PriSense: Privacy-Preserving Data Aggregation in People-Centric Urban Sensing Systems

Jing Shi, Rui Zhang, Yunzhong Liu, and Yanchao Zhang
Department of Electrical and Computer Engineering
New Jersey Institute of Technology
Email: {js39,rz23,yl92,yczhang}@njit.edu

***Abstract*—People-centric urban sensing is a new paradigm gaining popularity. A main obstacle to its widespread deployment and adoption are the privacy concerns of participating individuals. To tackle this open challenge, this paper presents the design and evaluation of PriSense, a novel solution to privacy-preserving data aggregation in people-centric urban sensing systems. PriSense is based on the concept of data slicing and mixing and can support a wide range of statistical additive and non-additive aggregation functions such as Sum, Average, Variance, Count, Max/Min, Median, Histogram, and Percentile with accurate aggregation results. PriSense can support strong user privacy against a tunable threshold number of colluding users and aggregation servers. The efficacy and efficiency of PriSense are confirmed by thorough analytical and simulation results.**

I. INTRODUCTION

People-centric urban sensing is gaining momentum. It refers to using human-carried mobile devices such as mobile phones and PDAs with ever-growing capabilities in sensing, computation, storage, and communications for urban-scale distributed data collection, analysis, and sharing. With many systems and applications being proposed [1]–[12], people-centric urban sensing is expected to open a new era of exciting scientific, social, and commercial applications.

A people-centric urban sensing system differs significantly from a traditional wireless sensor network. First, system devices are no longer owned and managed by a single authority but belong to individuals with diverse interests. Second, system devices have much more powerful resources than sensor nodes and can be charged regularly. Third, the system features dynamic node mobility. Fourth, sensing data are more related to interactions between people and between people and their surroundings instead of some physical phenomena of interest. Fifth, but not the last, people are no longer just passive data users but also active data contributors.

The widespread deployment and adoption of people-centric urban sensing face many obstacles, among which user privacy is one of the most challenging [3], [9], [10]. For example, in a study of the relationship between air quality and public health, researchers desire some aggregate statistics of people's health data such as heart rates, blood pressure levels, and weights at different sections of an urban area. Individuals are likely unwilling to disclose their personal data if there were no guarantee that their data would not be used to invade their privacy. Such reluctance to participation will obviously jeopardize the future of people-centric urban sensing. This motivates our design of *PriSense*, a solution to enabling accurate statistical aggregates of sensing data while providing strong user privacy in urban sensing systems.

A. Related Work

Due to space limitations, we only review some work closely related to this paper. While people-centric urban sensing, also known as participatory or opportunistic sensing, has been gaining popularity (e.g., [1]–[12]), there is relatively little work focusing on its security and privacy aspects. Kapadia *et al.* [10] survey the security and privacy challenges in opportunistic sensing systems. Cornelius *et al.* [3] present the AnonySense architecture for anonymous tasking and reporting in people-centric sensing systems. AnonySense relies on a Mix network like Minimaster [13] to ensure user privacy, which we will not assume in our scheme. Ganti *et al.* [9] propose PoolView for computing community statistics of time-series data in a privacy-preserving manner, which is best suited for a closed community with a known empirical data distribution.

The closest work to ours is on privacy-preserving aggregation in sensor networks [14]–[19]. The work [14]–[17] can only support additive aggregation functions such as Sum and Average. GP^{2}S [18] can support both additive aggregation functions and non-additive ones such as Max/Min, Median, and Histogram at the sacrifice in data accuracy. The work [19] applies a particular class of encryption transformations to compute two aggregation functions, Average and "movement detection" specific to sensor networks. Such work [14]–[19] is developed for sensor networks with a single authority and static topology thus cannot be directly applied to people-centric urban sensing systems with dynamic node membership and mobility.

There is also a big chunk of work on secure aggregation in sensor networks, see [20]–[22]. Such work ensures that aggregation results are not so different from the true values despite malicious intermediate aggregator nodes and does not address individual nodes' data privacy.

B. Our Contribution

The contribution of this paper is the design and evaluation of PriSense, a novel solution to privacy-preserving data aggregation in people-centric urban sensing systems. PriSense consists of the following two components.

