Anonymity-based Privacy-preserving Data Reporting for Participatory Sensing

Yonglei Yao, Laurence T. Yang, IEEE Senior Member, and Neal N. Xiong, IEEE Senior Member

Abstract—In this paper, we propose an efficient anonymous data reporting protocol for participatory sensing, which provides strong privacy protection, data accuracy and generality. The protocol consists of two stages, namely slot reservation and message submission. In the slot reservation stage, a group of $N$ participants cooperate to assign each member a message slot in a vector which is essentially a message submission schedule, in such a manner that each participant’s slot is oblivious to other members and the application server. In the message submission stage, each participant transmits an encoded data to the application server based on the slot information known only to herself, in such a way that the application server cannot link a data to a specific participant. With such a data reporting protocol, the link between the data and the participants is broken, and as a result, participant’s privacy is protected. We conduct theoretical analysis of the correctness and anonymity of our protocol, as well as experiments to demonstrate the efficiency in small-scale applications with periodic data sampling.

Index Terms—Anonymity, Data reporting, Participatory sensing, Privacy

I. INTRODUCTION

The ubiquitous sensor-rich mobile phones have promoted the emergence of a fast-growing people-centric sensing paradigm, participatory sensing applications [1], and thus have been playing an increasing important role in the evolution of the Internet of Things (IoT). The key idea behind participatory sensing is to use mobile phones as a platform for sensing research, by empowering ordinary citizens to collect and share data sensed from the environments using sensor-equipped phones. Participatory sensing offers a number of advantages over traditional sensor networks, such as greatly reduced cost, unprecedented spatiotemporal coverage, easy deployment, and so on [2].

A typical participatory sensing application follows a client/server architecture, in which data collected by participants’ phones are sent to a central server. The server processes all the data, and produces useful statistics. The statistics may be published in various forms, such as graphical representations or maps showing the sensing results at individual and/or community scale. Based on the sensing scale, participatory sensing applications can be classified into three categories [3]: 1) Personal sensing applications, which are designed for a single individual, such as personal health [4] or sport experiences monitoring [5]; 2) group sensing applications, which are designed for a set of individuals who share a common goal, concern, or interest, such as social media enhancing [6]; and 3) community sensing applications, which conduct large-scale data collection, analysis, and sharing for the good of the community, such as environment [7] and traffic monitoring [8]. Participatory sensing campaigns will revolutionize many sectors of our society, if a large number of participants are willing to submit their data.

On the other hand, participating in a participatory sensing task, especially a community-scale task, could result in private information leakage. Some tasks require users to submit data containing sensitive information, for example, disease symptoms [4]. Some applications don’t directly use sensitive data, but still result in privacy leakage. For example, in a power consumption monitoring application [9], from temporally fine-grained energy consumption reports submitted by users, household activities can be inferred easily. In addition, data in participatory sensing application are usually geo- and time-tagged. From multiple data reported by a participant, an adversary can derive much sensitive information [10]. Thus, users are reluctant to contribute to the sensing campaigns, if their privacy cannot be protected. This would diminish the impact and relevance of sensing campaigns deployed at large scale, as well as limiting the benefits to the users. Therefore, protecting privacy of participants is highly important to the success of participatory sensing applications.

A variety of methods have been proposed to protect the privacy of each participant for participatory sensing applications. A naive mechanism to protect the privacy is to use pseudonyms. However, as demonstrated in [11], the use of pseudonyms does not necessarily guarantee privacy. Some privacy protection methods employ generalization [12, 13] or perturbation [14, 15], both of which intend to allow the application server to determine community trends without revealing individual data, by deliberately reducing the accuracy or precision of the sensed data. Nevertheless, this reduction of data accuracy or precision will inevitably degrade the derived statistics. Researchers also propose privacy protection methods...
for certain applications, such as data aggregation [16], regression modeling [17], or map generation [18]. However, these methods are only applicable in specific applications and lack generality. In addition, with existing methods, if the application server colludes with a global eavesdropper who can monitor the traffic across the network, it can link each data with its contributor.

In this paper, we propose a privacy-preserving data reporting protocol for participatory sensing applications, based on the notion of anonymity. The key intuition is that, if we can break the link between any data and the participant who reports the data, the participant’s privacy can be protected, without degradation of data accuracy. In this way, as long as the data itself does not contain identification information, this is sufficient to protect participants’ privacy. The advantage of our protocol includes: 1) Participants can report original sensed data to the application server, with which the server can produce statistics with the highest accuracy. 2) The protocol is general to all kinds of sensing applications, as the reported data are non-altered original data, the application server can use the data freely for any kind of data processing and produce any statistics. 3) Even the application server colludes with a global eavesdropper, it cannot link any data with its contributor.

