PROVIDING PRIVACY-AWARE INCENTIVES IN MOBILE SENSING SYSTEMS

KORABANDI SRUJANA[1] **M. PRAVEEN KUMAR**[2]

[1]M.Tech Student, Dept of CSE, NALANDA INSTITUTE OF ENGINEERING TECHNOLOGY, AP
[2]Assoc professor, Dept of CSE, NALANDA INSTITUTE OF ENGINEERING TECHNOLOGY, AP

***Abstract*-Mobile sensing relies on data that users contribute through their mobile devices (e.g., smart phones) to get useful information about people and their surroundings. However, users may not want to contribute due to lack of incentives and concerns about the potential for privacy leakage. To enhance user engagement effectively, both incentive and privacy issues should be addressed. Although incentives and privacy have been dealt with separately in mobile sensors, they remain an open problem to be addressed simultaneously. In this paper, we propose two credit-based credit-based incentives for mobile sensor systems, where the focus is on protecting privacy rather than designing incentive mechanisms. Our plans enable mobile users to earn credits by contributing data without leaking data they have contributed, and ensuring that malicious users do not abuse the system for unlimited balances. Specifically, the first scenario addresses scenarios in which a trusted third party is available over the Internet (TTP) and relies on TTP to protect user privacy and prevent abusive attacks. The second schema is scenarios where TTP is not available over the Internet. They apply to blind signature, partially blind signing, and the new extended Merkle tree technology to protect user privacy and prevent abusive attacks. Security analysis and cost assessment shows that our schemes are safe and effective.**

Index Terms—Privacy, incentive, mobile, sensing

I. INTRODUCTION

The ever-increasing popularity of mobile devices such as smart phones, tablets and a rich array of built-in sensors that usually come with it (eg, GPS, accelerometer and microphone) has created a great opportunity for sensing. Mobile sensor tries to take advantage of this opportunity by collecting sensor data from mobile devices and using data to get rich information about people and their surroundings. It has many applications in the field of health care [1], [2], traffic control [3], and environmental monitoring [4].However, the widespread deployment of mobile sensorBlocks applications by two obstacles. First, there is a lack of incentives for mobile users to participate in mobile sensing. To participate, the user must operate their sensors to measure the data (for example, to obtain GPS sites), which may consume a lot of power from its Smartphone. Also, the user needs to upload the data to a server that may consume a lot of its share of 3G data (for example, when data is images). In addition, the user may have to go to a specific location to sense the required data. Given these efforts and resources required from the user, the incentive plan is highly required for the multiplication of mobile sensor applications. Second, private information may be derived from data contributed by the user. This privacy concern also prevents users from participating. For example, to monitor the spread of a new flu, the server will collect information about whohave He was hit by the flu. However, the patient may not wish to provide such information because it is very sensitive. To motivate users to participate effectively, obstacles must be overcome. [5], [6], [7], [8], [9], [10], [11], [12], [13], [14] were proposed to provide anonymity for users , [19], [20], [21], [22], [23], [24], [25], [26], [27], [28] were designed to encourage participation by paying balances to users. However, they address privacy and incentives separately. It is not an alternative to addressing incentives and privacy at the same time. One may simply consider combining a privacy protection system with a credit-based incentive system to provide both privacy and motivation, but this group is not easy because these schemes have been designed under various system models and assumptions. More importantly, simple combination can not address the new challenges that arise only when both motivation and privacy are seen, and are not addressed by the privacy protection system or incentive system. In particular, existing privacy systems provide anonymity for users. Hiding a greedy user may allow unlimited anonymous data reporting for the same sensor function (which is always undesirable) and gain unlimited balances without being detected. This will increase the cost of data collection. Moreover, under the protection of anonymity, a malicious user who has threatened mobile devices to other users can steal user security credentials such as cryptographic keys and use anonymous identity data to cheat and gain as many credits as possible without being detected. Thus, the main major challenge in designing credit-based credit-based incentive schemes for mobile sensor is how to prevent many assault attacks while maintainingOur previous work [29] is to design a privacy incentive system for a special scenario for mobile sensing where each sensor task requires only one data report from each user (this task is referred to as an individual reporting task). An example of a single report task is "reporting the noise level around you now", which only requires each user to report individually on a measured noise level. But in the real world, there are many sensor tasks that require multiple reports to be presented at different times of each user (this task is referred to as the Multi-Report Task). An example of a multi-task report is "Noise report around you every 10 minutes in the following week" . Several other examples can be found in various mobile sensor systems [3], [4]. Unfortunately, this work can not be extended directly to support multiple reporting tasks, since building its own encryption allows only each user to earn credits from a single report. Although it is possible to create one task per report and then apply this schema, this will result in a significant increase in account and connectivity, and greatly complicates task management. For example, to gather the same amount of data as the multiple reporting task described above, you must create an individual report task every 10 minutes, and you must calculate, distribute, and process each set of encrypted credentials for each task.

In this paper, we propose two privacy incentives for mobile sensors, and they can support multiple report tasks. We adopt a credit-based approach that allows each user to earn credits by contributing to his data without leaking the data he has contributed. At the same time, the approach ensures that malicious users can not abuse the system to earn an unlimited amount of credit. In particular, the first scenario is designed for scenarios where a reliable third-party online (TTP) is available. It relies on TTP to protect privacy and prevent attacks, and has a very low account cost per user. The second schema does not require any TTP over the Internet. They apply to blind signature, partially blind signing, and the extended Merle tree to protect privacy and prevent abusive attacks. The rest of this paper is organized as follows. Section 2 introduces system models. Section 3 provides an overview of our solution. Section 4 and Section 5 provide incentive programs. Section 6 provides cost estimates. Section 7 provides discussions. The last two sections review the relevant work and conclude the paper. Total. This challenge calls for new designs that address the overall synergy and privacy. This challenge calls for new designs that deal in an integrative and motivating manner.

2 PRELIMINARIES

2.1 System Model

The system contains a data collector and a set of mobile nodes (for example, mobile devices such as smart phones carried by people and installed on vehicles). The mobile contract with the complex continues through 3G / 4G networks, WiFi and other available networks. The collector collects sensor data from the mobile contract and may use data to provide services to third parties Entities for various applications.

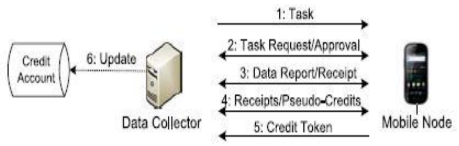


Fig. 1.System model.