The first component aims at additive aggregation functions. Its basic idea is for each node to slice its data into a certain number (say, $n+1$) of slices before answering the query from an aggregation server. Then it randomly chooses n other nodes, called its *cover nodes*, to which a unique data slice is sent. Finally, each node sends to the aggregation server the sum of its own slice left and the slices received from others along with little side information, based on which the aggregation server can compute an accurate additive aggregation result. In this way, a user's data will be disclosed only when the aggregation server and all his cover nodes collude. This technique shares the similar idea as PDA [16] but differs significantly in the application scenario and thus how the cover nodes are selected. In particular, we present three cover-selection schemes to cope with the dynamic nature of urban sensing systems, including a random scheme, a one-hop scheme, and an adaptive h-hop scheme using expanding-ring search.

The second component is a non-trivial combination of the slicing and mixing technique and binary search to enable privacy-preserving Count queries and to further support a wide range of non-additive aggregation functions like Max/Min, Median, Histogram, and Percentile, all with accurate results.

PriSense is the first work of its kind as far as we know. The tradeoffs between the privacy-preserving performance of the proposed techniques and the related overhead are thoroughly analyzed and evaluated with detailed simulations.

II. System and Adversary Models

In this section, we present the system and adversary models.

A. System Model

There is no universally accepted model for a people-centric urban sensing system. We assume an urban-sensing service provider which deploys a large-scale system similar to a metro-scale wireless mesh network (WMN) [23], as shown in Fig. 1. The extension of our work to other system models is left as future work. The system features a high-speed wireless backbone consisting of M powerful aggregation servers (ASs for short) which also provide network access services for system nodes. Each AS is in charge of a certain region referred to as a *cell* and interacts with nodes there. Here we use the term "node" to indicate a human being who carries a portable device like mobile phones, PDAs, laptops, and MP3 players. The devices have different communication and computation capabilities as well as various embedded sensors such as digital compass, proximity sensors, and health sensors [11].

A node may join the system at will to participate in data sensing and sharing and also enjoy network access. To prevent fraudulent use of system resources and also provide basic privacy assurance to nodes, the system and nodes need mutually authenticate each other. We assume a similar mutual authentication protocol as in [23]. Assume that an AS, denoted by $\mathcal{A}$, can simultaneously accommodate up to 2^λ users. After achieving mutual authentication with a node, say i, $\mathcal{A}$ assigns it a temporal integer-valued ID ID_i, which is an unused one

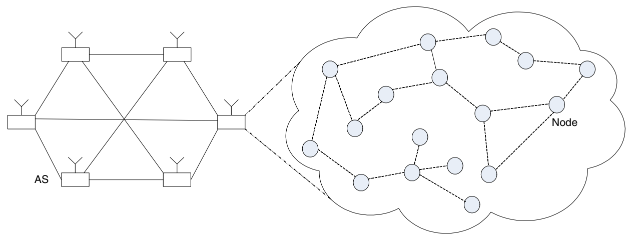


Fig. 1. An urban-sensing system architecture.

between $[0, 2^\lambda - 1]$, and also an ID-based private key K_i. The pair ID_i/K_i will serve as the temporal public/private keys of node i which are valid only in $\mathcal{A}$'s cell. We assume an efficient method for $\mathcal{A}$ to keep track of node mobility in its cell. For example, node i need periodically notify $\mathcal{A}$ about its existence; otherwise, $\mathcal{A}$ would assume that i has left its cell and then reclaim ID_i to be allocated to new nodes. In the latter case, $\mathcal{A}$ updates all the private keys of the remaining nodes in its cell using a single broadcast message with the approach in [24]. The use of such ID-based public/private keys will be illustrated soon. Note that the mutual authentication process is performed each time a node moves into a new cell.

As in [23], we assume an asymmetric communication paradigm in each cell. In particular, each AS is much more capable than regular nodes in communication and computation capacities and other resources. It is, therefore, reasonable to assume that an AS sends packets in one hop to all nodes in its coverage. In contrast, a node may transmit packets in one hop or multiple hops to an AS within or beyond its transmission range. A single-hop downlink can be highly beneficial as shown in [23]. Note that, however, PriSense can be easily extended for use with both multi-hop uplinks and downlinks.