Specifically, our anonymous data reporting protocol operates in two stages, namely the slot reservation and data submission. In the slot reservation stage, a group of participants collaborate to construct a vector which contains a set of slots, all with the same length. Each slot contains a reservation message from a participant, which consists of a pseudonym and the length of the data that the participant will submit in the data submission stage. Thus, the vector denotes a schedule that all the participants will follow when submitting data. The vector is constructed in such a way that other than the slot owner, no one knows the owner of the slot. The data submission stage may include multiple rounds. In each round, based on the schedule established in the slot reservation stage, each participant submits a bit stream to the application server, who then XOR all participants’ bit streams together to yield a concatenation of all participants’ raw data, each at its slot. As all the bit streams are encoded with multiple secrets and have the same length, an adversary can’t ascertain which data belongs to which participant.

Our contributions can be summarized as follows:

- We introduce the notion of anonymous data collection for participatory sensing, where “anonymous” indicates unlinkable, i.e., an adversary can’t link any piece of data to the participant who reports that data.
- We present a two-stage protocol for anonymous data reporting, which provides strong privacy, accuracy and generality.
- We propose an anonymous slot reservation schemes for the first stage, and an efficient XOR-based data submission scheme for the second stage of the anonymous data reporting protocol.
- We give theoretical analysis on the correctness and anonymity property of our protocol.
- We provide experimental results to measure the practical efficiency of our solutions, which show that the proposed protocol is applicable to small-scale (with tens of participants) and periodically sampling applications.

The rest of the paper is organized as follows. Section 2 outlines the problem that we address in this paper. Section 3 describes the anonymous slot reservation scheme, and section 4 presents the data submission scheme. Section 5 describes our experimental results. Section 6 summarizes related work, and Section 7 concludes.

II. PROBLEM FORMULATION

A. System Architecture

A typical participatory sensing application follows the Client/Server architecture, as shown in Fig. 1.

![Fig. 1. System Architecture](image)

A sensing application consists of an application server and a large number of participants. Each participant owns a mobile phone equipped with sensors required for the application. With the sensor equipped phones, each participant collects data from the surroundings and periodically reports the sensed data to the application server, which intends to process the data to find useful statistics. The statistics in various forms may be displayed on each participant's mobile phone, or accessed by members of a larger community through web-ports, depending on the application.

We assume that all the participants are formed into groups of size $N$. The application server collects data from each group in a period manner, i.e., round by round. We require that the data collection process should be conducted in an anonymous manner, that is, the application server should get $N$ pieces of data in a message collection round from a group, but should not know which piece came from which group member. In other words, a group is in fact an anonymity set and each member in the group is "hidden" in the $N$ members.

In the sequel, we restrict our discussion to one group, as every group operates in the same manner. We assume that there is a common ordering of all members, and each participant knows this ordering. Without loss of generality, the $N$ members are denoted as $P_1$, $P_2$, ..., $P_N$. The piece of data sensed by $P_i$ in a collection round is denoted as $d_i$. We assume that, $d_i$ does not contain identification information, such as participant’s identifier or telephone number. Thus, so long as the application server can’t observe that $d_i$ comes from $P_i$, her privacy can be reserved.

B. Adversary Model

We assume that there exists a global and passive adversarial client/
eavesdropper who can monitor all the traffic in the network. This powerful eavesdropper is realistic in practice, such as an ISP or a government agency. The global eavesdropper may collude with the application server. We require our protocol to preserve participants’ privacy against such a powerful eavesdropper.

We assume that the application server and all the participants are honest-but-curious. That is, they honestly follow the protocol specification exactly, but may try to learn other participants’ private information. We also assume that, some participants may collude with the application server to de-anonymize the data of honest participants. Our protocol should preserve the honest members’ privacy when the number of colluded member in a group is less than \( N-2 \).

C. Design Goals

In this paper, we aim at designing a data reporting protocol for participatory sensing which satisfy the following requirements:

- **Strong Anonymity**: the protocol provides strong anonymity in a sense that, the application server and/or a global eavesdropper that can monitor all the traffic can’t link a piece of data to the participant who reports the data.

- **Collusion resistance**: the protocol provides collusion resistance that a certain number of colluding members in a group can’t link a data to its owner.

- **Efficiency**: the protocol should be conducted in an efficient manner in a sense that it should not consume much resource of the mobile phones.

In addition, our protocol should enable the application server to produce accurate statistics, and be general enough for any participatory sensing task.

