Privacy-Preserving Data Aggregation in Participatory Sensing Networks

Sarah M. Erfani, Shanika Karunasekera, Christopher Leckie, Udaya Parampalli

NICTA Victoria Research Laboratory
Department of Computing and Information Systems
University of Melbourne, Australia
email:{sarahme, shanika, caleckie, udaya}@csse.unimelb.edu.au

***Abstract*—Participatory sensing using mobile devices is emerging as a promising method for large-scale data sampling. A critical challenge for participatory sensing is how to preserve the privacy of individual contributors' data. In addition, the integrity of the data aggregation is vital to ensure the acceptance of the participating sensing model by the participants. Existing approaches to these issues suffer from excessive communication cost, long delays or rely on a trusted third party. The objective of our research is to design a data-aggregation scheme for participatory sensing systems that addresses *user privacy* and *data integrity* while keeping *communication overhead* as low as possible. We propose four techniques to address these challenges and validate them through analytical models and simulations.**

I. Introduction

Participatory sensing is an emerging field in sensing applications, which employs sensors embedded in mobile devices such as smart phones to enable users to monitor, share and learn from their surrounding environment. Due to the personal nature of the data shared in Participatory Sensing Networks (PSNs), their success depends on the goodwill of users to contribute their data. Consequently, a key challenge in PSNs is how to ensure the privacy of users [1]. Since sensor readings may contain sensitive information concerning a user's private life, any lack of confidence in the privacy of an individual's data will prevent participants from contributing or providing faithful observations. Hence, an essential precondition to the success of these networks is maintaining participants' privacy.

In many PSN applications, aggregation queries allow the system to aggregate individuals' measurements in a way that provides exact summary results while hiding personal information. An open challenge in this context is how to aggregate raw data from users when the aggregator is untrusted. A practical solution for this issue has been proposed by [2], in which *users' privacy* is preserved based on a mutual protection approach called data slicing. In this approach, sensed values are split in to "slices" and distributed among neighbours before being transmitted to the server. Acting as intermediate aggregators, neighbours partially aggregate the received data slices and then forward the result to the server. While this scheme ensures that captured sensor measurements are likely to remain private, there is potential for the intermediate aggregators to make inferences about their neighbours or the whole network. For example, consider the case of a group of participants who are on a low-cal diet. Even receiving a slice of measured calories with a greater value than the defined limit reveals that the corresponding neighbour did not follow the diet plan.

A further challenge that arises from sharing data slices according to this scheme is that there is no way to guarantee that the collected sensor readings are trustworthy. If malicious nodes in the network modify other participants' data, then the integrity of the system can be compromised. Additional, factors like delay or collisions over wireless network channels may cause messages to be lost or corrupted, thus degrading the integrity of the aggregation results from the PSN.

To the best of our knowledge, the latter issue, *data integrity*, has not been addressed in the existing research on aggregation in PSNs. The lack of a comprehensive method for data aggregation in PSNs that simultaneously ensures user privacy and integrity motivates our work.

To address the above-mentioned issues, we adapt a secret perturbation scheme [3] for use in the PSN architecture, shown in Fig. 1a, to achieve the following objectives.

- *User Privacy*: We require that the sample measurements of a source node are not revealed to any other entity in the network. A similar requirement applies to fully or partially aggregated data, that they should only be accessible to the aggregation server.
- *Data Integrity*: We require that the aggregated value at the aggregation server should be equal to the sum of the original data sensed by the participating source nodes.
- *Efficiency*: While privacy and integrity are requirements of data aggregation, we also require that the system complexity should be kept as low as possible.

II. Related Work

Since the idea of participatory sensing was first introduced in [4], it has recently become an active topic of research. A broad overview on the importance of PSNs, their challenges and opportunities are illustrated in [5], [6], [7]. Two major issues for data aggregation in participatory systems are the privacy of personal information and data accuracy. Although some similar issues have been addressed in WSNs [8], [3], [9], participatory sensing systems inherit a complex communication environment. Accordingly, with dynamic architectures, untrusted aggregators and the personal content of queries, PSNs cannot directly employ such solutions.

Privacy - In contrast to the large number of studies on privacy-preserving data aggregation conducted in WSNs, it is largely an open problem in PSNs.

The closest works to our focus on privacy-preserving data aggregation are [2] in participatory networks, and [3], [9] in WSNs. In SMART [9], user privacy is addressed by slicing each data measurement randomly and relying on neighbours to transmit data slices. Recently, PriSense [2] applied SMART to PSNs, and studied the effectiveness of neighbour selection for data sharing.