Without loss of generality, we assume the following general scenario throughout the paper. We assume that the service provider wants to get statistical aggregates of some kind of data such as heart rates, weights, and blood pressure levels. A query will be sent to selected ASs which in turn broadcast the query to the nodes inside their cells via the one-hop downlink. If some nodes have data satisfying the query, they will submit their data to its AS if provided with privacy guarantees. The ASs can then aggregate the returned data and forward the aggregation result to the service provider.

B. Adversary Model

This paper focuses on thwarting attacks on breaching nodes' data privacy. Other important issues such as DoS defenses [10] are outside the scope of this paper. We assume that $\bar{M}$ out of M ASs are malicious and that there are on average $\bar{N} \in [0, 2^\lambda - 1]$ malicious nodes in each cell. Malicious ASs and nodes are internal attackers and may collude to attempt deducing personal sensing data of target nodes. There might also be external eavesdroppers outside the system. Since external eavesdroppers are fairly easy to defeat using end-to-end encryption (which we will use), we will focus on defending against internal attackers hereafter.

III. Privacy-Preserving Additive Aggregation

In this section, we present a slicing technique to ensure privacy-preserving additive aggregation. Without loss of generality, our discussion focuses on a cell with AS $\mathcal{A}$ and N nodes, where N is a random number between $(1, 2^{\lambda}]$. We will also use Sum as an example, based on which other additive aggregation functions such as Average and Variance [15] can be easily realized.

The key idea of data slicing is as follows. Let Ψ denote the set of users with data satisfying the sum query, where $1 \le |\Psi| \le N$. Before answering the query, each node $i \in \Psi$ slices its data D_i[1] into $n+1$ random slices with $n \le N-1$, denoted by $\{d_{i,j}\}_{j=1}^{n+1}$, such that $D_i = \sum_{j=1}^{n+1} d_{i,j}$. Then node i keeps $d_{i,n+1}$ to itself while sending each other slice to a unique node called a cover node. Next, the nodes in Ψ and the chosen cover nodes not in Ψ each send to $\mathcal{A}$ the sum of the received slices and its remaining slice (if any). Finally, $\mathcal{A}$ adds up all the received values. It is easy to see that the result is exactly the Sum aggregate of interest. To withstand external attackers, the data sent between nodes and their respective cover nodes and between nodes and $\mathcal{A}$ need be encrypted in an end-to-end fashion.

This slicing technique shares the similar idea as PDA [16]. Our application scenario is, however, totally different. In particular, PDA works in a sensor network which is divided into clusters with relatively static network topology. All the nodes in a cluster have data to report and can be each other's cover node; they also know each other and have pairwise shared keys whereby to encrypt/decrypt data slices transmitted from any node to its chosen cover nodes. Such assumptions no longer hold in PriSense, where nodes in a cell are dynamically changing. Since the nodes do not know each other beforehand, they have no pre-shared keys for end-to-end encryption. In addition, not all the nodes have data satisfying the aggregation query. These significant differences necessitate a new set of cover-selection strategies. In the following, we present three novel cover-selection schemes to cope with the dynamic network topology of urban sensing systems and analyze their performance using the following metrics.

- P (**hidden probability**): the probability that a node's data is hidden from the adversary.
- T (**communication cost**): the total transmission energy consumption in bits per query incurred by PriSense in a cell.

A. Random Cover Selection

In this method, each node $i \in \Psi$ randomly chooses n nodes in a cell and sends a data slice to each of them. The challenge is that i does not know which nodes are in the cell, not to mention having a shared key with them. PriSense uses the following method.

Recall our system model in which the AS $\mathcal{A}$ maintains a list of integer-valued IDs currently in use. As a mesh router [23] or an 802.11 base station, $\mathcal{A}$ need periodically broadcast beacons to announce its existence and other service information. Since each ID is of λ bits, $\mathcal{A}$ can include in each beacon a 2^{λ}-bit vector with corresponding positions set for active IDs currently in use. Various compression algorithms may be used to reduce the size of this vector, which we will not discuss for simplicity. The beacons can also be authenticated using the methods in [23]. When hearing the beacon, all the nodes immediately know the set of active IDs, denoted by $\Delta \subseteq \{0, \ldots, 2^{\lambda} - 1\}$.