III. ANONYMOUS SLOT RESERVATION

In this section, we present an anonymous slot reservation scheme, which can assign a data submission slot for each participant in an anonymous way. The protocol leverages the idea of message shuffle followed by work in [19, 20, 21, 22].

A. The Scheme

We assume that each participant \( P_i \) has a private/public key pair \((x_i, y_i)\), and each participant knows the public keys of all the others. Thus, all participants can identify each other by public keys. In practice, the participants apply their key pairs from a trusted certification authority whose job is to associate each individual with her public key. Or alternatively, each participant can generate the key pair herself and publish the public key at the application server, if she does not trust any authority.

We use \( C = \{m\}_{y} \) to denote encryption of a plaintext \( m \) with a public key \( y \), and denotes the corresponding decryption with the private key as \( m = dec_y(C) \). In addition, we use \( \{m\}_{y_1,y_N} \) to denote a serial of encryptions under multiple public keys, i.e.,

\[
\{m\}_{y_1,y_N} = \{\ldots \{m\}_{y_N}\}_{y_{N-1}} \ldots \}_{y_2} \}_{y_1}
\]  (1)

When the slot reservation stage starts, each participant \( P_i \) first prepares a slot reservation message which is in the following form

\[
SRM_i = < PN_i, L_i >
\]  (2)

Where \( PN_i \) is a pseudonym generated in a random manner, and \( L_i \) is the length of the data that \( P_i \) will report in the data submission stage.

Note that, all the reservation messages have the same length. All the participants and the server have an agreement on the format and length of this message. \( P_i \) then encrypts her slot reservation message with public keys of all the members, i.e.,

\[
C_i = \{SRM_i\}_{y_1,y_N}
\]  (3)

Each participant sends its encrypted slot reservation message to the first participant \( P_1 \) (Recall that, there is a predefined ordering of all members, as mentioned above).

\( P_1 \) collects all the \( N \) encrypted slot reservation messages and obtains a vector \( C_1 = < C_1, C_2, \ldots C_N > \). \( P_1 \) permutes \( C_1 \), strips off one layer of encryption of each element using her private key \( x_1 \), and obtains another ciphertext vector \( C_2 = < C_{\pi_1(1)}, C_{\pi_1(2)}, \ldots, C_{\pi_1(N)} > \), where \( \pi_1 \) is a permutation on \(<1, 2, \ldots, N>\). Then \( P_1 \) sends \( C_2 \) to \( P_2 \).

This process proceeds at each participant \( P_i (i< i< N) \) in a step by step manner: receiving a ciphertext vector \( C_i \), reordering the vector using a random permutation \( \pi_i \), stripping off one layer of encryption of each element, and finally sending \( C_{i+1} = < C_{\pi_i(1)}, C_{\pi_i(2)}, \ldots, C_{\pi_i(N)} > \) to the successor. Finally, \( P_N \) strips off the final layer encryption of all the ciphertexts and sends the result to the application server.

Thus, the application server obtains a slot reservation messages vector, and each message is in the form of \(< PN_i, L_i >\). The server then publishes the vector. All the participants can access the vector. If every participant finds her slot reservation message in the vector, then the slot reservation stage complete successfully. Or if some participant can’t find her reservation message, she will give an error message, and the slot reservation process may run again.

Fig. 2. Illustration of a slot reservation

\[
\text{Anonymous Slot Reservation}
\]

\[
P_1 \quad P_2 \quad P_3
\]

\[
\text{PN}_1 \quad L_1 \quad \text{PN}_2 \quad L_2 \quad \text{PN}_3 \quad L_3
\]
An illustration of slot reservation for a 3-member group with a final permutation <2, 3, 1> is shown in Fig. 2. By this slot vector, \( P_1 \) knows that her slot starts from the \( L_3+1 \) bit, with length of \( L_1 \) bits.

B. Analysis

1) Correctness

If all the participants exactly follow the slot reservation scheme, the application server will get a permutation of all the slot reservation messages.

Each participant encrypts its slot reservation message with the public keys of all the participants in a serial manner, and sends the message to \( P_1 \). As a result, \( P_1 \) obtains a permutation of all encrypted messages. Then each participant processes the encrypted message vector in the same manner, by applying only two operations: permuting the message vector, and stripping one layer of encryption of each message. So when all the \( N \) participants have processed the messages, the total \( N \) layers of encryption of each message are all stripped off. Finally, the application server will get a permutation of all the slot reservation messages in plaintext.

2) Anonymity

The slot reservation scheme will preserve the anonymity of each honest participant against the application server and at most \( N-2 \) corrupted participants in the semi-honest model, assuming that the underlying public key encryption scheme is semantic secure.