To enhance participation, the pool pays balances for the contract for the sensor data it contributed. Credits can be converted into real-world financial rewards, or used to purchase a mobile sensor service from the pool. In this way, nodes are stimulated to contribute data. The system model appears in Figure 1. For data collection, the assembler creates remote sensing tasks and adds them to an active task queue. The task determines the type of sensor readings needed, where and when to sense, the number of required data reports from each node, the number of credits paid per node, the creation time, and the expiration time. At random intervals, each node (using a randomly generated and unlinked alias) connects with the wrapper to retrieve active tasks. For example, a random interval of time can be expected between two consecutive retrieval processes, and can also retrieve tasks at random time regularly during each predetermined period (for example, every day). Random loop times are allocated to prevent the wrapper from binding the loopback sequence by the node itself. Among retrieved jobs, the node identifies the tasks to be accepted. If they want to assign an accepted job, they send a request to the assembly tool in a new connection using a new alias. The pool displays approval if it approves the node request. The task is then assigned to the node. For a specific task, the node collects the sensor data as specified by the task. Then, using a new pseudonym, the sensor data is sent in a report, and the collector issues a receipt at the same communication session. If multiple reports are needed at different times or locations through the task, the node will submit each report using a different alias in a separate connection to the community. When a task collects enough reports, which are assigned to sufficient or expired nodes, the wrapper will delete them from the active task queue. After the node has finished reporting the task, you (using an alias) send the receipts for that task to the collector for refunds. Because the wrapper does not know the identity of the node, it issues pseudo credits for the node that is converted to credit codes by the node. The conversion between a pseudo credit and a credit code depends on a secret known only to the node, so the collector can not associate the token with the false balance or know the task from which the credit is earned. For each token, Wait for the node randomly, and then, using its real identity, you can load the token to the cover. The pool maintains a credit account for each node in the system, and the deposit account's credit account accordingly. For different tasks, the number of balances paid per node may vary depending on factors such as the sensor data type needed by the task (such as image or acceleration reading), the number of reports required per node, and other task requirements (for example,). In general, high cost (for example, bandwidth, power consumption and human attention) is stimulated in order to connect the nodeData for a task, you must pay more credits for the task. In this paper, we use c to indicate the number of balances paid for each report node for the task. The value of the task c is set by the assembly tool. Note that the node can choose not to accept a task if the task is pushed at a very low rate. One of the interesting questions for the complex is how to adjust c to reduce the total balances paid for the task, but this issue is beyond the scope of this paper due to limited space, and we plan to explore it in future future work. Although the value of task c is unknown until the task is created, we assume that it contains a maximum C (system parameter), since it is not entirely possible to pay unlimited balances for the task. In practice, the compiler can set an initial value of C based on the estimate, and then update C according to dynamic needs. One important issue for the complex is to control the cost of data collection (ie, the number of balances paid to the contract). To do this, the assembler needs to control the number of nodesIt can report to each task. Such control is done through the task request and approval step.

2.2 Threat Models

Threats to privacy. The compiler wants to know the reports sent by the node and the tasks that the node has accepted. Tries to obtain this information by analyzing the text of our protocol. A task-intensive attack (where a character assembles a task that only a narrow set of nodes can answer), so it is difficult to identify the nodes that respond to the task (and the selective attack of tasks) in which the compiler divides only one or a few nodes, Easy to link reports provided by the node itself) is not the focus of this paper. However, we note that these attacks have been addressed by the current work [5], [6] and these solutions can be easily adapted to our own setup. Specifically, to alleviate narrow tasks, we can provide a recording authority (as applicable in [5]) to ensure that tasks are not targeted to a narrow set of nodes, and the pool can publish those tasks that have been verified and signed by the authority only. The basic idea of defense against the selective task proposed in [6] can also be applied as follows. By comparing active tasks that were retrieved at different times, the node can estimate the length of time that the task remains in the active task queue. Then, based on the predetermined period in which each node retrieves active tasks at once, the node can estimate the number of nodes that recovered the task, and accept the task only if it retrieves the contract. As in [5], [6], connections between the collector and nodes are assumed to be anonymous, for example, by Mix Networks and IP technologies for address recycling.

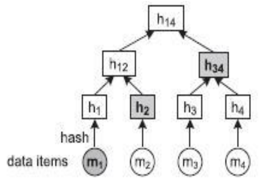


Fig. 2.An example of Merkle tree.

Threats to incentives. The contract is greedy and they may deviate from our protocol to earn as many loans as possible. For example, the node may send multiple groups (instead of one set that is supposed to be submitted) from reports for each task to earn multiple rewards of the task. Also, the node may be able to break through some other nodes, get its secrets, and use these secrets to earn more credits. As far as motivation is concerned, assume that the collector is sincere. Credits will be paid for the contract to report their data and keep their credit accounts properly as defined in our protocol. Because the complex may make a profit before Provision of services to other entities Based on the data collected, it is in the interest of the funds to be paid and encourage participation. For authentication purposes, the assembler and each node are issued a pair of public and private keys by an offline certificate authority. To thwart Sybil attacks, the CA ensures that users can not create nodes, and each node can obtain only one set of authentication keys. The opponent may display a node and identify node keys, but can not bind the keys to another arbitrary node. The connection between the key and the node can only be done through a certificate signed by the CA. Data falsification processes are removed where there is a malicious contract that presents disturbing sensor data outside this paper, and possible solutions are discussed in Section 7.

2.3 Our Goals

In terms of privacy, our goal is to ensure that the compiler can not link any report to the report node, link multiple reports sent from the same node, see if a particular node has accepted a particular task, or link multiple tasks accepted by the node. For the stimulus, our goal is to ensure that the node will not be able to earn more credits than our protocol allows. Specifically, if the node reports on a task, it can only gain c and c (the task push rate) from the task; if a task is not assigned to the task or it does not report the task, it does not achieve anything.

2.4 Primitives Encryption

Our plan mainly uses three basic encrypted objects, the Merkle tree, a blind signature, and a partially blind signature. The Tree of Merkel [30]. The Merkle tree is an efficient and secure binary structure commonly used to verify that a set of committed data elements has not been changed. It is based on the use of one-way hash functionality. In the Merkle tree, each node of the tree tree is a fragmentation of one data element, and each node is an integral part of its two children. Figure 2 shows an example of a Merkle tree based on four elements of data. For linking data elements, h14 is sent to the root of the tree to the checker. Later, to demonstrate that a data element, for example, m1, is included in the tree, h2 and h34 are sent to the verification tool.

