}$, the least significant of the spatial points are masked. If the query wants the trajectory of an object for a time duration such as range queries (example is between 8 am-12am), the Security Service decides whether the time information of the trajectories should be revealed based on the $gl^{temporal}$ value.

For obtaining original information, the Security Service uses the secret values to re-generate the random numbers for reversing scrambling and noise addition. For the former, the queried data point needs to be retrieved from the permuted index $p_{\pi(i)}^{'}$. And then, for the latter, the watermark values ($\alpha_x \times r_i$ and $\alpha_y \times r_i$) needs to be subtracted from $p_{\pi(i)}^{'}$. The last step is obfuscating the original coordinates by, for example, masking the least significant bits.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

Fig. 6. Number of vehicles per time interval for the TAPASCologne dataset [43] over 24 hours (The x-axis shows the time and the y-axis shows the number of vehicles).

For implementation, we consider, maximum precision of 1 meter that is well below the precision of GPS coordinates, and at the same time sufficient for many applications. We then, consider 4 different granularity levels, such that for the lowest level, the three LSBs are set to zero, whereas for the highest level no bit will be masked. Geocentrically, the obfuscated location, $f^{Ob}(p_i)$ can be represented by a circle that with p_i at center, and $2^{gl^{spatial}}$ is the radius in meters. The more obfuscated bits, the larger the circle area will be, the better privacy can be achieved. The privacy preservation technique that we described is illustrated in Fig. 5.

We mentioned that Security Service mainly deals with protection of data content as opposed to the data transmission. If the data transmission is also required to be protected, we recommend using a partial encoding such as one-time pad with a user session key to protect the unmasked bits (i.e. most significant bits) of the location data to enforce transmission security as well. This way, there is no point form point of origin to the final destination, where data is left unprotected.

5 Performance Evaluation

We developed the described smart vehicle system as a proof of concept. For this purpose, the large-scale TAPASCologne dataset [43] is used, which contains trajectories of the car traffic in a city of Cologne, Germany. The trajectories covers a region of 400 square kilometers for a period of 24 hours. The dataset comprises more than 700,000 individual car trips. Each record of TAPASCologne dataset contains the time (with 1-second granularity), the vehicle identifier, its position and speed. Fig. 6 shows the distribution of vehicles over the 24 hours. Also, we have defined 5000 disclosure policies expressed based on SWRL (see Section 3.2.2). All the experiments are performed on a DELL workstation with an Intel i7-4790 3.26GHz CPU and 8GB RAM.

The performance of our system is investigated in terms of processing time for retrieving trajectories. For our system, this includes the time that is required for reversing the privacy transformations (permutation and watermark extraction) and dynamic obfuscation (data masking). For simulating queries, we divided the 24 hours into time intervals of size 7.5 minutes, and retrieved the trajectory of vehicles that are present in that particular interval for different services that we named in Section 4.1.

Our method is compared against the following systems:

1) *Two crypto-systems*. Instead of spatiotemporal cloaking, in these systems, GPS coordinates are encrypted to measure the security overhead of data protection. Therefore, every time data leaves the system, it is decrypted in order to retrieve original values and then masked with the desired granularity level completing the dynamic obfuscation goal. Two block cipher schemes are chosen: a conventional block cipher, 64-bit DES and a lightweight block cipher, 128-bit CLEFIA (developed by Sony Corporation in 2007) [44]. The latter is a proposal under consideration in ISO/IEC 29192-2 and it has the best overall performance compared to other comparable lightweight encryption methods [22].
2) *A differentially private system*: Geo-Indistinguishability [40] is the extension of differential privacy for spatial data stream by adding noise drawn from a 2D Laplacian distribution (refer to Section 4.5.3). For efficiency, this is done in polar coordinates by selecting an angle uniformly and a radius from a Gamma distribution. Because, the noise addition needs to be done on-the-fly, a window is defined by which the neighbouring points are only considered within the scope of the window. This means, an adversary is allowed to distinguish locations which are far away from each others. This results in a $\frac{\epsilon}{w}$ privacy guarantee, where ϵ and w are privacy level (that quantifies the privacy risk) and window size, respectively[3].