Without loss of generality, we assume that the honest participants are \( P_1, P_2, \ldots, P_m \) \((m>1)\), and let \( \pi \) be an arbitrary permutation on \(<1, 2, \ldots, m>\). Informally, anonymity means that the adversary, i.e., the application server with the help of dishonest participants, cannot determine which reservation data comes from which honest participant. This means that, the server cannot tell the difference between a slot reservation process in which the honest participants’ inputs are \((M_1, M_2, \ldots, M_m)\) and another reservation process in which the honest participants’ inputs are \((M_{\pi(1)}, M_{\pi(2)}, \ldots, M_{\pi(m)})\). Formally speaking, this means the views of the adversary on two different slot reservation data, \( \text{VIEW}(M_1, M_2, \ldots, M_m) \) and \( \text{VIEW}(M_{\pi(1)}, M_{\pi(2)}, \ldots, M_{\pi(m)}) \), are computational indistinguishable [23]. We prove this by contradiction.

We decompose the permutation into a number of primitive permutations, each of which only exchange the order of two reservation data of honest participants. Thus, we have \( \pi = \pi_1 \pi_2 \ldots \pi_m \). Define

\[
\text{VIEW}_0 = \text{VIEW}(M_1, M_2, \ldots, M_m)
\]

and for \( 1 \leq j \leq m \)

\[
\text{VIEW}_j = \text{VIEW}(M_{\pi_1 \pi_2 \ldots \pi_{j-1}(1)}, \ldots, M_{\pi_1 \pi_2 \ldots \pi_{j-1}(m)})
\]

Then

\[
\text{VIEW}_m = \text{VIEW}(M_{\pi(1)}, M_{\pi(2)}, \ldots, M_{\pi(m)})
\]

Suppose that the slot reservation scheme cannot preserve the honest participants' anonymity, which means that the two different views of the adversary \( \text{VIEW}(M_1, M_2, \ldots, M_m) \) and \( \text{VIEW}(M_{\pi(1)}, M_{\pi(2)}, \ldots, M_{\pi(m)}) \) can be distinguished by an efficient algorithm with a non-negligible probability. So there must exist a \( j \in [1, m] \) such that \( \text{VIEW}_j \) and \( \text{VIEW}_{j+1} \) are also distinguishable with a non-negligible probability. AS \( \pi_j \) only exchanges two reservation data, this means that views on two slot reservation data vectors with only two elements’ positions exchanged are distinguishable. As a result, we can distinguish the ciphertexts of two different plaintexts with a non-negligible probability, which contradicts the semantic security of the underlying encryption. This ends the proof.

In addition, the protocol can preserve anonymity even though the application server colludes with a global eavesdropper that can monitor the communication pattern of each participant. This is because, the volume of traffic at each participant is the same, making the tracking attack infeasible.

3) Efficiency

The computation overhead mainly includes public key encryption and decryption. To generate a slot reservation message, each participant needs to apply encryption \( N \) times. To process the slot reservation message vector, each participant needs to decrypt \( N \) messages. Thus, the total computation cost for each participant is \( O(N) \) encryptions and \( O(N) \) decryptions.

In addition, as all the participants need to process the slot reservation message vector, each participant will receive and send \( N \) messages, which results in \( O(N) \) communication cost for each participant.

4) Pseudonym collision

Since each participant randomly chooses pseudonym which is independent of other participants’ choices, there may be a chance that two participants choose the same pseudonym. As a result, pseudonym collision arises. This will result in an incorrect data submission schedule. Following we show that, by choose appropriate pseudonym length, the probability is very small.

![Fig. 3. Probability of pseudonym collision](image-url)
Let $L$ denote the length of pseudonym in bits. The probability that there is no collision between pseudonyms is

$$\frac{2^L (2^L - 1) \ldots (2^L - N + 1)}{(2^L)^N}$$

(7)

Thus, the collision probability is

$$1 - \frac{2^L (2^L - 1) \ldots (2^L - N + 1)}{(2^L)^N}$$

(8)

Figure 3 shows the collision probability for different group size and pseudonym length. For a small group that includes tens of participants, if we set the length of pseudonym to be 32 bits, the collision probability is near zero (in fact, it is bellow 0.0005%). Thus, the success probability of slot reservation is near 100%.