Blind signature. Through the blind signature scheme [31], the user can obtain a signature from one of the signatories on message m without revealing m to the site. Specifically, the user m evaluates a random blind to obtain a m0 blind message and sends m0 to the site. Using the standard digital signature algorithm (for example, RSA), the site signs m0 and resets s0 to the user. Then the user can obtain a signature on CE by removing the cryptographic agent from s0. The blind signature has two properties, blindness which guarantees that hm; si cannot be linked to m0 or s0, and unforgeability which guarantees that the user Cannot get valid signature of s0 for another message m00 6 m m. In this search, the blind RSA signature is used [32] due to its simplicity. RSA algorithm is based on. Qi and HD; Qi refers to the public key of the site and private key respectively, where Q is the general standard. If the user wants to get a blind signature on message m, it calculates m0 mm zemodQ, where z is a random value chosen by the user and relatively initial to Q. The sign is m0 using the standard RSA algorithm, and modq s0 ðm0dd modQ returns to the user. Then the user calculates s ss0 z 1Þ modQ which is the signature m.Partially blind signature Partially blind signature systems (eg, [33]) also enable the user to obtain a signature on a letter from one of the signatories without allowing the site to be identified. However, the site can clearly indicate some common information (for example, the release date) in the signature. The site can not associate the signature with the message or the contact session that is obtained from the signature, since the shared information is included in many signatures. The above absorption properties are also mholds here. In this paper, we do not assume that any signature system is partially obscured. Let PBSKpp; m refers to a partially blind signature of message m, where p is the common information and K is the signature key.

3 AN OVERVIEW OF OUR APPROACH

The main challenge is to design a credit-based incentive system for mobile sensing in how to prevent abuse attacks (which are mismanaged to gain too much credit) while maintaining privacy. This problem becomes particularly difficult to resolve when an attacker compensates other nodes and uses their credentials anonymously to earn credits.
Adopt our plans creatively and use a set of symbols to achieve goals on incentives and privacy. It includes the request code used to request a task, a token used to send a data report, a receipt issued to a node after a data report, and a credit symbol that can be deposited to earn credits. To prevent abuse attacks, each node already identifies the token for the request, receipts, and credit token that you will use to process each future task and is committed to using it for this task.

To protect privacy, codes and obligations are designed and used in a manner that preserves privacy. To facilitate the distribution of symbols, tasks are indexed as 1, 2, 3, ... in order of creation time. Tasks are grouped into important windows of W size (system parameter, for example, W 1 1; 000). The first Task window contains Tasks 1, 2, and. . The second task window contains tasks W þ 1, W þ 2, and W. . . , 2W; etc and so on. In our schemes, tokens are created and distributed based on task windows. When the system starts (that is, before any task is created), the assembler and nodes create distinct symbols for the first task window. When you create more mtasks, the first task window is filled with more tasks. When the number of created tasks is approaching W (that is, when the first task window is almost full), the assembler and nodes create distinct icons for the second task window;

Our plans work independently for each important window in five stages: Setup. This stage occurs before you create any task in the task window. At this stage, the tokens and obligations nfor the task window are marked and distributed appropriately on the nodes and collector. Task tasks. Assume that a node has retrieved task I from the wrapper over an anonymous connection session. If the node wants to assign this task, it sends a request to the assembly tool that contains its request code. The pool checks the commitment to function i in the preparation stage. If approved by the Synod request, it returns an approval message to the node. From the acceptance message, the node can calculate report tokens for task i. However, the

node can not derive any token without the approval message. Submission of the report. After the node creates a report for task i, it presents the report and the token for task i through an anonymous session. The wrapper checks that the report token has been committed for task i, and
And then issues a receipt to the node. Submit Receipt. After you submit all the required reports for a task, the node waits for some random time and then sends the receipts to the wrapper. The wrapper checks the receipts, and then issues false credits to the node. From false credits, the contract can generate some credit codes. Can not get any token without false credits. Deposit of credit. After the node gets a token, it waits for some random time, then you upload the token to the collector.

The collector checks that the token has been committed and increases its credit account. To prevent abuse attacks, in the preparation and depositing phases, each node contacts the university using its true identity and validates the keys issued by the certification authority.

Although the same phases of the protocol are shared, the proposed planners have different approaches to symbolic structures and commitment. The first schema assumes a reliable third-party presence over the Internet, and uses TTP to create each node's token and its obligations. It relies on TTP to protect privacy and prevent attacks, and has a very low account cost. The second schema does not assume any TTP over the Internet. Each node generates symbols and commitments in collaboration with the Synod using the blind signature, the partially blind signature, and the extended Merkle tree. These technologies have a higher account cost, but they protect the privacy of each party against any third party. The codifications used in table 1 are summarized.

4 A TTP-Based STEME

This schema assumes a TTP connection over the Internet, but this assumption can be mitigated as described in Section 7. The compiler uses two K2 and K3 keys to generate partially blind signatures. These keys are issued by a certificate reference (possibly offline).

4.1 Basic Plan

Without losing the public, consider the first task window when describing our plan (see Figure 3).

Table 1
The symbols used in this paper

K1;2;3;4 The collector’s private keys to generate partially
blind signature
e;d The collector’s public and private key to generateblind RSA signature
s1;2;3;4 The secrets of a node
NNum. of nodes in the system
W Num. of tasks in each task window
n Num. of reports that a task needs from each node
c Num. of credits paid for a task to each node
C Max.num. of credits paid for a task to each node
H A cryptographic hash function
tRequest token in the TTP-based scheme;
Request token identifier in the TTP-free scheme
g Report token identifier
b Receipt identifier
’ Credit token in the TTP-based scheme
m Credit token identifier in the TTP-free scheme
m0 Blinded credit token id in the TTP-free scheme
.