The results of the two crypto-systems against ours are shown in Fig. 7. One interesting observation is that the processing time of all systems follows the data distribution behavior. For instance, during the peak hours (such as morning and afternoon), the processing time of retrieving trajectories for all 3 systems is higher which is clearly because there are more cars in those intervals. The other observation is that the processing time of the more lightweight CLEFIA is much better than DES, however still far from our system (roughly speaking our system is 10 times better than DES,

3. The original definition is for ϵ-differential privacy, where ϵ is a difference between the probability of receiving the same outcome on two different datasets.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

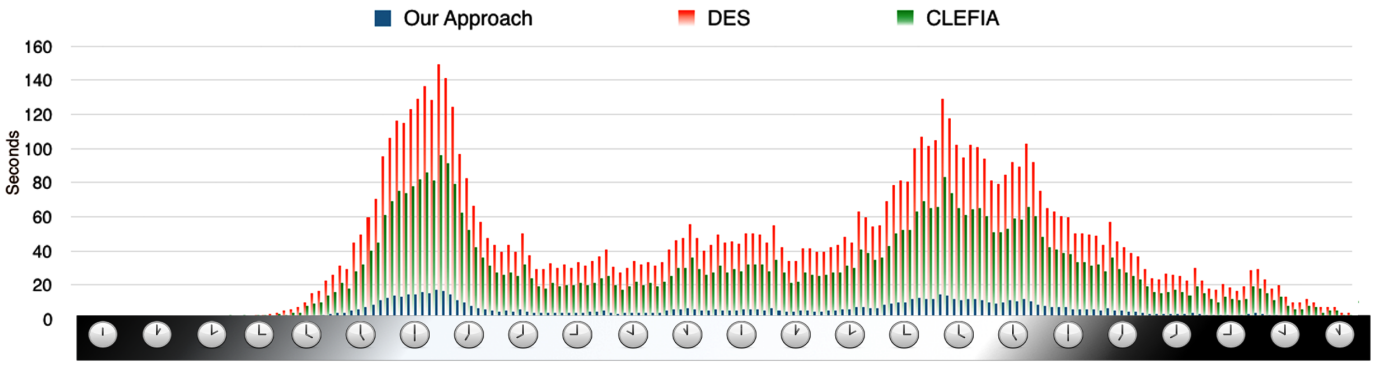

Fig. 7. Processing time of our approach (using 16-bit DLFSR and 160-bit SHA permutation) against the 64-bit DES and 128-bit CLEFIA crypto-systems.

and 4 times better than CLEFIA). Apart from processing time, the crypto-systems are inferior to ours in a sense that the data cannot be protected by encryption methods at the sensor level, due to high complexity of the operations and therefore the data is protected only at the database. Hence, we can conclude that our system outperforms the encryption counterparts in terms of both response time and data protection at sensor level and database as well.

For the Geo-Indistingishibility, the performance and privacy guarantee vary by window size. Therefore, we chose 15 window sizes (from 16 to 240 trajectory samples per window) and only report the mean processing time for resolving queries. The results are given in Fig. 8. The horizontal lines indicate where the performance of our system and CLEFIA, approach Geo-Indistingishibility. It can be seen that Geo-Indistingishibility is comparable with our system for window size of about 16, and CLEFIA for window size of 112. Further discussion shall be given in Section 5.1.

To better understand the security overhead related to different parts of our system, we breakdown the running time that is taken by the Security Service for query contextualisation, reversing temporal cloaking (permutation), reversing the spatial cloaking (watermark insertion), and dynamic obfuscation (data masking) in Fig. 9. According to this figure, the most time consuming part is related to permutation and query contextualisation. In particular, permutation and contextualisation contribute on average 55% and 30% of the overall running time. This is still an acceptable cost for having multi-granular obfuscation. Another important point is that, not all components are present for all accesses. For example, the security overhead of the Paramedic Service is much less compared to other services as the time order does not matter to this service and therefore the total running time does not include the permutation component (saving 55% in this case).

5.1 Discussion

We argued that, from a thing-oriented perspective, data protection at dissemination stage is insufficient to preserve privacy of individuals. In order to push protection to earlier stages, digital watermarking is used as an intriguing alternative for highly complex privacy preservation techniques. At the time of data dissemination, the obfuscator masks the sensitive data based on contextual information. The above results confirmed that the cost of our multi-stage data protection is reasonable for the implemented smart vehicle system. In particular, our approach is more efficient and scalable than crypto and differentially-private alternatives. In regards to the crypto-systems, we applied a lightweight encryption scheme on IoT data (CLEFIA), but the performance is still not desirable in terms of query response time.