A promising approach for secure data aggregation is CMT [3], which is an efficient and provably secure additive homomorphic stream cipher that provides effective encrypted data aggregation. However, there are some disadvantages associated with CMT. First, source nodes need to send their non-aggregatable ID to the server, thus increasing the communication cost. Second, the additive homomorphic property of CMT means that a malicious participant can simply add fraudulent data to the content of a message. Finally, it does not prevent a malicious server seeking to disclose participants' records.

A number of studies have attempted to address some of these deficiencies of the CMT secret perturbation scheme. For example, in [10] to avoid sending user IDs, it is assumed that all the nodes are participating. Li and Cao[11] applied CMT to mobile sensing networks with a dynamic architecture and a malicious server. However, the authors rely on a trusted third party for assigning secrets to users, which may be a restrictive assumption in a real world scenario.

Privacy and Integrity - Privacy and integrity are still open challenges for PSNs. The authors in [10] adopt a secret sharing scheme for integrity assurance, in which the system is required to have background knowledge about the nodes that are willing to contribute. This might be a feasible assumption in WSNs with a more static environment, and where it can be assumed that all nodes are participating. However, these assumptions are too restrictive for PSNs.

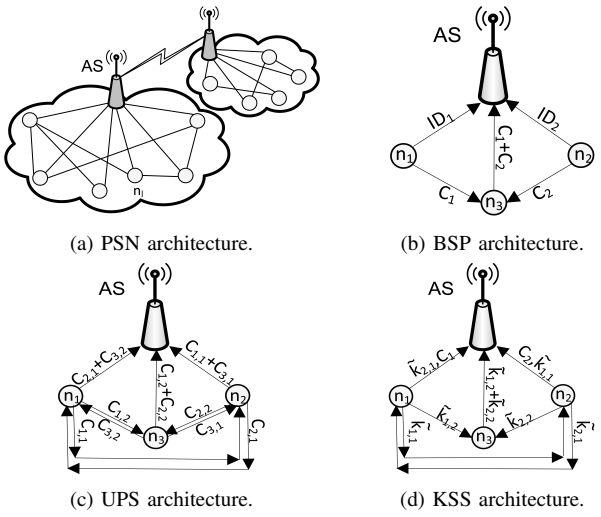
III. Preliminaries and Problem Statement

In this section, we outline the network architecture, query model and security assumptions for our problem statement.

A. Participatory Sensing Architecture

We consider the case of a participatory sensing network that comprises an aggregation server (AS) and a set of participating mobile user nodes $\mathcal{N} = \{n_l | l = 1, \ldots, N\}$, as can be seen in Fig. 1a. Similar to the architecture in [12], we assume that the AS transmits packets via one hop to the N nodes, and the nodes (e.g., using WiFi or cellular) can communicate directly with the AS. Nodes within the network can also form a wireless ad hoc network with each other, so that they can communicate with their neighbours, e.g., using AODV [13] for route discovery. For ease of understanding, in this paper we assume the network has a flat architecture. However, our schemes can be applied to hierarchical architectures as well.

Without loss of generality, we describe the case of a single measurement variable at each node. The AS issues a query Q to the nodes $\mathcal{N}$, and computes the aggregation value of the measurement values returned by those nodes that choose to respond to the query or have a valid value satisfying Q. Let $\mathcal{SN} \in \mathcal{N}$ denote the set of nodes that report to the AS query, such that $\mathcal{SN} = \{sn_i | i = 1, \ldots, SN\}$, and let D_i denote the measurement value returned by sn_i, where D_i is an integer.



(a) PSN architecture. (b) BSP architecture.

(c) UPS architecture. (d) KSS architecture.

Fig. 1 Architectures of PSN and aggregation schemes.

B. Aggregation Query Model

In this paper, we consider the case of the widely used exact summation query, where the AS is to compute $SUM = \sum_{i=1}^{SN} D_i$. We also expect that the AS is able to determine the number of source nodes SN that have provided their measurement value. In this way, the participatory network is able to compute a range of queries, such as `MEAN, AVERAGE` or `STANDARD DEVIATION`.

C. Security Model

Our aim is to ensure the integrity of the data aggregation process in response to a query, while ensuring the privacy of the participant source nodes. In particular, our aim to minimise the possibility that a source node sn_i can be associated with its measurement data D_i by any third party eavesdropper, the AS or any other node $n_l \in \mathcal{N} \backslash \{sn_i\}$. In this section we summarise the main security assumptions we have made and the type of threat model that we address.