4.1.1 Setup

TTP sets and delivers s s to each node and the key secret key to the assembly tool. The different secrets of the contract differ. TTP also creates a nonce r to identify this set of secrets, and send them to each node and compiler.If a new set of secrets for the collector and nodes are set later, a new unused number will be created.TTP calculates other credentials using a set of secrets and manual dislocation. First half how to create tokens and obligations for a single node. Let us show the secret of this knot. Of s and nonce r, TTP is derived from two other families

$$s_1 = H(\frac{s}{p}/1)$$

and

$$s_2 = H(\frac{s}{p}/2)$$

Step 1. The TTP computes W request tokens for the node.Each token will be used for one task. The token for task I (i 2 ½1;W) is. Here, the one-wetness of hash chain is exploited to calculate ti (see explanations in Section 4.2). The commitment to ti is

$$< H(\tau_i), i >$$

Step 2. Calculate the TTP CW token for the node. Because TTP does not know at this time how many credits the collector will pay for each task, it generates as many token as possible for each task. The tokens for task i are calculated. The ij commitment is when NID is the true identity of the node. Similarly, TTP can create codes and obligations for other nodes. It randomly mixes each type of obligation and sends it to a collector.

Finally, each node gets one secret and the other inaccurate. The mosque has one secret key, which is an unrelated obligation, to the NW's commitment to distinctive requests and the NCW's commitments to credit codes. TTP stores the secret key of the pool, the secret of each node and nonce.

4.1.2 Tasks task

When the wrapper publishes task i, it also publishes n and c (see Table 1). Suppose a node has recovered task i. If you decide to accept this task, they send an anonymous request without mentioning the name. The request contains the request token for this task, which is

$$\tau_i = H(o/H(s_i))$$

where

$$s_1 = H(\frac{s}{p}/1)$$

node !collector :i; τ_i:
(1) The collector verifies that is a valid commitmentand delete this commitment to avoid reuse of this token. If it approves this request, it tags t as approved. In this case, the node can request n report tokens for task i. It generates n random values g1, g2, . . .,gn, and

4.1.3 Report Submission

The node can submit one report using each report token. To submit the jth (j1; . . . ; n) report for task i, it anonymously sends the following message:
node !collector :$i, \gamma_j, PBs_{k2}(i, \gamma_j)$,report

(3) The collector verifies the signature and acceptsthe report. Then it can issue a receipt to the node. Specifically, the node generates and obtains
node !collector :$i, \gamma_j, PBs_{k2}(i, b_j)$,report

4.1.4 Receipts Submission

After submitting n reports for task i, the node can collect n receipts. After waiting for some random time, it can submit these receipts to the collector to redeem c credits. It sends: node !collector :i; ti;
The collector checks that ti is an approved request token identifier for i, which means the node has been assignedtask i. It verifies that the n partially blind signatures arevalid, which means the node has submitted n reports for task i. It also verifies.

4.1.5 Credit Deposit

After the node gets a token, wait for a randomly selected time period of 00; T_ to mitigate the timing attacks (see section 4.3) and then deposit the code using its real identity NID: node! The sample collector: NID; '(7) checks the wrapper of this, and NIDi is a valid obligation and deletes it to avoid reusing the code. Then increases the node's credit account by one.

4.1.6 Renew commitment

When the current task window is almost full, the assembler must communicate with TTP to obtain another set of commitments for the next task window. The secret key of the complex and the secrets of the contract and reference are not changed.

4.2 Dealing with links and dynamic papers

join. In the setup phase, TTP assumes that there is a virtual contract V (system parameter) next to the real N nodes. Generates symbols and commitments for both real and virtual nodes. Also, it sends commitments to the application codes of the default nodes, mixed with the obligations of the real contract, to the collector. When a new node joins, TTP sets it to an unused default node and sends the r secret to the virtual node. Also, the TTP generates the credit codes for the new node (ie the default node assigned) and sends their obligations to
Mosque. Then, it marks the specified default node used. No changes are made to the other nodes.
If no unused virtual node is available when the new node joins, TTP restarts the setup phase again as a new set of secrets for the pool and all current nodes are released, as well as a new set of default nodes. Some nodes may not have access to the network during the setup phase and therefore can not receive the new nonce and their new secrets.

To work around this problem, when the node retrieves the tasks from the assembly tool, they verify whether they contain the same non rce value with the assembly tool. Note that the collector always has the latest version of nonce. If the node is not old, it means that the node has missed the previous setup and that its secret is also old. In this case, the node connects to TTP to update its secret and disarm.

In practice, the value of parameter V can be adjustedOn a foam basis. If the torque rate is high (ie, joining the new contract frequently), larger V can be used to reduce the number of restarts of the expensive setup stage, at the highest storage expense in the pool. If the shrinkage rate is low, smaller V can be used to reduce storage space in the pool.

Leave. When you leave the node, you must revoke the symbols you requested for future tasks in the wrapper. Note that if the dialing token has been deactivated for a future task, the credit tokens for the same task will also be automatically deactivated so that the node can not leave its account.

4.3 Timing Processing

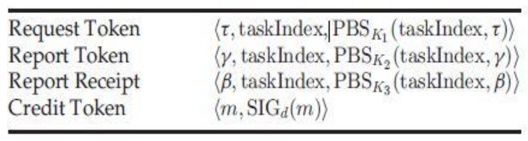
If the node deposits an earned credit code from the report as soon as it submits the report, since it uses its real identity to deposit the token, the compiler may be able to link the report to it by analyzing the timing.

Thus, the node must wait some time before you deposit the credit code. In particular, after the node gets a token, it waits for a randomly selected time period of 00; T_ and then submits the token.
The T parameter is large enough (for example, one month) so many tasks can be created in each T period of time periods, and most nodes have opportunities to connect to the complex.

4.4 Remove Commitment

The pool removes obligations to previous W tasks as follows. Note that part of the obligations are removed to avoid reusing the code immediately after verifying the corresponding tokens. Since all contracts do not accept all functions, some obligations may remain after the previous W
Tasks processed. Let texp indicate the maximum time that each previous W task ends. Note that all reports for tasks W are submitted before texp and all paid credit codes for these reports are sent to a contract before texp. Thus, the compiler can remove the remaining commitments to request the tokens after texp time. To allow the contract to deposit their acquired credit codes, the collector stores the remaining commitments of the token for a different time period than T (as shown in section 4.3), and removes it after texp þ time. If the node (for example,) Wants to deposit some credit tokens after deleting their obligations, the collector can verify the validity of these tokens using TTP, and update the node's credit account accordingly.