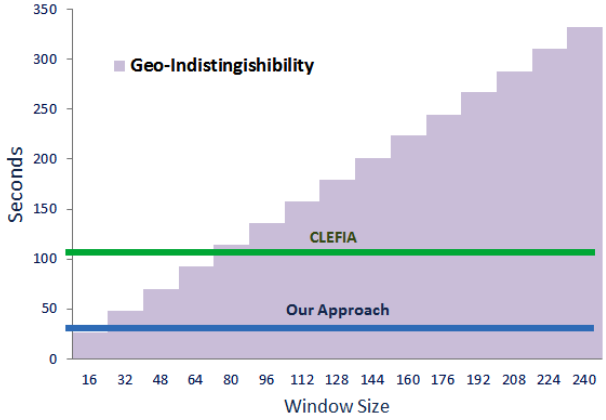

Fig. 8. Mean processing time of Geo-Indistingishibility against our approach and 128-bit CLEFIA.

In related to the differentially-private implementation, first of all the security of data from source is not considered. This results in formation of weak links that makes the privacy guarantee questionable. Second, the scheme calculates all possible combinations between neighbouring data points to find the amount of noise to be added within a window. Therefore, each sample is $\frac{\epsilon}{w}$ private. For a fixed amount of ϵ, the larger window size, the stronger the privacy guarantee will be. According to Fig. 8, Geo-Indistinguishability is compelling with ours with window size of 16. This number is very small and results in weak privacy guarantee. That is to say, there is a strong privacy guarantee (only at the storage stage), but with the cost of expensive calculations for a large window size. Apart from that, the temporal correlation between the neighbour points (temporal privacy) is not considered which opens up the possibility of time-and-location correlation attack [26] against this implementation.

From the security standpoint, two parts influence our privacy preservation, spatial and temporal cloaking. Firstly, the security of spatial cloaking is based on the linear complexity of the DLFSR generator. For the implementation, we used a DLFSR with polynomials of the order 16 and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TBDATA.2017.2737463, IEEE Transactions on Big Data

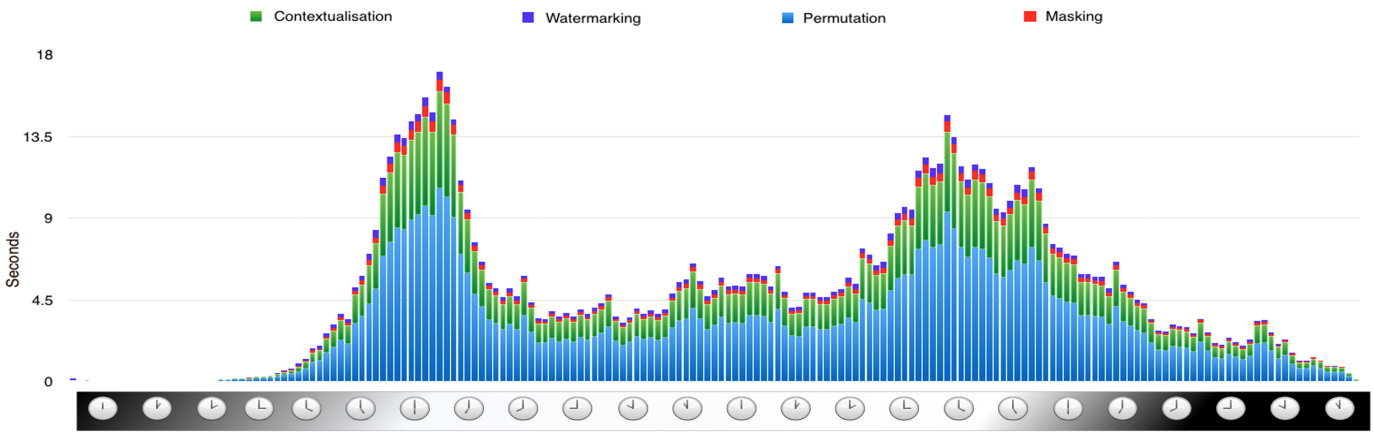

Fig. 9. Running time breakdowns for different parts of our system including query contextualization, finding permuted indexes, extracting watermarks, and dynamic data masking.