Security Assumptions - Each node n_l is assigned an identifier ID_l and a private key K_l that is shared with the AS. Each pair of neighbouring nodes can derive their own symmetric key, so that any communication between that pair of nodes can be encrypted to prevent eavesdropping attacks. The allocated key is temporal and is valid as long as the node is in the network (or active). Each node is also given a list of all available neighbouring nodes in the network, with which it can communicate, following the approach proposed in [12].

Threat Model - We consider that no entity can be guaranteed to be trustworthy, including both the AS and the nodes. In the case of the AS, we consider it to be an "honest-but-curious" adversary, i.e., it is interested in the correct aggregation results,

but may want to violate the privacy of a source node by being able to associate a source node sn_i with its measurement data D_i. In the case of a node, we assume it can exhibit two types of malicious behaviour: *a*) it can try to violate the privacy of a neighbour by attempting to obtain the measurement data D_i collected from a neighbour sn_i, *b*) if used to aggregate values from other nodes, it may want to manipulate the result of aggregation to produce an incorrect result.

IV. Our Schemes for Privacy Preservation

To achieve privacy for exact `SUM` queries in participatory networks, we adopt the basic ideas of secret perturbation [3] and data splitting [9]. In the following four subsections, we build different schemes based on these basic ideas to meet our desired objectives for PSNs. We refer to these schemes as: *(A)* Basic Secret Perturbation, *(B)* Universal Participation, *(C)* Key Splitting, and *(D)* Key Splitting with Integrity.

A. Basic Secret Perturbation Scheme (BSP)

The idea of secret perturbation [3] is based on additive homomorphic encryption. Let $D \in [0, \mathcal{M}-1]$ denote a measurement value, where $\mathcal{M}$ is a large integer, and let K denote a key. The encryption of D using K gives a ciphertext $C = Enc(D, K, \mathcal{M}) = (D + K) \, mod \, \mathcal{M}$, where the decryption gives $D = Dec(C, K, \mathcal{M}) = (C - K) \, mod \, \mathcal{M}$. Using this scheme, it can be shown that if $C_1 = Enc(D_1, K_1, \mathcal{M})$ and $C_2 = (D_2, K_2, \mathcal{M})$ then $Dec(C_1 + C_2, K_1 + K_2, \mathcal{M}) = D_1 + D_2$, i.e., the encryption is homomorphic with respect to addition.

We can use secret perturbation in participatory sensing as follows. Each source node sn_i has a measurement value D_i and a secret key K_i that is shared with the aggregation server AS. The source node sn_i also has an identifier ID_i that is assigned by the AS. If the source node sn_i wants to participate in a query it takes the following steps:

i. Encrypt the observation $C_i = (D_i + K_i) \, mod \, \mathcal{M}$
ii. Notify the AS of the participation of sn_i by sending ID_i directly from sn_i to the AS
iii. Send C_i to a neighbour node of sn_i for aggregation

A source node selects a neighbour randomly from the list of available neighbours and forwards its data via the neighbour to the AS. We refer to these randomly chosen neighbours as a *cover node* (*cn*) [2] of sn_i. Although all nodes can communicate directly to the AS, a cover node helps to protect the user's privacy. As shown in Fig. 1b, when a node n_3 receives perturbed observations (say, C_1 and C_2) from its neighbours (sn_1 and sn_2), it transmits the summation $C_1 + C_2$ to the aggregation server AS. Thus, the AS receives $C_1 + C_2$ and separately ID_1 and ID_2, from which it infers that it must use K_1 and K_2 in decryption. The final aggregation $D_1 + D_2$ can then be determined as $D_1 + D_2 = Dec(C_1 + C_2, K_1 + K_2, \mathcal{M})$.

Note that by sending the C_i values in aggregated form to the AS, we ensure that the AS does not have the ability to decrypt individual data observations. In addition, the cover nodes are unable to decrypt the encrypted values C_i.

Potential Drawbacks - Applying secret perturbation to PSNs enhances data secrecy by concealing measurement values from cover nodes. However, it suffers from the following issues:

Privacy - Note that the AS could be malicious in PSNs. Although the cover nodes receive encrypted values, the scheme is vulnerable to a violation of privacy if there is collusion between an aggregating cover node and the AS.

Efficiency - *BSP* requires each source node to inform the AS about their participation in order to calculate the decryption key. This is accomplished by forwarding the source node's ID. However, it imposes extra communication overhead, especially when the number of participants is relatively large.

Integrity - Data may become corrupted either due to the weakness of homomorphic aggregation, where one can add any artificial message, or accidental causes such as a node leaving before conveying received messages to the AS.