Consider node $i \in \Psi$ as an example with data D_i to share. It first slices D_i into $\{d_{i,j}\}_{j=1}^{n+1}$ and then randomly chooses n nodes from Δ as its cover nodes. For any cover node, say $l \in \Delta$, node i computes a shared key $k_{i,l}$ based on its temporal public/private keys ID_i/K_i and ID_l according to [23] and then sends an encrypted unique slice $\langle d_{i,I_{i,l}} \rangle_{k_{i,l}}$ to l, where $I_{i,l} \in [1, n]$ and $\langle \cdot \rangle_*$ denotes an OCB-like length-preserving authenticated encryption primitive [25] using the key on the subscript. Since the route to l might not be known, the packet transmission is normally preceded by an on-demand route discovery process using protocols like AODV [26]. On receiving $\langle d_{i,I_{i,l}} \rangle_{k_{i,l}}$, node l can derive the same key $k_{i,l}$ using its temporal public/private keys ID_l/K_l and ID_i according to [23] and then decrypts the packet to get $d_{i,I_{i,l}}$. Node i repeats this process for all its cover nodes, and so does every other node in Ψ.

Each node waits for sufficient time before finally submitting data to $\mathcal{A}$. Let $\Psi_l \subset \Psi$ denote the set of nodes choosing l as a cover node. Then node l computes the following based on whether it has its own data to report.

$$\begin{cases} D'_l = d_{l,n+1} + \sum\limits_{i \in \Psi_l} d_{i,I_{i,l}} & l \in \Psi \ , \\ D'_l = \sum\limits_{i \in \Psi_l} d_{i,I_{i,l}} & l \notin \Psi \ . \end{cases} \tag{1}$$

There may be data overflows introduced by data slicing and aggregation. So we introduce a carry-bit header to record all the carry bits. The length of the carry-bit header is node-specific and related to the number of times a node was chosen as a cover node. Since x times of aggregation will introduce at most $\log_2 x$ carry bits, node l should use a carry-bit header of at most $\log_2 |\Psi_l|$ bits.

Finally, node l submits $\langle F_l, D'_l, \text{carry-bits} \rangle_{k_{l,\mathcal{A}}}$ via a possibly multi-hop path, where $F_l = 1$ for $l \in \Psi$ and $F_l = 0$ for $l \notin \Psi$, and $k_{l,\mathcal{A}}$ denotes a shared key with $\mathcal{A}$ which can be computed similar to $k_{i,l}$ [23]. If a node neither has its own data nor receives slices from others, it sends nothing to $\mathcal{A}$.

Let Θ denote the set of nodes reporting to $\mathcal{A}$. After receiving all their data and decrypting them, $\mathcal{A}$ simply computes $\sum_{l \in \Theta} D'_l = \sum_{i \in \Psi} D_i$, which is exactly the desired sum. $\mathcal{A}$ can also compute $|\Psi|$ as $\sum_{l \in \Theta} F_l$ whereby to compute the average data value.

Hidden Probability Analysis

Now we analyze the random cover-selection scheme with regard to the hidden probability P. Consider as an example node $i \in \Psi$ whose data D_i is sliced into $\{d_{i,j}\}_{j=1}^{n+1}$. Recall our assumption that there are $\bar{M}$ out of M ASs are malicious

[1]As in [16]–[18] we assume that each data item is an integer for simplicity. Note that some non-integer data can be transformed to integers.

and that there are on average $\bar{N}$ malicious nodes in each cell. Assuming that $\mathcal{A}$ is malicious, which happens with probability $\bar{M}/M$, $\mathcal{A}$ can know D_i if all the $n+1$ slices can be recovered or all the other nodes in Ψ are malicious and colludes with $\mathcal{A}$.

Denote by Υ_i the set of cover nodes chosen by node i which sends a unique slice $d_{i,I_{i,l}} \in \{d_{i,j}\}_{j=1}^{n}$ to node $l \in \Upsilon_i$. The slice $d_{i,I_{i,l}}$ can be recovered by $\mathcal{A}$ if node l is malicious and collude with $\mathcal{A}$, which happens with probability $\bar{N}/N$. In contrast, $d_{i,n+1}$ is kept at node i itself and covered by all received slices. $\mathcal{A}$ can recover $d_{i,n+1}$ if all nodes in Ψ_i are malicious and collude with $\mathcal{A}$, where Ψ_i is the set of nodes choosing node i as a cover node.