4, that generates random numbers with period and linear complexity of $p = 7864200$ and $LC = 1920$ [30]. This increases the linear complexity by a factor of almost 120 compared to a simple LFSR –and this is achieved without losing the good statistical properties of the LFSR[4]. It is also possible to use higher order LFSRs for the primary and secondary polynomials to further increase the security. Because the linear complexity of this DLFSR generator is not mathematically investigated for a general case by the authors [30], we have chosen this length to be able to compare it with the basic LFSR. Apart from the random numbers, the scale factors are part of the secret information that is only known to the data owner and Security Service.

Secondly, the security of temporal cloaking is related to the security of the permutation. For this purpose, we used a one-way keyed hash function to scramble watermarked data points. A good hash function $Hash()$ must have the two properties:

1) One way transformation: Given a hash value h it should be difficult to find any message m such that $h = Hash(m)$;
2) Collision resistance: It should be difficult to find two different messages m_1 and m_2 such that $Hash(m_1) = Hash(m_2)$. Such a pair is called a hash collision, and should be rare or absent in a good has function.

Due to the one-way property of hash functions, even if given $h = Hash(M, K)$ in the temporal cloaking, the attacker is unable to obtain the secret key K from the hash value. In addition, hash functions are often used in the generation of pseudorandom bits (e.g., NIST special publication [45]). The randomness of hash function guarantee the permutations for scramble temporal cloaking is unpredictable without knowledge of the secret key K. Also, the uniformity of the distribution of hash values guarantees that all valid permutations can be generated on the basis of hash values.

In terms of privacy guarantee, a transport layer connection such as TLS can be used to maintain privacy from IoT device to data storage. For devices incapable of handling the complexity of TLS, we offer a lightweight alternative, while not a guarantee, may be sufficient for the application (Please refer to Section 3.2.2, last paragraph). We remind that our obfuscation framework extends that privacy *beyond* the storage to end-users. Even though, strong privacy from data storage onwards to the end-users can be achieved by a differential privacy approach, the high complexity of the calculations, prohibits its utilization for IoT applications in which data are usually unbounded, transient, and require query results to be updated when new data arrive [2].

4. The period and linear complexity of an LFSR with order 16, is 65535 and $\log_2 2^{16} = 16$, respectively as described in Section 4.5.

6 Conclusion and Future Work

In this paper, we took a leap forward to proactively discuss the possibility of breaching privacy at the early stages of IoT data collection which is more likely to happen in the future of smart city developments. Although, privacy enhancing approaches have already been proposed for some specific applications, a comprehensive framework, general and flexible enough to deal with IoT constrained environments in a real setting, is still missing. In our novel framework, data is not only protected before leaving its device, but also it is protected using other security methods at different points of the entire system to afford maximum data protection. Additionally, our framework supports flexible data granularities for obfuscating sensitive information.

The proposed framework was implemented for a smart vehicle system, whereby the spatiotemporal data stream is protected by means of digital watermarking and data scrambling. In order to achieve dynamic data obfuscation, contextual information was integrated with disclosure rules using our rule indexing scheme. Having more contextual information, one can obtain finer discourse granularity. However, taking too much context into account for disclosure control is not a prudent decision, since the complexity of the system increases. There is a middle ground here, but it is very dependent on individual application and performance limits. The results confirmed that the computational complexity of our system is very modest and outperforms the crypto and differentially-private solutions. Our privacy preserving method mostly depends on secure pseudorandom generation. However, being a framework, one can substitute other

privacy-preserving algorithms which can then be combined to build more complex solutions to meet the security requirements, while respecting the IoT characteristics.

REFERENCES

[1] S. Hodges, S. Taylor, N. Villar, J. Scott, D. Bial, and P. T. Fischer, ''Prototyping connected devices for the internet of things,'' *Computer*, vol. 46, no. 2, pp. 26--34, 2013.

[2] E. Bertino, ''Big data-security and privacy,'' in *IEEE Inter. Congress on Big Data*, pp. 757--761, IEEE, 2015.

[3] H. Li, X. Lin, H. Yang, X. Liang, R. Lu, and X. Shen, ''Eppdr: an efficient privacy-preserving demand response scheme with adaptive key evolution in smart grid,'' *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2053--2064, 2014.