IV. ANONYMOUS DATA SUBMISSION

The data submission may include multiple rounds. For example, based on the slot reservation vector in Fig.2, the data submission schedule is as the following:

```
<table>
<thead>
<tr>
<th>d3</th>
<th>d1</th>
<th>d2</th>
<th>d3</th>
<th>d1</th>
<th>d2</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round 1</td>
<td>Round 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Fig. 4. Illustration of Data Submission round

In each round, each participant has a data $d_j$ to report. To achieve this anonymously, each participant $P_i$ prepares a bit stream with length $\sum_{j=1}^{N} L_j$ bits, and encodes the data $d_i$ in her slot. After collecting all the $N$ data pieces, the server XORs them all and obtains all the data.

A. The Scheme

We build the anonymous data submission scheme on the basic idea of DC-nets [24]. In classic DC-nets, an anonymous sender in a group wants to share a message with other members. To do so, every pair of members first establishes a pairwise shared secret. Then each member XORs all the secrets she shares, while the anonymous sender additionally shares a secret. Then each member XORs them all and obtain all the data.

Our scheme leverages the efficient XOR operation used in DC-net. In addition, as all the participants have anonymously established a message submission schedule, all the participants can report their data at the same time without collision.

We assume that each pair of participants shares a common secret seed $S_{ij}$, and note that $S_{ji} = S_{ij}$. The scheme also uses a pseudorandom function (PRF). We use $PRF[L, S]$ to denote the most significant $L$ bits generated from a seed $S$ using the PRF.

Without loss of generality, we assume that a slot reservation vector is as follows

```
PN_1 L_1 PN_2 L_2 ... PN_{N-1} L_{N-1} PN_N L_N
```

Suppose participant $P_i$ has a data $d_i$ and slot $(PN_i, L_i)$. For each slot $j$, $P_i$ generates $N-1$ pseudorandom bit streams, all with length $L_j$ and based on the seed shared with other participants:

$$c'_i[k] = PRF(L_j, S_{ik}, || nonce_j), k \neq i$$

(9)

Where $||$ denotes concatenation and $nonce_j$ is a random number which depends on the slot and the round. All the participants agree on the rule of generating the nonce. For example, a simple rule is that, nonce is the concatenation of $PN_j$ and the round number. This nonce can assure that different pseudorandom bits will be used for different submission slots and different rounds.

Then $P_i$ generates a bit stream with length $L_i$ for slot $j$:

$$c'_i = c'_i[1] \oplus c'_i[i - 1] \oplus c'_i[i + 1] \ldots \oplus c'_i[N]$$

(10)

$P_i$ hides its data $d_i$ in her own slot, i.e.,

$$c'_i = c'_i \oplus d_i$$

(11)

$P_i$ now concatenate all the ciphertexts for each slot and generates the final ciphertext, i.e.,

$$C_i = c'_i || c'_2 || \ldots || c'_N$$

(12)

Finally, $P_i$ sends the ciphertext to the application server. After receiving the $N$ bit streams, the application server XORs all of them and obtains a concatenation of all the data items, i.e.,

$$M = C_1 \oplus C_2 \ldots \oplus C_N$$

(13)

Fig. 5 shows an illustration of data submission in the first round for a group of three participants, based on the data submission schedule in Fig.4, assuming that the nonce is the concatenation of the pseudonym and the round number.