Denote by $\mathcal{C}(\star)$ the event that the entity $\star$ is malicious. Combining the above two cases, we have

$$\begin{aligned}\mathsf{P} &= 1 - \Pr(\mathcal{C}(\mathcal{A}) \cap ((\mathcal{C}(\Upsilon_i) \cap \mathcal{C}(\Psi_i)) \cup \mathcal{C}(\Psi \setminus \{i\}))\\ &\geq 1 - \Pr(\mathcal{C}(\mathcal{A}) \cap \mathcal{C}(\Upsilon_i)) - \Pr(\mathcal{C}(\mathcal{A}) \cap \mathcal{C}(\Psi \setminus \{i\}))\\ &= 1 - \frac{\bar{M}}{M}\prod_{l\in\Upsilon_i}\Pr(\mathcal{C}(l)) - \frac{\bar{M}}{M}\prod_{l\in\Psi\setminus\{i\}}\Pr(\mathcal{C}(l)) \qquad (2)\\ &= 1 - \frac{\bar{M}}{M}(\frac{\bar{N}}{N})^{n} - \frac{\bar{M}}{M}(\frac{\bar{N}}{N})^{|\Psi|-1} .\end{aligned}$$

In the above approximation, we ignore the effect of Ψ_i which may overlap with Υ_i and whose expected cardinality $n|\Psi|/(N-1)$ may be much smaller than $|\Upsilon_i| = n$.

Communication Cost Analysis

Now we evaluate the extra communication cost incurred by random cover selection, which consists of three parts. The first, denoted by T_1, is associated with distributing data slices, the second, denoted by T_2, is incurred by the data submissions from those which are not data sources but are chosen as cover nodes by nodes in Ψ, and the third, denoted by T_3, is incurred by transmitting carrier bits from nodes in Ψ to $\mathcal{A}$. Note that we do not consider the cost of transmitting the data from Ψ to $\mathcal{A}$, which is incurred anyway with or without our scheme.

We first estimate T_1, which can be further divided into the cost incurred by route discoveries and the one incurred by unicasting the slices. For simplicity, we assume that both route discovery requests and replies are of l_r bits and that the length of each data slice is of l_s bits. Each route discovery involves a route request broadcasted in the cell, which incurs Nl_r bits (assuming that each node forwards a route request once), and a route reply unicasted from the cover node to the data source, which incurs $\bar{L}l_r$ bits with $\bar{L}$ being the average number of hops between two nodes. Once a route is discovered, a data slice is sent from the data source to the cover node, which again incurs $\bar{L}l_s$ bits. Since there are $|\Psi|$ data sources with each choosing n cover nodes, we have

$$\mathsf{T}_1 = |\Psi| \cdot n(Nl_r + \bar{L}l_r + \bar{L}l_s) . \qquad (3)$$

Now we estimate T_2. Let N_c be the number of nodes which are not a data source but cover nodes. Any node $j \in \Delta \setminus \Psi$ is chosen as a cover node by a data source with probability $n/(N-1)$, and the probability that j is a cover node of at least one data source is thus $1 - (\frac{N-n-1}{N-1})^{|\Psi|}$. Since there are $N - |\Psi|$ nodes not generating any data, we can then estimate N_c as

$$N_c = (N - |\Psi|) \cdot (1 - (\frac{N-n-1}{N-1})^{|\Psi|}) . \qquad (4)$$

Assuming that each node can reach $\mathcal{A}$ in $\bar{L}_{\mathcal{A}}$ hops on average, T_2 can be computed as $\mathsf{T}_2 = N_c(l_s+1+l_c)\bar{L}_{\mathcal{A}}$, where l_c denotes the average number of carry bits. In addition, $\mathsf{T}_3 = |\Psi|(1 + l_c)\bar{L}_{\mathcal{A}}$. Finally, we can compute $\mathsf{T} = \mathsf{T}_1 + \mathsf{T}_2 + \mathsf{T}_3$.

B. One-Hop Cover Selection

Random cover selection may not be efficient because cover nodes are randomly chosen regardless of their locations. As a result, an on-demand route discovery process is often incurred to find a route to a chosen cover node multi-hop away. This may cause unnecessarily high energy consumption because a route request often involves cell-wide broadcasting. This motivates the design of one-hop cover selection. The basic idea is for each node to distribute multiple data slices by a single broadcast to its neighbors, thus avoiding random selection of cover nodes.