TABLE 2
Tokens in the TTP-Free Scheme



Request Token	$\langle \tau, \text{taskIndex}, \| \text{PBS}_{K_1}(\text{taskIndex}, \tau) \rangle$
Report Token	$\langle \gamma, \text{taskIndex}, \text{PBS}_{K_2}(\text{taskIndex}, \gamma) \rangle$
Report Receipt	$\langle \beta, \text{taskIndex}, \text{PBS}_{K_3}(\text{taskIndex}, \beta) \rangle$
Credit Token	$\langle m, \text{SIG}_d(m) \rangle$

5 A TTP-FREE SCHEME

The compiler uses a special key d to generate blind RSA signatures, and uses four special keys K1, K2, K3 and K4 to generate partially unsigned signatures. These keys are issued by a certificate reference (possibly offline). Each node has four secrets s1, s2, s3 and s4 that are created by itself. You do not have to change keys and secrets in different task windows. The icons are summarized in Table 2.

5.1 Basic Plan

Without losing the public, consider the first task window when describing our plan (see Figure 5).
5.1.1 Setup Before you create any task in this task window, each nodeConnects to the collector using his or her real identity to get symbols and commitments for tasks in this window. The reason for calculating m in thisThe method will be explained later. The node will use these identifiers to generate C credit codes to handle task i. Specifically, each ID token consists of the compound's RSA ID and signature above the identifier, meaning that the contract can not obtain the signature until it reports to that task.At this stage, the node is committed to collecting amateurs
These credit codes will be used for task i. To do this at a low computation cost, the node builds a Merkle tree extended to mi1,. . . , MiC (see details in section 5.2), and then gets a blind partial signature from the root collector'sTree fragmentation, ie PBSK1D. TTH. This signature is a commitmentTo C Codes Credit. Hizbut-Tahrir. I; PBSK1D. Tii will also be used as the node request icon for task i. The node also needs to associate these credit codes with its identity. To do this with low cost, andThe node builds an extended Merkle tree (see section 5.2) on all CW token credit identifiers for tasks in the task window. Let a sign indicate the root of this tree. Node, to move, Alice, send ha; Alicei to the pool.
In total, node W gets partially blind signatures, one for each task in the task window. These signatures store these tasks in the window later.

5.1.2 Task Request

When the wrapper publishes task i, it also publishes n and c (see Table 1). Suppose a node has recovered task i. If you want to accept this task, it uses a pseudonym to send a request to the assembly tool that contains the request code for task i.Detection tokenSince the node usually does not report all tasks and does not pay each assignment at C Credits, some of the credit token IDs that are committed in the setup phase are not used in credit codes. To prevent a node from reusing these IDs for more credits than is allowed, each node is required to detect unused credit token IDs. (Note that those credit-specific IDs used in credit codes can also be detected when you deposit tokens.) There are two detection cases that match specific tasks and non-specified tasks, respectively. For a task assigned to a node, the node detects the m unused whenSend report receipts to the pool and get proof of the task, such as signing the partially blind site with the K4 key. For those tasks that are not assigned to a node, the node does not set it

5.2 Remove the token

For a node, the reporting token, the report receipt, and the credit token can be ignored after use. The token for the dialing and task token IDs for a task after the send phase can be ignored if the node has sent task reports or after the token detection phase otherwise.

The compound stores ha ha. Pair nodeIDi each
A node is used to associate its credit token IDs from an important window until the duration of T passes after the last unexpired job in the window Although the task index can be linked to its report and the request code (in addition to the objects accessed through them in Figure 7) It can not be linked to stored credit codes, tree root, or node identity. Therefore, the assembler does not know whether the node has accepted or submitted reports for a particular task. Because the report can only be linked to the report code, and the tokens used by the same node are created independently using partially unsigned signatures, the compiler can not link multiple reports sent by the same node. Because node request token is created independently with partially guaranteed signatures, it is impossible to bind multiple tasks requested by the same node.

5.3 Security Analysis

This section analyzes how to achieve privacy and incentive goals.

5.3.1 Attacks on privacy

Figure 7 shows the possibility of linking the different symbols and objects in our chart. From the format, it is easy to see that the assembler can

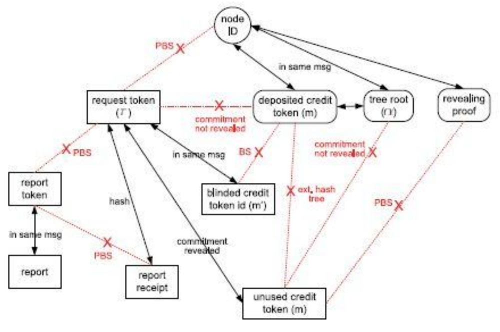


Fig. 3.The linkability between different Ingredients. Round rectangles refer to sent items with the true identity of the node (random aliases). Texts along solid arrows explain why the two documents are linked (non-cancellable).

5.3.2 Attacks on incentives

The striker works alone. In the setup phase of the task window, each node (with its real identity) can bind to only one and one digit (and the C token identifiers used for calculating the extended Merkle tree) for each task in the window. The CW token symbolic ID node can also bind its identity through another extended Merkle tree. Since the binding takes place before the node identifies any task in the window, the best strategy for the node is to associate its knowledge with the distinctive symbols that it has committed to each task through t, so that it has the ability to gain balances from each task. Also, the node can not use the request token, the report code, and task j receipts to obtain balances from another task, as those tokens have been adhered to by partially guaranteed signatures. As a result, the node can only use the request token setC credit to which you have committed the first task to earn credits from task i. If the task is not assigned to an attacker, it can not report. If the job is assigned but does not provide reports, it can not receive a receipt for the report. Either way, you will not get any credit. Thus, an attacker acting alone can not make our scheme fail to achieve the goal of motivation.

The attacker controls the other nodes. Suppose an attacker has offered some contract. Since each node must use its true identity in the setup phase, like an analysis of attackers working on their own, an attacker can bind only one set of request token and token ID (m1, ..., mc) to each task and its true identity even if She had offered the other contract. Furthermore, the token detection scheme ensures thatAll credit token IDs will be detected in the build-up phase of the pool (because if the attacker or the hacked node does not detect the token IDs, they will not get new icons for future task windows), and the collector checks that it is different. Thus, if a credit code is committed to a hacked node, even if the attacker can steal the token from the node, it can not upload the token to its own account. In the light of the order code, the receipts sent with it and the credit scores already obtained in the setup phase are determined by the use of one-way fragmentation, the extended Merkle tree, and the non-permeability characteristics of the blind signature (see Figure 3). Thus, requesting token and committed receipts to a hacked node can only result in the appearance of credit codes committed to hackersNode, an attacker can not use them to earn more credits.