In the following, first we seek to address the issues of privacy and efficiency, then we address the problem of integrity.

B. Universal Participation Scheme (UPS)

In order to overcome the issue of privacy in *BSP*, we use the approach of *data splitting* in combination with secret perturbation to prevent any single node having all the data from a source node. Moreover, our *UPS* scheme avoids the need to transmit the IDs of the participating source nodes by having all source nodes contribute a value, which can be zero if the source node has no valid measurement value to contribute.

Inspired by the fact that relying on a single cover node is not a significant challenge for malicious entities to expose measurements, as suggested in [9], we apply a splitting technique to augment users' privacy. Without loss of generality, we express measurements and keys as a function of $D_i = f(d_{i,j})$ and $K_i = f(k_{i,j})$, for $j = 1, \ldots, s$, where $s < N$. f is an additive function under which $X_i \in \mathbb{N}$ is sliced randomly to s number of pieces subject to $X_i = \sum_{j=1}^{s} x_{i,j}$, $x_{i,j} \in \mathbb{N}$.

Let $\mathcal{CN}_i$ be a set of cover nodes selected at random by sn_i. Each source node sn_i takes the following steps:

i. Slice the data and key as $D_i = \sum_{j=1}^{s} d_{i,j}$ and $K_i = \sum_{j=1}^{s} k_{i,j}$, for $j = 1, \ldots, s$, and generate $C_{i,j} = (d_{i,j} + k_{i,j}) \, mod \, \mathcal{M}$
ii. Select s number of cover nodes and send each a $C_{i,j}$

The selected cover nodes, similar to the previous scheme, submit the aggregated ciphertext, e.g., $C_{1,2} + C_{2,2}$ to the AS. With the assumption that all nodes have contributed their data, on receiving partiality aggregated values from the cover nodes, the AS performs the following steps:

i. Calculate the `SUM` of the ciphertexts

$$\sum_{i=1}^{SN} \sum_{j=1}^{s} C_{i,j} = C_{1,1} + \ldots + C_{SN,s}$$

ii. Decrypt the aggregated value

$$\sum_{i=1}^{SN} D_i = (\sum_{i=1}^{SN} \sum_{j=1}^{s} C_{i,j} - \sum_{i=1}^{SN} K_i) \, mod \, \mathcal{M}$$

Due to slicing, the absolute data D_i remains secret unless the AS as well as all of the cover nodes selected by sn_i collude.

Potential Drawbacks - While this scheme is less vulnerable to collusion due to the use of data splitting, it has two potential drawbacks of its own. First, this scheme has more risk of producing an erroneous result, with all the N nodes each submitting s slices, if any of the source nodes fails to contribute a value, or if the value is lost in the transmission. Second, if the number of source nodes with a valid measurement is small compared to the total number of nodes, then there is a high communication overhead. Though having all nodes contribute data is a common solution [11], such an assumption may not be realistic for PSNs with highly dynamic membership.

C. Key Splitting Scheme (KSS)

In order to address the potential drawbacks of the previous two schemes, we propose a variation on data splitting with secret perturbation. This new scheme uses a random key from each source node, rather than the AS shared key.

Nodes with an observation value to contribute generate a random integer $\widetilde{K}$ and use this value to perturb their data. The $\widetilde{K}$ can be generated by using a pseudo-random function, like HMAC, taking the K and an arbitrary string. The advantage of perturbing sensor measurements with $\widetilde{K}$, instead of the shared key K, is that these values can be aggregated and source nodes are not required to inform the AS by sending their IDs.

On receiving a query, sn_i generates $C_i = (D_i + \widetilde{K}_i) \, mod \, \mathcal{M}$ and forwards it directly to the AS, while the implemented $\widetilde{K}$ can be transmitted via a set of s randomly chosen cover nodes as $\widetilde{K}_i = \sum_{i=1}^{s} \tilde{k}_{i,j}$, using the random slicing technique. In this way, the malicious AS cannot disclose the perturbed data C_i unless all the cover nodes CN_i are malicious.

Receiving the ciphertexts and the partially aggregated keys, the AS takes the following steps:

i. Calculate the SUM of the perturbed values and the keys

$$\sum_{i=1}^{SN} C_i = (D_1 + \ldots + D_{SN} + \widetilde{K}_1 + \ldots + \widetilde{K}_{SN}) \, mod \, \mathcal{M}$$

$$\sum_{i=1}^{SN} \sum_{j=1}^{s} \tilde{k}_{i,j} = (\tilde{k}_{1,1} + \ldots + \tilde{k}_{SN,s}) \, mod \, \mathcal{M}$$

ii. Decrypt the aggregated data

$$\sum_{i=1}^{SN} D_i = (\sum_{i=1}^{SN} C_i - (\sum_{i=1}^{SN} \sum_{j=1}^{s} \tilde{k}_{i,j})) \, mod \, \mathcal{M}$$

Potential Drawbacks - While our *KSS* scheme keeps perturbed values out of the reach of malicious neighbours, still they can falsify the key slices $\tilde{k}$. Consequently, we propose the following approach to address this issue.