Under this method, data slicing involves a period of length T, which is a system parameter known to all nodes in the cell. Assume that each node knows all its one-hop neighbors and can thus establish a pairwise shared key with each of them as before. Continue with the previous example that nodes in $\Psi \subseteq \{0, \ldots, 2^{\lambda}-1\}$ have data to submit. During time T, each node $i \in \Psi$ is allowed to broadcast α random *seeds*, denoted by $\{r_{i,x}\}_{x=1}^{\alpha}$, to its one-hop neighbors at time instances decided by itself. Due to node mobility, $\{r_{i,x}\}_{x=1}^{\alpha}$ are highly likely to be received by different sets of neighbors. A highly privacy-sensitive node may choose to broadcast at locations with more neighbors, as each neighbor serves as its cover node. The data slice for some neighbor receiving $r_{i,x}$ is the keyed hash of $r_{i,x}$ computed using the shared key with that neighbor. It is possible that one node may receive multiple random seeds from i, in which case only the first seed will be processed by both nodes.

To be more specific, each node l maintains an aggregation counter D_l, which is equal to zero for $l \notin \Psi$. Each time overhearing a random seed, say $r_{i,x}$, from some node, say $i \in \Psi$, node l updates $D_l := D_l + h_{k_{i,l}}(r_{i,x})$, where $h_*(\cdot)$ denotes a good hash function using the key on the subscript. On the other hand, if $l \in \Psi$ and its random seed $r_{l,x}$ is overheard by a set $\mathcal{N}_{l,x}$ of neighbors, l updates $D_l := D_l - \sum_{s\in\mathcal{N}_{l,x}} h_{l,s}(r_{l,x})$.

Let $\Psi_l \subset \Psi$ denote the set of nodes choosing l as a cover node. When the slicing period expires, l's aggregation counter is of the following value based on whether it has its own data to report:

$$\begin{cases} D'_l = D_l - \sum\limits_{x=1}^{\alpha}\sum\limits_{j\in\mathcal{N}_{l,x}} h_{k_{l,j}}(r_{l,x}) + \sum\limits_{i\in\Psi_l} h_{k_{i,l}}(r_{i,f_{i,l}}) & l \in \Psi ,\\ D'_l = \sum\limits_{i\in\Psi_l} h_{k_{i,l}}(r_{i,f_{i,l}}) & l \notin \Psi ,\end{cases} \qquad (5)$$

where $r_{i,f_{i,l}}$ means the first seed l received from node i.

Finally, node l submits the following message $\langle F_l, D'_l, \text{carry-bits}\rangle_{k_{l,\mathcal{A}}}$ in a possibly multi-hop path, where $F_l = 1$ for $l \in \Psi$ and $F_l = 0$ for $l \notin \Psi$, and $k_{l,\mathcal{A}}$ denotes a shared key with $\mathcal{A}$ which can be computed similarly to $k_{i,l}$ [23]. If a node neither has its own data nor receives slices from others, it sends nothing to $\mathcal{A}$. As before, $\mathcal{A}$ can get the desired sum by decrypting the received data and adding them up and also compute the average.

Hidden Probability Analysis

Different from random cover selection, one-hop cover selection results in $\sum_{x=1}^{\alpha} |\mathcal{N}_{i,x}|$ cover nodes for any source $i \in \Psi$. Assume that every node has μ neighbors anytime, anywhere in a cell. The first seed $r_{i,1}$ of source i will be received by $|\mathcal{N}_{i,1}| = \mu$ cover nodes. For simplicity, we assume that the rest $\alpha-1$ random numbers each are received by $\mu' \leq \mu$ new cover nodes on average, where $|\mathcal{N}_{i,x}| = \mu', x \in [2, \alpha]$, depends on node mobility and the interval between broadcasting $r_{i,x-1}$ and $r_{i,x}$. This assumption holds when α is small such that $\alpha\mu \ll N$. It follows that $\sum_{x=1}^{\alpha} |\mathcal{N}_{i,x}| \approx \mu+(\alpha-1)\mu'$. Similar to the analysis about random-cover selection, we then have

$$\mathsf{P} \geq 1 - \frac{\bar{M}}{M} \cdot (\frac{\bar{N}}{N})^{\mu+(\alpha-1)\mu'} - \frac{\bar{M}}{M}(\frac{\bar{N}}{N})^{|\Psi|-1} . \quad (6)$$