```
<table>
<thead>
<tr>
<th>d3</th>
<th>d1</th>
<th>d2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRF[S_{i1}, PN_i, L_i]</td>
<td>PRF[S_{i2}, PN_i, L_i]</td>
<td>PRF[S_{i3}, PN_i, L_i]</td>
</tr>
</tbody>
</table>
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
| PRF[S_{i1}, PN_i, L_i] | PRF[S_{i2}, PN_i, L_i] | PRF[S_{i3}, PN_i, L_i] |
  +   +   +
```

Fig.5. Illustration of a single round data submission
B. Analysis

1) Correctness

If all the participants follow the data submission scheme exactly, the application will get all the data that the participants sensed.

Without loss of generality, we consider the message slot \( <P_{N}, L_{i}> \), and assume that the owner is \( P_{i} \).

The slot owner \( P_{i} \) uses all the seeds that she shares with all the other participants to generate the cipher text for her slot as follows

\[
c_{i} = d_{j} \oplus PRF(L_{i}, S_{j} || nonce_{j}) \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \qu
because the slot reservation message is small. In the experiments, we observe that, with \( N \) participants in a group, the size of the slot reservation message vector in ciphertexts is about \( N \) KB. Thus, each participant can process it quickly. Of course, as the slot reservation message must be processed by all participants in a one-by-one manner, the application server will wait for a longer time (about 1 minute) to get the final data submission schedule. We also observe that, the length of the pseudonym has no effect on the latency. The reason is that, each participant encrypts her message using keys of all participants in a serial manner. Thus, no matter what the length of plaintext message is, after the first encryption, the length of the ciphertext message is 1024 bits. As a result, each participant will process a message vector in ciphertexts with length \( N \times 1024 \) bits. The same rationale also applies to the energy, as shown in Fig. 7.

The energy of the phone battery is measured using a multi-meter. We first run the slot reservation scheme for 40 times (about 1 minute running), and measure the energy use. Then we measure the base energy use, i.e., without running of the slot reservation scheme, for the same time period as 40 times running of slot reservation. The net energy use of 40 times running of slot reservation is computed by subtracting the base energy use from the energy use when slot reservation was running. Finally, we divide the net energy use by 40, and use it to approximate the energy consumption of a single running of slot reservation.

We measure the energy consumption of our slot reservation scheme, and compare it with that of the scheme based on \( (N, N)\)-SS protocol in [25]. As Fig. 7 shows, our slot reservation scheme incurs little energy overhead. Even for a group with 40 members, the energy consumed for slot reservation is only about 2.1J. For a 1500mAH battery that contains about 20000J energy, this energy overhead is negligible. Our scheme consumes much less energy than the scheme based on \( (N, N)\)-SS protocol, as the latter is a communication-intensive process. Specifically, in the slot reservation based on \( (N, N)\)-SS protocol, each member of an \( N \)-group will send and receive at least \( 2(N - 1)M \left\lfloor \log_2 N \right\rfloor \) bits, where \( M \) is the larger of the two values \( 361 + N \) and \( 2N^2 - 2N \), and as a result, consume a lot of energy. Note that, we do not consider collision. However, if collision is considered, the scheme based on \( (N, N)\)-SS protocol will consume much more energy, because to resolve a collision, even more bits need to be transferred than in slot reservation.

### B. Efficiency of Data Submission

Next, we evaluate the efficiency of the data submission scheme in terms of latency and energy overhead at a participant.

In the experiments, we investigate the effect of group size and data size on the efficiency. For simplicity, we assume that all the data from different participants have an equal length. We increase the group size from 10 to 40 with a step increment of 5. For each group size, we vary the data size from 0.5 KB to 512 KB. Fig. 8 shows the latency results.

From Fig. 8 we can see that, when data is small, i.e., less than 8 KB, the latency at each participant is less than 2.5s. However, when the data size reaches 512 KB, the latency will increase dramatically. For a group size of 40, the latency is nearly 90s. We notice that latency in data submission is much longer than in slot reservation. This is because the main task in slot reservation is computation, i.e., public key encryption and decryption, and the plaintext is small (hundreds of bits), while data submission is a communication-intensive process and the data size is much larger than the slot reservation message. As a result, the latency in data submission is much longer.

![Fig.7. Energy of Slot Reservation at a participant](image1)

![Fig.8. Latency of Data Submission at a participant](image2)
The energy overhead of data submission is shown in Fig.9. Again, the variables are the group size and the data length.

From Fig.9, we can see that if the data length is not more than 32 KB, the energy overhead per participant is relatively small. With a 512 KB data, the energy overhead is relatively large, i.e., about 25 J. We also can see that, the energy overhead of data submission is heavier than that of slot reservation. The reason is that, data submission is a communication-intensive process, while slot reservation is computation-intensive one. However, communication is much more energy-hungry than computation, as confirmed by [26].

We also compare the energy consumption of our data submission scheme and the scheme based on (N,N)-SS protocol in [25]. We set the message size to be 2 KB, and use AES-128 as the symmetric key cryptosystem used in the scheme based on (N,N)-SS protocol.

![Fig.9 Energy of Data Submission at a participant](image)