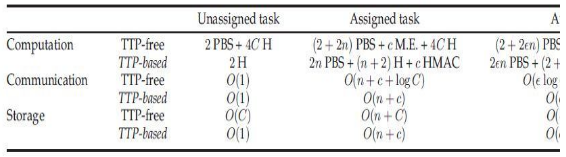
6.COST RATING

In this section, we analyze and evaluate the cost of our products Incentive plans. The cost of reading sensors and data is not analyzed here.

6.1 Cost Analysis

Cost per node. Table 3 summarizes the calculation, communication and storage costs of planners. The cost of a task to a node depends on whether the node has been assigned to the task (that is, if the nodes are approved on

TABLE 3
Each Node’s Computation, Communication, and Storage Cost per Task



		Unassigned task	Assigned task	A
Computation	TTP-free	$2\,\mathrm{PBS} + 4C\,\mathrm{H}$	$(2+2n)\,\mathrm{PBS} + c\,\mathrm{M.E.} + 4C\,\mathrm{H}$	$(2+2\epsilon n)$ PBS
	TTP-based	$2\,\mathrm{H}$	$2n\,\mathrm{PBS} + (n+2)\,\mathrm{H} + c\,\mathrm{HMAC}$	$2\epsilon n$ PBS + (2 +
Communication	TTP-free	$O(1)$	$O(n+c+\log C)$	$O(\epsilon$ log
	TTP-based	$O(1)$	$O(n+c)$	O(
Storage	TTP-free	$O(C)$	$O(n+C)$	O(
	TTP-based	$O(1)$	$O(n+c)$	O(

Reports to this task). In both systems, for an unspecified task, the cost is low; for the particular task, the incremental cost comes from reporting and obtaining credits. In both systems, the cost of an undefined task is much lower than the assigned task. To evaluate the average cost per task, we use a parameter to indicate the average fraction of assigned tasks per node out of allCreate tasks and show the average cost in the table. In a large system with many tasks, we expect that each node can accept and only a small portion of the tasks are assigned dueReduce resources. For example, a node that lives in Newark may not be able to answer tasks that require location-based data from New York. Thus, expected to be very small, for example, 1. The average cost will be close to the cost of unallocated tasks. Because the Hash function runs size commands faster than the partially-closed signature and C can reach a few hundred or even greater practicality (for example, a task may be paid from $ 6 to $ 220 in Gigwalk [34] which means C> 220), the TTP-based system has a much lower calculation cost than the TTP-free scheme and especially for unassigned tasks.

For storage cost, each node is stored in one secret and one vulnerability in the TTP-based schema, and the C token IDs are stored for each task for a short time in the TTP system (see section 5.3). Hence the cost of storage is low. If a stonemason has sent mission reports, it also stores report codes, receipts, and credit token for a short time. As modern smart phones typically contain many GB volumes,Storage cost is not a big issue. Cost in the pool. Note that the average can also be indicatedPart of the nodes that each task is assigned to. The calculation, storage and storage cost of the compound is summarized in Table 2. We provide a rough estimate of the storage size. In the TTP-based schema, the complex mainly stores WNN þ V ðCC þ 1 for commitments for next W tasks. It also stores code obligations CNN þ V to credit for each taskIt was created in the past time window for T. Let's think of a simple case. Suppose N 10 10; 000, V 1 1; 000, WAlso, suppose you use SHA-256 as an H-H function, and each task ID or node ID has an 8-byte. Then the storage in the wrapper is about 400 GB. We expect that this storage cost is not a problem for modern servers. In the TTP-free system, the pool mainly stores code IDs for the last few tasks. The amount of storage is expected to be lower than the TTP-based schema.

6.2 Implementation

We have implemented our plans in Java. The NSA is partially blinded [35] as a PBS schema, and SHA-256 is used as a HASH function. Based on the application, we measure PBS runtime, RSA signature, standard exponential,And HMAC on Android Nexus S Phone (Android 4.0.4 OS, 1 GHz CPU and 512 MB RAM) and a laptop (Windows 7 OS, 2.6 GHz, and 4 GB of RAM). The results show that when generating a partially blind signature, the processes in the node and pool differ. Then we calculate the runtime of the planners according to Tables 3 and 4. Here, we set C 256 256 and 0 0: 1. For n and c, we consider four extreme cases that correspond to four typical types of tasks with varying numbers of reports and credits: n c c 1 1 (Type I), n 1 1; c 256 256 (Type II), n 256 256 ; C 256 256 (type III), n 256 256; c 1 (type IV). Results appear at run time. We found that in the TTP-free system, when small, Hash operations become an important source of operating time. However, the running time of both systems can be seen tooIn short all four kinds of tasks. The TTP-based system operates at least one order faster than the TTP-free schema on each node (on the smartphone), because of the use of more efficient encryption alternatives such as HASH and HMAC. For similar reasons, it runs faster in the wrapper (on your laptop). To study the feasibility of our plans, we also measure energy consumption for the TTP-free scheme on your Nexus S phone using Monsoon Power Monitor. Here, a TTP-free scheme is measured because it has higher power consumption than the TTP scheme. In this set of experiments, Nexus S Phone manages the full life cycle of one task at 100 Tasks. In this process, the smartphone connects to a laptop (Windows 8.1 OS, 2.4GHz CPU, and 4 GB RAM) with a TCP connection over WiFi, launching a new TCP connection for each stage.

Each data report contains 8 bytes, which is similar in size to the accelerometer, temperature, noise, and GPS reading. The results are shown in Table 3. It can be seen that the energy consumption in our scheme is very low. When 0 0: 1, it is as well

TABLE 4
The Average Running Time of Processing a Task

		Type I	Type II	Type III	Type IV
Node	TTP-free	90 ms	95 ms	116 ms	112 ms
	TTP-based	0.25 ms	0.31 ms	22 ms	22 ms
Collector*	TTP-free	10 ms	14 ms	37 ms	33 ms
	TTP-based	0.08 ms	0.34 ms	21 ms	21 ms

7 DISCUSSIONS

Relaxation Assumption TTP. In a TTP-based system, a trusted third party can be replaced by a fair but outsiders who do not collude with the complex or any node. Despite following our protocol, this third part attempts to deduce private information from the protocol version and through eavesdropping connections.