D. Key Splitting Scheme with Integrity (KSSI)

Having devised an efficient privacy-preserving data aggregation scheme, we now address the last issue, *data integrity*. As discussed in Section IV-A, transmitted data, whether it is a data slice of an observation value or a key, can be tampered with by a malicious node or corrupted unintentionally. For either reason, falsified data degrades system accuracy and reliability. Consequently, we equip the AS with an integrity check to detect distorted results. We propose a secure *homomorphic MAC*, and use *KSS* as our underlying scheme, although our integrity check can be applied to the other approaches as well.

The proposed MAC is based on the discrete logarithm, and its homomorphic property allows the AS to verify the integrity of aggregated keys. Let g be a generator of a multiplicative cyclic group G_q of prime order q and p be a large prime number. The AS circulates g and p to all users joining the network as public values. A source node, sn_i, generates the MAC of its randomly generated key as $MAC(\widetilde{K}_i, g) = g^{\widetilde{K}_i} \, mod \, p$ and sends it to the AS. This MAC has the homomorphic property since $MAC(\widetilde{K}_1, g) \times MAC(\widetilde{K}_2, g) = g^{\widetilde{K}_1 + \widetilde{K}_2} \, mod \, p$.

Following *KSS*, the AS receives partially aggregated key slices and calculates $\sum_{i=1}^{SN} \sum_{j=1}^{s} \tilde{k}_{i,j}$. The integrity of aggregated keys can be checked against the $MAC(\widetilde{K}, g)$ values sent by the source nodes. To do this, the AS takes the following steps:

i. Aggregate the MAC values

$$MAC(\sum_{i=1}^{SN} \widetilde{K}_i, g) = MAC(\widetilde{K}_1, g) \times \ldots \times MAC(\widetilde{K}_{SN}, g)$$

ii. Generate the MAC of the aggregated keys

$$MAC'(\sum_{i=1}^{SN} \sum_{j=1}^{s} \tilde{k}_{i,j}, g) = g^{\tilde{k}_{1,1} + \ldots + \tilde{k}_{SN,s}} \, mod \, p$$

Any inconsistency between MAC and MAC′ implies that the keys have been corrupted.

V. Performance Analysis

In the following, we analyse the performance of the aforementioned schemes using the measures introduced in [2].

- *Hidden Probability* ($\mathcal{H}_{pr}$): the probability that a sensor measurement, D_i, remains hidden from malicious entities.
- *Communication Cost* ($\mathcal{T}$): the total communication overhead associated with aggregated sensor measurements in response to a query.

A. Hidden Probability

Hidden probability measures the likelihood that a data observation cannot be associated with its corresponding source node. Since both data and keys are shared between cover nodes and the AS, this metric can be defined in terms of the number of malicious nodes in the network. In this paper we use $\widehat{\ }$ as a symbol for malicious entities and we assume an average of $\widehat{A}$ out of A ASs and $\widehat{N}$ out of N nodes are malicious.

To overcome the privacy issues in PSNs we have used a splitting technique. Here we study the impact of data splitting (as in *UPS* and *KSS*) on hidden probability compared to the non-splitting approach (*BSP*).

Non-Splitting - Since the AS can also be malicious in PSNs, schemes like *BSP* that forward their measurements via other nodes fall short of providing robust privacy. If the selected neighbour is malicious it can collude with the $\widehat{AS}$ and disclose the data. Recall our assumption that on average, $\widehat{N}$ out of N nodes are malicious, the probability that sn_i selects a

malicious cover node is equal to $\frac{\widehat{N}}{N}$. Moreover, if all the source nodes except for sn_i are malicious, a malicious $\widehat{AS}$ can collud with them, to breach sn_i's privacy by revealing their records. Suppose $\Gamma(x)$ is the event that x is malicious, then the hidden probability $\mathcal{H}_{pr}$ for *BSP* is estimated as[1]:

$$\begin{aligned}\mathcal{H}_{pr} &= 1 - Pr\,(\Gamma(AS) \cap (\Gamma(\mathcal{CN}_i) \cup \Gamma(\mathcal{SN}\backslash\{i\})))\\ &= 1 - (\frac{\widehat{A}}{A}\,Pr\,(\Gamma(cn)) + \frac{\widehat{A}}{A}\prod_{sn\in\mathcal{SN}\backslash\{i\}} Pr\,(\Gamma(sn)))\\ &= 1 - (\frac{\widehat{A}}{A}(\frac{\widehat{N}}{N}) + \frac{\widehat{A}}{A}(\frac{\widehat{N}}{N})^{SN-1})\end{aligned}$$