As the table shows, our data submission scheme consumes orders of less energy than the scheme based on (N,N)-SS protocol. The reason is that, in the latter, each member submits with all the other members using (N,N)-SS protocol. The reason is that, each member shares each element in the message vector in the slot applied in the slot reservation. Then each member shares each element in the message vector with all the other members using (N,N)-SS. The total communication cost for each member is N(N−1)(r+h)+N(r+h) bits sending and N(N−1)(r+h)+N(r+h) bits receiving, where r is the message length and h is the key size. As a result, the communication overhead is very heavy and a lot of energy will be used.

### TABLE I

<table>
<thead>
<tr>
<th>Group Size</th>
<th>Our Scheme</th>
<th>Data Submission based on (N,N)-SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1.75</td>
<td>91.65</td>
</tr>
<tr>
<td>15</td>
<td>2.21</td>
<td>209.83</td>
</tr>
<tr>
<td>20</td>
<td>2.53</td>
<td>376.25</td>
</tr>
<tr>
<td>25</td>
<td>3.12</td>
<td>590.91</td>
</tr>
<tr>
<td>30</td>
<td>3.54</td>
<td>853.80</td>
</tr>
<tr>
<td>35</td>
<td>3.91</td>
<td>1164.94</td>
</tr>
<tr>
<td>40</td>
<td>4.14</td>
<td>1524.31</td>
</tr>
</tbody>
</table>

In this section, we summarize the most relevant existing research.

### VI. RELATED WORK

Privacy in participatory sensing systems has been addressed by many works, e.g., [12-18]. In [12], instead of sending an accurate location to the server, spatial cloaking is employed to calculate an anonymity set. As a result, the user is protected. The work in [13] also follows this idea of generalization, in which k pieces of data are combined together before sending to the server, with the intention of adding enough “confusion” in the data to make it difficult to obtain exact times and locations for the individual data.

Another line of work employs the idea of perturbation, with the intention of determining community trends and distributions without revealing individual data. In [14], a noise with characteristics similar to a realistic data set is generated and then added to the data at the mobile phone side. As the statistical characteristics of the noise are known, the server can estimate the community trends and distribution by subtracting the average noise time series from the sum of all individual perturbed data. In [15], a scheme using empirical negative surveys is proposed, in which the probability distributions for perturbation vary according to the data distributions of public information. In [16], by combining ring-based interleaved grouping and data perturbation, the authors propose a scheme which can allow the server to obtain an aggregate with low error and protect privacy of each node. However, both generation and perturbation reduce the accuracy of sensed data, and as a result degrade the quality of statistics. In our approach, the application server can obtain the original raw data, and thus can produce high quality statistics.

Some researchers focus on specific applications and propose privacy protection measures, e.g., work in [17, 18]. In [17], the authors focus on regression modeling. A series of data transformation and aggregation operations at the participatory nodes is launched, in such a way that the server can learn a model from the data collected from participants, without any need of the original private data. The work in [18] focuses on map generations and user trace privacy. Two novel techniques are employed: The first is to limit the number of points of a trace reported to the server, and the second is to shuffle the points of a trace. Thus, the server can’t recover a trace of an individual user. However, these protocols are proposed for specific applications, and as a result lack generality. In contrast, the application server in our protocol can obtain raw data and produce any type of statistics. Thus, our protocol can be applied to any participatory sensing application and thus provides great generality and

In addition, in current privacy protection approaches, if the application server colludes with a powerful global eavesdropper that can monitor the traffic across the network, it can link data with its contributor. The reason is that, the message that each participant sends may have different length and pattern, and as a result, the eavesdropper can track the traffic of each participant and de-anonymize the data. In our protocol, both in the slot reservation and data submission stage, the messages sent by all the participants have the same length.
Hence, our protocol can provide anonymity against a global passive eavesdropper.

B. Anonymous Communication

The anonymous data collection problem for data mining was first proposed in [19]. In this work, the author proposed the idea of message shuffle, in which $t$ out of $N$ respond are chosen as leaders and responsible for data reported by all the members. The re-randomization property of the ElGamal encryption scheme is exploited. Each data is encrypted by the owner’s public key, and then shuffled and re-encrypted by each leader. In [20], the authors show that the anonymity in [19] will be violated if the last leader colludes with the collector, and propose a collusion-resistant scheme. In their scheme, each user own two pair of keys, namely, the primary and the secondary private/public key pair. Data of each user is first encrypted using the collector’s public key, then encrypted using each member’s secondary public key, and finally encrypted using each member’s primary public key. In both schemes, the data submission stage is also based on public encryption, which lack efficiency.

In [21, 22], the data collection is divided into two stages, the slot reservation and the bulk transfer. A cloud-like architecture is adopted in which a set of dedicated anonymity server shuffle data from users. However, in practice, we doubt the existence of reliable and trustful third party servers.