Under this semi-honest model, the only change that must be made is that all communications between the complex and the contract must be encrypted. Support for report-based payment. In the systems described above, the node is pushed after all reports are submittedMission. In practice, the node may be able to generate fewer than n reports for the task. In such scenarios, the pool can determine the number of credits paid to the node (for example, based on the number of receipts owned by the node) and issue false credits accordingly. Greed attacks. In our plans, after the node retrieves a task, it waits for a random time before asking the specialist to assign the task to him. This is to protect the privacy of the node. However, the greedy gang that does not care about its privacy may recover tasks constantly and request a task immediately after recovery, in order to have a better chance of assigning the task. Such behavior may prevent other nodes from earning credits. To ease it, the pool cana group ofSimilar tasks, it accepts one of them with a certain probability (for example, 0.5). This ensures that the number of admissible admissions does not exceed the number of similar task groups. Of the number of credits acquired by a node, the assembler does not know the tasks reported by the node, and therefore can not infer any special information about the node. Since each node deliberately ignores certain tasks, this method sacrifices some opportunities to earn better balancesTotal. Given the limited space, we will explore this issue in future work. Data fraud attacks.

Malicious nodes may send annoying sensor data to obtain balances. To mitigate this attack without breaking privacy, anonymous reputation schemes were proposed in the literature [9], [12] to filter the data provided by the reputational weaknesses. Another possible method is for each user to create a group signature [36] and attach it to their data report. If a report is detected as bad data, the compiler can use a reliable reference to retrieve the identity of the data source from the group signature. However, these methods rely on TTP over the Internet to protect privacy. When TTP is not available online, howTo mitigate the attacks of data fraud without privacy violation is still an open search problem will be explored in our future work.

8 RELATED WORK

[5], [6], [7], [8], [9], [10], [11], [12], [13], [14] Remote Sensing. Among them, AnonySense [5], [6] and PEPSI [8]Provide frameworks for collecting anonymous data. Numerous studies [37], [38], [39], [40] [41] address the collection of perceived data of privacy. Christine et al. [9] Wang et al. [12] Plan a reputation for privacy, employ a reputation to filter out incorrect sensor readings. DeCristofaro and Di Pietro, [42] consider a scenario where external entities are inquiredSpecific data for users and study how to hide any userMatches the query. TPM is also used to protect user data [10]. However, none of these privacy protection systems are incentives. Many of the incentive programs [15], [16], [17], [18], [19], [20], [21], [22], [23]], [27], [28] and has been designed for mobile sensing to pay user-based creditsGames and Auctions

9 CONCLUSIONS

To enhance user participation, we proposed two privacy-sensitive incentive schemes for mobile sensing, corresponding to scenarios with and without TTP, respectively. The TTP-based schema relies mainly on the hash and HMAC account on a very low cost basis for each node. Based on the blind signature, partially blind signature and extended Merkle tree techniques, the TTP-free plan has higher expenditures than the TTP-based schema,Ensures that no third party is exposed to user privacy. Both planners can support links and dynamic papers effectively.Implementation shows that both systems have a shorter run time and lower power consumption.

REFERENCES

[1] J. Hicks, N. Ramanathan, D. Kim, M. Monibi, J. Selsky, M. Hansen, and D. Estrin, “AndWellness: An open mobile system for activity and experience sampling,” in Proc. Wireless Health, 2010, pp. 34–43
.
[2] N. D. Lane, M. Mohammod, M. Lin, X. Yang, H. Lu, S. Ali, A. Doryab, E. Berke, T. Choudhury, and A. Campbell, “Bewell: A smartphone application to monitor, model and promote wellbeing,” presented at the 5th Int. ICST Conf. Pervasive Computing Technologies for Healthcare, Dublin, Ireland, 2011.

[3] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J.

Eriksson, "VTrack: Accurate, Energy-aware road traffic delay estimation using mobile phones," in Proc. 7th ACM Conf. Embedded Netw. Sens. Syst., 2009, pp. 85–98.

[4] 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. 7th Int. Conf. Mobile Syst. Appl. Serv., 2009, pp. 55–68.

[5] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos,"Anonysense: Privacy-aware people-centric sensing," in Proc. 6th Int. Conf. Mobile Syst. Appl. Serv., 2008, pp. 211–224.

[6] M. Shin, C. Cornelius, D. Peebles, A. Kapadia, D. Kotz, and N. Triandopoulos, "Anonysense: A system for anonymous opportunistic sensing," J. Pervasive Mobile Comput., vol. 7, no. 1, pp. 16–30, 2011.

[7] T. Das, P. Mohan, V. N. Padmanabhan, R. Ramjee, and A. Sharma, "PRISM: Platform for remote sensing using smartphones," in Proc. 8th Int. Conf. Mobile Syst. Appl. Serv., 2010, pp. 63–76.

[8] E. D. Cristofaro and C. Soriente, "Short paper: PEPSI-privacyenhancedparticipatory sensing infrastructure," in Proc. 4th ACM Conf. Wireless Netw. Security, 2011, pp. 23–28.

[9] D. Christin, C. Rosskopf, M. Hollick, L. A. Martucci, and S. S. Kanhere, "Incognisense: An Anonymity-preserving reputation framework for participatory sensing applications," in Proc. IEEE Int.
Conf. Pervasive Comput.Commun., 2012, pp. 135–143.

[10] P. Gilbert, L. P. Cox, J. Jung, and D.Wetherall, "Toward trustworthy mobile sensing," in Proc. 11th Workshop Mobile Comput. Syst. Appl., 2010, pp. 31–36.

[11] K. L. Huang, S. S. Kanhere, and W. Hu, "Towards privacy-sensitiveparticipatory sensing," in Proc. 5th Int. Workshop Sensor Netw.Syst. Pervasive Comput., 2009, pp. 1–6.

[12] X. O. Wang, W. Cheng, P. Mohapatra, and T. Abdelzaher,"Artsense: Anonymous reputation and trust in participatorysensing," in Proc. IEEE Conf. Comput. Commun., 2013, pp. 2517–2525.

[13] H. To, G. Ghinita, and C. Shahabi, "A framework for protectingworker location privacy in spatial crowdsourcing," Proc. VLDBEndowment, vol. 7, no. 10, pp. 919–930, 2014.

[14] I. Vergara-Laurens, D. Mendez, and M. Labrador, "Privacy, qualityof information, and energy consumption in participatory sensingsystems," in Proc. IEEE Int. Conf. Pervasive Comput. Commun.,2014, pp. 199–207.

[15] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones:Incentive mechanism design for mobile phone sensing,"in Proc. 18th annu. Int. Conf. Mobile Comput.Netw., 2012, pp. 173–184.