Splitting - Under *UPS* and *KSS*, source nodes share their perturbed data and key with their neighbours, respectively. Sharing makes data disclosure more difficult for the $\widehat{AS}$, as it requires collaboration of all cover nodes $\mathcal{CN}_i$ or $\mathcal{SN}\backslash\{i\}$ to obtain D_i, otherwise the data remains secret. Therefore, the hidden probability for *UPS* can be estimated as:

$$\begin{aligned}\mathcal{H}_{pr} &= 1 - Pr(\Gamma(AS) \cap\ (\Gamma(\mathcal{CN}_i) \cup \Gamma(\mathcal{SN}\backslash\{i\})))\\ &= 1 - (\frac{\widehat{A}}{A}(\frac{\widehat{N}}{N})^{s} + \frac{\widehat{A}}{A}(\frac{\widehat{N}}{N})^{SN-1})\end{aligned}$$

B. Communication Overhead

For each of the aforementioned schemes, we measure the communication overhead associated with each node's interaction separately. Let $\mathcal{T}_{SN-CN}$ and $\mathcal{T}_{SN-AS}$ denote the communication cost incurred due to sending data from source nodes to their cover nodes and the AS, respectively, and $\mathcal{T}_{CN-AS}$ denote the cost incurred due to transmitting data from selected cover nodes to the AS. Finally, the total communication cost in each scheme is obtained as $\mathcal{T} = \mathcal{T}_{SN-CN} + \mathcal{T}_{SN-AS} + \mathcal{T}_{CN-AS}$.

BSP - Usually in PSNs not all users respond to each query. Using BSP, the AS needs to be informed about the involved source nodes to derive the decryption key, and it is accomplished by forwarding source nodes identities.

We first estimate $\mathcal{T}_{SN-CN}$, corresponding to the cost incurred by route discovery to a cover node, in addition to the cost of unicasting messages. We assume the average cost associated with route discovery and the route reply is φ bits. Let $\mathcal{M}_\ell$ be the average length of a message in bits, and let SN denote number of source nodes, then $\mathcal{T}_{SN-CN} = SN \times (\mathcal{M}_\ell + \varphi)$.

Assuming that devices are capable of direct communication, and I_ℓ is the length of an ID in bits, then $\mathcal{T}_{CN-AS}$ is obtained as $\mathcal{T}_{SN-AS} = SN \times I_\ell$. However, our scheme is flexible enough to be extended to hierarchical architectures as well.

Next we model the overhead $\mathcal{T}_{CN-AS}$ that results from a set of cover nodes transmitting the aggregation of data and IDs. Let N_C be the expected number of nodes selected as cover nodes. A node $n_l \in \mathcal{N}$ is selected as a cover node by a source node with a probability of $1/(N-1)$. Therefore, the probability of the node n_l being chosen by at least one source node is $1 - (\frac{N-2}{N-1})^{SN}$, and N_C is calculated as $N_C = N \times (1 - (\frac{N-2}{N-1})^{SN})$.

[1]For ease of estimating $\mathcal{H}_{pr}$, we ignore the impact of those source nodes selected by sn_i as a cover node, which also overlap with $\widehat{CN_i}$.

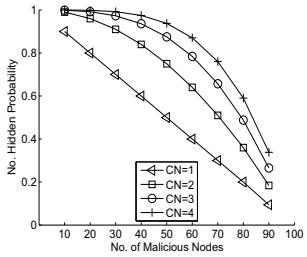


Fig. 2 Hidden probability given number of cover nodes.

Hence, the cost incurred from cover nodes communicating to the AS can be calculated as $\mathcal{T}_{CN-AS} = N_C \times \mathcal{M}_\ell$.

Using the *BSP* scheme, the risk of privacy violation potentially increases, as well as the overhead incurred due to the transmission of all user IDs in large communities.

UPS - In order to avoid the communication overhead of source nodes sending their IDs in *BSP*, the *UPS* scheme requires every node n_l, whether or not $n_l \in \mathcal{SN}$, to reply to the disseminated queries. Accordingly, N_C can be estimated as $N_C = N \times (1 - (\frac{N-s-1}{N-1})^N)$.