The most efficient anonymous message protocol so far is proposed in [25], where a $(N, N)$-SS secret sharing protocol is employed for both slot reservation and data submission. The computation is efficient, as it doesn’t use public key encryption. But the communication cost is very heavy. Specifically, in slot reservation, the communication cost of each member in an $N$-group is at least $4(N-1)M \left\lceil \log_2 N \right\rceil$ bits where $M$ is the larger of the two values $361 + N$ and $2N^2 - 2N$. In data submission, to send a $r$-bit message, the communication cost of each member is $2(N-1)(r+h)+2(r+h)$ bits, where $h$ is the length of the encryption key. Thus, this protocol is suitable for very small messages, e.g., integer numbers.

VII. CONCLUSION

Privacy protection is an important issue in participatory sensing. We propose an anonymous data reporting protocol for participatory applications to protect user privacy. The intuition behind the protocol is that, if the data itself does not contain identification information, and we can break the link between the data and the participant that reports the data, the user’s privacy can be protected. The anonymous data reporting protocol is divided into two stages, a slot reservation stage and a data submission stage. We propose an anonymous slot reservation scheme based on public key encryption and message shuffle, and a data submission scheme based on efficient XOR operation. The theoretical analysis verifies the correctness and the anonymity of the protocol. The experiments demonstrate that, for small-scale applications with only tens of participants where data is collected in a periodic manner, the proposed protocol is efficient and applicable.
Symposium on Operating Systems Design and Implementation (OSDI), 2012; pp. 179-192


**Yonglei Yao** received his Ph.D in computer science in 2007 from Beijing University of Posts and Telecommunications. He is currently an associate professor at the School of Computer and Software, Nanjing University of Information Science & Technology. His research interests include privacy and security in wireless networks, Wearable Devices and pervasive computing.

**Dr. Laurence T. Yang** is a professor in computer science at St Francis Xavier University, Canada. His research includes high performance, embedded and ubiquitous/pervasive computing. He has published around 300 papers (including around 100 international journal papers such as IEEE and ACM Transactions, SCI/EI indexed around 292 papers) in refereed journals, conference proceedings and book chapters in these areas. He has been involved in more than 100 conferences and workshops as a program/general/steering conference chair and more than 300 conference and workshops as a program committee member. He served as the vice-chair of IEEE Technical Committee of Supercomputing Applications (TCSA) until 2004, currently is the chair of IEEE Technical Committee of Scalable Computing (TCS), the chair of IEEE Task force on Ubiquitous Computing and Intelligence. He is also in the steering committee of IEEE/ACM Supercomputing conference series and National Computing Allocation Committee of Canada.

Dr./Prof. Yang is the editors-in-chief of several international journals. He is serving as an editor for around 20 international journals. He has been acting as an author/co-author or an editor/co-editor of 25 books. He has won 5 Best Paper Awards; 4 IEEE Best Paper Awards; 2 IEEE Outstanding Paper Awards; one Best Paper Nomination; Distinguished Achievement Award, 2005; IEEE Distinguished Achievement Award, 2009; Canada Foundation for Innovation Award, 2003; He has been invited to give around 20 keynote talks at various international conferences and symposia.

**Neal N. Xiong** is current a Professor at School of Computer Science, Colorado Technical University, Colorado Spring, CO, USA. He received his both Ph.D degrees in Wuhan University (about software engineering), and Japan Advanced Institute of Science and Technology (about dependable networks), respectively. Before he attends Colorado Technical University, he worked in Wentworth Technology Institute, Georgia State University for many years. His research interests include Cloud Computing, Security and Dependability, Parallel and Distributed Computing, Networks, and Optimization Theory.

Dr./Prof. Xiong published over 100 international journal papers and over 100 international conference papers. Some of his works were published in IEEE JSAC, IEEE or ACM transactions, ACM Sigcomm workshop, IEEE INFOCOM, ICDCS, and IPDPS. He has been a General Chair, Program Chair, Publicity Chair, PC member and OC member of over 100 international conferences, and as a reviewer of about 100 international journals, including IEEE JSAC, IEEE SMC (Park: A/B/C), IEEE Transactions on Communications, IEEE Transactions on Mobile Computing, IEEE Trans. on Parallel and Distributed Systems. He is serving as an Editor-in-Chief, Associate editor or Editor member for over 10 international journals (including Associate Editor for “IEEE Tran. on Systems, Man & Cybernetics: systems” and “Information Science”; and Editor-in-Chief for Journal of Parallel & Cloud Computing (PCC) and International Journal of Security and Its Applications (IJISIA)), and a guest editor for over 10 international journals, including Sensor Journal, WINET and MONET. He has received the Best Paper Award in the 10th IEEE International Conference on High Performance Computing and Communications (HPCC-08) and the Best student Paper Award in the 28th North American Fuzzy Information Processing Society Annual Conference (NAFIPS2009).

Dr./Prof. Xiong is the Chair of “Trusted Cloud Computing” Task Force, IEEE Computational Intelligence Society (CIS), http://www.cs.gsu.edu/~cscnxx/index-TF.html, and the Industry System Applications Technical Committee, http://ieee-cis.org/technical/isatc/; He is a Senior member of IEEE Computer Society, E-mail: nxiong@coloradotech.edu.