[16] R. Kawajiri, M. Shimosaka, and H. Kashima, "Steered crowdsensing:Incentive design towards quality-oriented place-centriccrowdsensing," in Proc.ACM Int. Conf. Ubiquitous Comput., 2014,pp. 691–701.

[17] Z. Feng, Y. Zhu, Q. Zhang, L. Ni, and A. Vasilakos, "TRAC: Truthfulauction for Location-aware collaborative sensing in mobilecrowdsourcing," in Proc. IEEE Conf. Comput. Commun., 2014,pp. 1231–1239.

[18] D. Zhao, X.-Y. Li, and H. Ma, "How to crowdsource tasks truthfullywithout sacrificing utility: Online incentive mechanismswith budget constraint," in Proc. IEEE Conf. Comput. Commun.,2014, pp. 1213–1221.

[19] T. Luo, H.-P.Tan, and L. Xia, "Profit-maximizing incentive forparticipatory sensing," in Proc. IEEE Conf. Comput.Commun.,2014, pp. 127–135.

[20] L. Jaimes, I. Vergara-Laurens, and M. Labrador, "A location-basedincentive mechanism for participatory sensing systems with budgetconstraints," in Proc. IEEE Int. Conf. Pervasive Comput.Commun.,2012, pp. 103–108.

[21] J.-S. Lee and B. Hoh, "Sell your experiences: A market mechanismbased incentive for participatory sensing," in Proc. IEEE Int. Conf.Pervasive Comput. Commun., 2010, pp. 60–68.

[22] X. Zhang, Z. Yang, Z. Zhou, H. Cai, L. Chen, and X. Li, "Free marketof crowdsourcing: Incentive mechanism design for mobilesensing," IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 12, pp. 3190–3200, Dec. 2014.

[23] M. H. Cheung, F. Hou, and J. Huang, "Participation and reportingin participatory

sensing," in Proc. 12th Int. Symp. Model. OptimizationMobile, Ad Hoc, Wireless Netw., 2014, pp. 357–364.

[24] I. Koutsopoulos, "Optimal Incentive-driven design of participatorysensing systems," in Proc. IEEE Conf. Comput. Commun., 2013,pp. 1402–1410.

[25] J. Rula and F. E. Bustamante, "Crowd (soft) control: Movingbeyond the opportunistic," in Proc. 12th Workshop Mobile Comput.Syst. Appl., 2012, pp. 3:1–3:6.LI AND CAO: PROVIDING PRIVACY-AWARE INCENTIVES IN MOBILE SENSING SYSTEMS 1497

[26] K. Tuite, N. Snavely, D.-Y. Hsiao, N. Tabing, and Z. Popovic,"PhotoCity: Training experts at Large-scale image acquisitionthrough a competitive game," in Proc. SIGCHI Conf. Human FactorsComput. Syst., 2011, pp. 1383–1392.

[27] B. Hoh, T. Yan, D. Ganesan, K. Tracton, T. Iwuchukwu, and J.-S.Lee, "Trucentive: A game-theoretic incentive platform for trustworthymobile crowdsourcing parking services," in Proc. IEEE15th Int. Conf. Intell. Transp. Syst., 2012, pp. 160–166.

[28] S. Reddy, D. Estrin, and M. Srivastava, "Recruitment frameworkfor participatory sensing data collections," in Proc. 8th Int. Conf.Pervasive Comput., 2010, pp. 138–155.

[29] Q. Li and G. Cao, "Providing Privacy-aware incentives for mobilesensing," in Proc. IEEE Int. Conf. Pervasive Comput. Commun., 2013,pp. 76–84.
[30] R. Merkle, "Protocols for public key cryptosystems," in Proc. IEEESymp. Security Privacy, 1980, pp. 122–133.

[31] D. Chaum, "Blind signatures for untraceable payments," in Proc.Int. Conf. Adv. Cryptol., 1982, pp. 199–203.

[32] D. Chaum, "Blind signature system," in Proc. Int. Conf. Adv. Cryptol.,1983, pp. 153.

[33] M. Abe and T. Okamoto, "Provably secure partially blindmsignatures," in Proc. Int. Conf. Adv. Cryptol., 2000, pp. 271–286.

[34] [Online]: Available: http://www.gigwalk.com, 2013.
[35] M. Abe and E. Fujisaki, "How to date blind signatures," in Proc.Int. Conf. Theory Appl. Cryptol. Inf. Security: Advances Cryptol., 1996,pp. 244–251.

[36] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," inProc. Int. Conf. Adv. Cryptol., 2004, pp. 41–55.

[37] R. K. Ganti, N. Pham, Y.-E. Tsai, and T. F. Abdelzaher, "Poolview:Stream privacy for grassroots participatory sensing," in Proc. 6thACM C. Embedded Netw. Sens. Syst., 2008, pp. 281–294.

[38] J. Shi, R. Zhang, Y. Liu, and Y. Zhang, "Prisense: Privacy-preservingdata aggregation in People-centric urban sensing systems," inProc. IEEE 29th Conf. Comput. Commun., 2010, pp. 758–766.

[39] Q. Li and G. Cao, "Efficient and privacy-preserving data aggregationin mobile sensing," in Proc. IEEE Int. Conf. Netw. Protocols,2012, pp. 1–10.
[40] Q. Li and G. Cao, "Efficient privacy-preserving stream aggregationin mobile sensing with low aggregation error," in Proc. PrivacyEnhancing Technol. Symp., 2013, pp. 60–81.
\
[41] Q. Li, G. Cao, and T. F. Porta, "Efficient and Privacy-aware dataaggregation in mobile sensing," IEEE Trans. Dependable SecureComput., vol. 11, no. 2, pp. 115–129, Mar/Apr. 2014.

[42] E. De Cristofaro and R. Di Pietro, "Preserving query privacy inurban sensing systems," in Proc. 13th Int. Conf. Distrib. Comput.Netw., 2012, pp. 218–233.

[43] P. P. Tsang, M. H. Au, A. Kapadia, and S. W. Smith, "Blacklistableanonymous credentials: Blocking misbehaving users withoutttps," in Proc. 14th ACM Conf. Comput. Commun. Security, 2007,pp. 72–81.
[44] C. Garman, M. G. 0001, and I. Miers, "Decentralized anonymouscredentials," IACR Cryptology ePrint Archive, 2013.