Since nodes are not required to pass their IDs to the AS the total communication cost includes $\mathcal{T}_{SN-CN} = N \times s \times (\mathcal{M}_\ell + \varphi)$ and $\mathcal{T}_{CN-AS} = N_C \times \mathcal{M}_\ell$. Although all the nodes need to send some form of message, this scheme is beneficial when the majority of nodes have data to submit.

KSS - To estimate the communication cost of the *KSS* scheme where source nodes directly send their perturbed values C to the AS, the incurred $\mathcal{T}_{SN-AS} = SN \times \mathcal{M}_\ell$.

Let $\mathcal{K}$ be the size of generated key K, which should be at least the same size as D to ensure enough secrecy [14]. So the $\mathcal{T}_{SN-CN}$ is calculated as $\mathcal{T}_{SN-CN} = SN \times s \times (\mathcal{K} + \varphi)$.

The sliced keys are integer values and since data overflow may occur due to data aggregation, we include a carry-bit header and estimate its length as $H_{cb} = log\, N_S$. Let N_S be the expected number of source nodes choosing a node as a cover node, then $N_S = SN(\frac{s}{N-1})$ and $N_C = N \times (1 - (\frac{N-s-1}{N-1})^{SN})$. Therefore, $\mathcal{T}_{CN-AS}$ is obtained as $\mathcal{T}_{CN-AS} = N_C \times (\mathcal{K} + H_{cb})$.

KSSI - The additional option of an integrity check comes with a substantial communication overhead, which can be considered as the trade-off for improved accuracy. Let $\mathcal{K}_M$ be the size of $\widetilde{K}$ chosen for the *KSSI* scheme and M_ℓ be the size of the MAC, then $\mathcal{T}_{SN-AS} = SN \times (M_\ell + \mathcal{M}_\ell)$. While $\mathcal{T}_{SN-CN}$ and $\mathcal{T}_{CN-AS}$ are essentially the same as in *KSS*, except with a key size of $\mathcal{K}_M$ bits (n.b., at least $|\mathcal{K}_M| \geq 180$ bits).

VI. Evaluation Results

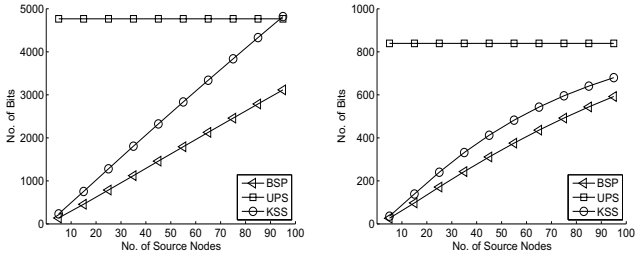
In this section, we evaluate the performance of the proposed schemes in terms of hidden probability and communication overhead based on our analytical model and empirical results. For this purpose, we analysed the number of malicious nodes $\widehat{N}$, source nodes SN and data slices s on the performance of a network with 100 participants and an aggregation server. Table I shows the other default parameter values.

A. Hidden Probability

Fig. 2 illustrates the effect of varying of the number of cover nodes s and malicious nodes $\widehat{N}$ on hidden probability,

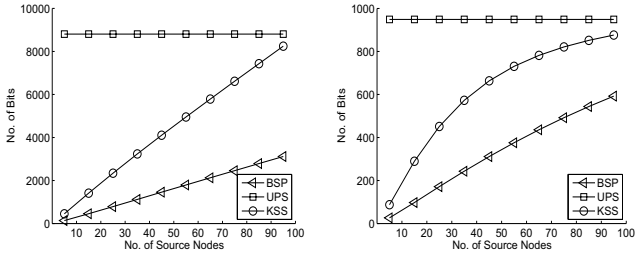
TABLE I DEFAULT EVALUATION PARAMETERS

Par.	SN	$\widehat{N}$	s	$\mathcal{M}_\ell$	M_ℓ	$\mathcal{K}$	$\mathcal{K}_M$	I_ℓ	φ
Val.	50	50	2	12	180	7	180	7	10



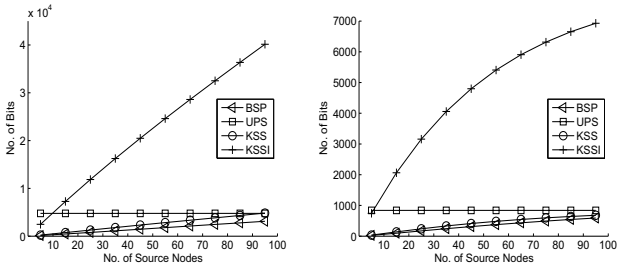
(a) Total communication cost (b) Cover nodes communication cost

Fig. 3 Communication cost under the default settings.