[4] D. G. E. Bertino, K.K.R Choo and S. Nepal, ''Internet of things (iot): Smart and secure service delivery,'' *ACM Transactions on Internet Technology (TOIT)*, vol. 16, no. 4, p. 22, 2016.

[5] e. a. David E. Bakken, ''Data obfuscation: Anonymity and desensitization of usable data sets,'' *IEEE Security and Privacy*, vol. 2, no. 6, pp. 34--41, 2004.

[6] L. Li, R. Lu, K.-K. R. Choo, A. Datta, and J. Shao, ''Privacy-preserving-outsourced association rule mining on vertically partitioned databases,'' *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 8, pp. 1847--1861, 2016.

[7] 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.

[8] W. B. G. G. Grispos and K.-K. R. Choo, ''Medical cyber-physical systems development: A forensics-driven approach,'' *In Proc. of IEEE/ACM Conf. on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, 2017.

[9] E. Bertino, ''Data security and privacy: Concepts, approaches, and research directions,'' in *Computer Software and Applications Conference, 2016 IEEE 40th Annual*, vol. 1, pp. 400--407, IEEE, 2016.

[10] A. Yavari, P. P. Jayaraman, D. Georgakopoulos, and S. Nepal, ''Contaas: An approach to internet-scale contextualisation for developing efficient internet of things applications,'' in *Proc. of the 50th Hawaii Inter. Conf. on System Sciences*, 2017.

[11] A. Yavari, A. Soltani Panah, D. Georgakopoulos, P. P. Jayaraman, and R. van Schyndel, ''Scalable role-based data disclosure control for the internet of things,'' in *Distributed Computing Systems (ICDCS), IEEE 37th International Conference on*, pp. 2226--2233, IEEE, 2017.

[12] E. Bertino and R. Sandhu, ''Database security-concepts, approaches, and challenges,'' *IEEE Trans. on Dependable and secure computing*, vol. 2, no. 1, pp. 2--19, 2005.

[13] M. J. Covington, W. Long, S. Srinivasan, A. K. Dev, A. Mustaque, and G. D. Abowd, ''Securing context-aware applications using environment roles,'' in *Proceedings of the sixth ACM symposium on Access control models and technologies*, pp. 10--20, ACM, 2001.

[14] J. Hu and A. C. Weaver, ''A dynamic, context-aware security infrastructure for distributed healthcare applications,'' in *Proc. of the first workshop on pervasive privacy security, privacy, and trust*, pp. 1--8, Citeseer, 2004.

[15] E. Bertino, E. Camossi, and M. Bertolotto, ''Multigranular spatio-temporal object models: concepts and research directions,'' in *Inter. Conf. on Object Databases*, pp. 132--148, Springer, 2009.

[16] K. Mivule, ''Utilizing noise addition for data privacy, an overview,'' *arXiv preprint arXiv:1309.3958*, 2013.

[17] C. Gentry, ''Toward basing fully homomorphic encryption on worst-case hardness,'' in *Annual Cryptology Conference*, pp. 116--137, Springer, 2010.

[18] L. Cranor, M. Langheinrich, M. Marchiori, M. Presler-Marshall, and J. Reagle, ''The platform for privacy preferences 1.0 (p3p1. 0) specification,'' *W3C recommendation*, vol. 16, 2002.

[19] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, M. Dean, *et al.*, ''Swrl: A semantic web rule language combining owl and ruleml,'' *W3C Member submission*, vol. 21, p. 79, 2004.

[20] http://primes.utm.edu/howmany.shtml.

[21] A. Fúster-Sabater and P. Caballero-Gil, ''Analysis of the generalized self-shrinking generator,'' *Computers & Mathematics with Applications*, vol. 61, no. 4, pp. 871--880, 2011.

[22] M. Katagi and S. Moriai, ''Lightweight cryptography for the internet of things,'' *Sony Corporation*, pp. 7--10, 2008.

[23] E. Bertino, L. Martino, F. Paci, and A. Squicciarini, *Security for web services and service-oriented architectures*. Springer Science & Business Media, 2009.

[24] J. Greenough, ''The connected car report.'' http://www.businessinsider.com/connected-car-forecasts-top-manufacturers-leading-car-makers-2016-4-29, 2016.

[25] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, ''Connected vehicles: Solutions and challenges,'' *IEEE internet of things journal*, vol. 1, no. 4, pp. 289--299, 2014.