(a) Total communication cost (b) Cover nodes communication cost

Fig. 4 Communication cost with $CN = 4$.



(a) Total communication cost (b) Cover nodes communication cost

Fig. 5 Comparison of communication cost for *KSSI*.

when the AS is malicious. It confirms that selecting a greater number of cover nodes and sharing information enhances the hidden probability, regardless of the number of malicious nodes. As can be seen in Fig. 2, when $CN = 1$, i.e., non-splitting, the hidden probability is much lower, and thus is more vulnerable. As the number of malicious nodes $\widehat{N}$ increases, so does the probability of data exposure. This figure shows how sensitive user privacy is to the number of malicious participants. Although the increase in $\widehat{N}$ gradually impacts the resistance of splitting techniques, it is more significant in the case of non-splitting. For example, when about half the nodes are malicious, the risk of violating the privacy of a source node exceeds 50%. However, with two or more cover nodes each node can have a confidence of privacy of at least 75%.

B. Communication Overhead

We use the analysis of Section V-B to calculate the total communication overhead for each scheme as well as the cost imposed on cover nodes as a result of forwarding their neighbours' data. In terms of communication overhead, the following three factors play the main role in defining the network communication cost: N, SN and s. Accordingly, we examine our scheme efficiency in terms of these factors. Fig. 3 shows the communication overhead for the *BSP*, *UPS* and *KSS* schemes under the default settings. As expected, the cost of *UPS* is independent of the number of source nodes, while the other two schemes experience a linear increase with the growth of SN. The higher overhead of the *UPS* scheme implies that it is not an efficient solution for those networks in which only a small proportion of nodes are source nodes.

Fig. 4 illustrates the impact of the number of cover nodes on communication overhead. Doubling the number of cover nodes (from $CN = 2$ in Fig. 3 to $CN = 4$ in Fig. 4), nearly doubles the transmission cost for *UPS* and *KSS*, which can be considered as the trade-off for privacy. Fig. 5 shows the overhead incurred by the integrity check. Though including the option of integrity imposes a greater burden on the system (especially on the cover nodes), this is the cost of accuracy.

VII. CONCLUSION

We presented four novel schemes to address the problems of data privacy and integrity in participatory sensing. Our simulation results and analytical models show that our approaches can ensure user privacy with high probability, while defecting a loss of integrity. As future work, we will investigate other classes of aggregation queries.

ACKNOWLEDGMENT

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

REFERENCES

[1] A. Kapadia et al., "Opportunistic sensing: Security challenges for the new paradigm," in *PCOMSNETS 2009*, pp. 1–10.

[2] J. Shi et al., "PriSense: Privacy-Preserving Data Aggregation in People-Centric Urban Sensing Systems," in *INFOCOM 2010*, pp. 1–9.

[3] C. Castelluccia et al., "Efficient aggregation of encrypted data in wireless sensor networks," in *MobiQuitous 2005*, pp. 109–117.

[4] J. Burke et al., "Participatory sensing," in *SenSys 2006*.

[5] A. T. Campbell et al., "People-centric urban sensing," in *WISe 2006*.

[6] D. Christin et al., "A survey on privacy in mobile participatory sensing applications," *JSS 2011*, vol. 84, pp. 1928–1946.

[7] M. Srivastava et al., "Human-centric sensing," *Phil Trans Math Phys Eng Sci 2012*, vol. 370, pp. 176–197.

[8] B. Przydatek et al., "SIA: secure information aggregation in sensor networks," in *SenSys 2003*, pp. 255–265.

[9] W. He et al., "PDA: Privacy-preserving data aggregation in wireless sensor networks," in *INFOCOM 2007*, pp. 2045–2053.

[10] S. Papadopoulos et al., "Exact in-network aggregation with integrity and confidentiality," *TKDE 2012*, vol. 24, pp. 1760–1773.

[11] Q. Li and G. Cao, "Efficient and privacy-preserving data aggregation in mobile sensing," in *ICNP 2012*.

[12] Y. Zhang et al., "ARSA: An attack-resilient security architecture for multihop wireless mesh networks," *JSAC 2006*, vol. 24, pp. 1916–1928.

[13] C. Perkins et al., "Ad hoc on-demand distance vector (AODV) routing," *RFC 3561*, 2003.

[14] A. Menezes et al., *Handbook of applied cryptography*, 1996.