[26] R.-H. Hwang, Y.-L. Hsueh, and H.-W. Chung, ''A novel time-obfuscated algorithm for trajectory privacy protection,'' *IEEE Trans. on Services Computing*, vol. 7, no. 2, pp. 126--139, 2014.

[27] R. Halder, S. Pal, and A. Cortesi, ''Watermarking techniques for relational databases: Survey, classification and comparison.,'' *J. UCS*, vol. 16, no. 21, pp. 3164--3190, 2010.

[28] A. Soltani Panah, R. van Schyndel, T. Sellis, and E. Bertino, ''On the properties of non-media digital watermarking: A review of state of the art techniques,'' *Special Section on Latest Advances and Emerging Application of Data Hiding,IEEE Access*, vol. 4, pp. 2670--2704, 2016.

[29] S. W. Golomb and G. Gong, *Signal design for good correlation: for wireless communication, cryptography, and radar*. Cambridge University Press, 2005.

[30] A. Peinado, J. Munilla, and A. Fúster-Sabater, ''Improving the period and linear span of the sequences generated by dlfsrs,'' in *Inter. Joint Conference SOCO14-CISIS14-ICEUTE14*, pp. 397--406, Springer, 2014.

[31] V. Ciriani, S. D. C. di Vimercati, S. Foresti, and P. Samarati, ''κ-anonymity,'' in *Secure data management in decentralized systems*, pp. 323--353, Springer, 2007.

[32] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, ''The new casper: query processing for location services without compromising privacy,'' in *Proceedings of the 32nd international conference on Very large data bases*, pp. 763--774, VLDB Endowment, 2007.

[33] P. Kalnis, G. Ghinita, K. Mouratidis, and D. Papadias, ''Preventing location-based identity inference in anonymous spatial queries,'' *IEEE transactions on knowledge and data engineering*, vol. 19, no. 12, pp. 1719--1733, 2007.

[34] M. Duckham and L. Kulik, ''A formal model of obfuscation and negotiation for location privacy,'' in *Inter. Conf. on Pervasive Computing*, pp. 152--170, Springer, 2005.

[35] J. Krumm, ''Inference attacks on location tracks,'' in *International Conference on Pervasive Computing*, pp. 127--143, Springer, 2007.

[36] H. Lu, C. S. Jensen, and M. L. Yiu, ''Pad: privacy-area aware, dummy-based location privacy in mobile services,'' in *Proceedings of the Seventh ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pp. 16--23, ACM, 2008.

[37] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda, ''Peir, the personal environmental impact report, as a platform for participatory sensing systems research,'' in *Proc. of the 7th Inter. Conf. on Mobile systems, applications, and services*, pp. 55--68, ACM, 2009.

[38] M. Duckham, ''Moving forward: location privacy and location awareness,'' in *Proc. of the 3rd ACM SIGSPATIAL Inter. Workshop on Security and Privacy in GIS and LBS*, pp. 1--3, ACM, 2010.

[39] K. P. Puttaswamy, S. Wang, T. Steinbauer, D. Agrawal, A. El Abbadi, C. Kruegel, and B. Y. Zhao, ''Preserving location privacy in geosocial applications,'' *IEEE Trans. on Mobile Computing*, vol. 13, no. 1, pp. 159--173, 2014.

[40] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, ''Geo-indistinguishability: Differential privacy for location-based systems,'' in *Proc, of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 901--914, ACM, 2013.

[41] M. Vlachos, J. Schneider, and V. G. Vassiliadis, ''On data publishing with clustering preservation,'' *ACM Trans. on Knowledge Discovery from Data*, vol. 9, no. 3, p. 23, 2015.

[42] J. W. S.H Seo and E. Bertino, ''pclsc-tkem: a pairing-free certificate-less signcryption-tag key encapsulation mechanism for a privacy-preserving iot,'' *Transactions on Data Privacy*, vol. 9, no. 2, pp. 101--130, 2016.

[43] S. Uppoor, D. Naboulsi, and M. Fiore, ''Vehicular mobility trace,'' 2016.

[44] T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, ''The 128-bit blockcipher clefia,'' in *International Workshop on Fast Software Encryption*, pp. 181--195, Springer, 2007.

[45] E. B. Barker and J. M. Kelsey, *Recommendation for random number generation using deterministic random bit generators (revised)*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, IT Laboratory, 2007.