Communication Cost Analysis

Similar to random cover selection, one-hop cover selection incurs a communication cost $\mathsf{T} = \mathsf{T}_1+\mathsf{T}_2+\mathsf{T}_3$, where T_1, T_2, and T_3 are defined similar to those in the analysis of random cover selection. Assume that each random seed broadcasted by a data source is of l_v bits. Since each source $i \in \Psi$ broadcasts α random seeds, we have $\mathsf{T}_1 = |\Psi|\alpha l_v$. To estimate T_2, we set $n = \mu + (\alpha - 1)\mu'$ in Eq. (4) to get

$$N_c = (N - |\Psi|) \cdot (1 - (\frac{N - \mu - (\alpha - 1)\mu' - 1}{N - 1})^{|\Psi|}) . \quad (7)$$

It follows that $\mathsf{T}_2 = N_c(l_s + l_c + 1)\bar{L}_{\mathcal{A}}$, where l_c denotes the average number of carry bits. In addition, $\mathsf{T}_3 = |\Psi|(l_c+1)\bar{L}_{\mathcal{A}}$. Finally, we can compute T.

C. *h*-Hop cover selection

The performance of one-hop cover selection highly depends on node mobility which in turn determines the number of distinct cover nodes chosen by each data source. We thus propose an h-hop cover-selection strategy to accommodate low node mobility.

Consider the previous example in which nodes in $\Psi \subseteq \{0, \ldots, 2^{\lambda}-1\}$ have data satisfying the query. As before, each node $i \in \Psi$ need select the cover nodes and send a unique data slice of its data D_i to each of them. In h-hop cover selection, each node with data to submit distributes its data slices within its h-hop neighborhood, where h is a system parameter.

Specifically, node i initiates the slicing process by broadcasting a random number r_i with a TTL value set to h. Upon receiving a request with a TTL larger than one, each node further broadcasts it after decreasing the TTL by one. A node should only process the first copy of the same request which may be heard multiple times. In addition, each node memorizes the parent node from which this request came. In this way, a routing tree is formed and rooted at node i. When a node receives a request with the TTL value equal to one, the node should send a reply to its parent node which in turn forwards the reply via the routing tree back to node i after appending its ID. The remaining operations are the same as in one-hop cover selection and omitted for brevity.

Similar to one-hop cover selection, the number of cover nodes is a random variable which cannot be determined before the process is completed. Intuitively, the larger h, the more cover nodes discovered, the higher privacy and the communication cost, and vice versa. In practice, h can be chosen as follows. Assume that the location distribution of the N nodes follows a 2-dimensional Poisson distribution in the cell of area $\pi\mathcal{R}^2$, where $\mathcal{R}$ denote the AS $\mathcal{A}$'s downlink transmission range. Given the transmission range R of nodes, the probability that there are n nodes within the h-hop range of a data source is approximately

$$\Pr[\chi(h) = n] = \frac{e^{-\gamma}\gamma^h}{h!} ,$$

where $\gamma = \frac{NR^2}{\mathcal{R}^2}$, then h can be set such that $\Pr[\chi(h) \geq n]$ is not smaller than a desired probability, say 0.7. Below we analyze its performance.

Hidden Probability Analysis

Similar to random and one-hop cover selection, the hidden probability P of h-hop cover selection is determined by the number n of cover nodes of each source, where n is a random variable with mean $\bar{n} = Nh^2R^2/\mathcal{R}^2$. Then we have

$$\mathsf{P} \geq 1 - \frac{\bar{M}}{M} \cdot (\frac{\bar{N}}{N})^{\bar{n}} - \frac{\bar{M}}{M}(\frac{\bar{N}}{N})^{|\Psi|-1} . \quad (8)$$

Communication Cost Analysis

The communication cost T consists of four parts: T_1 incurred by the h-hop broadcasting of the random number, T_2 incurred by all the cover nodes except data sources submitting combined data slices and corresponding carrier bits, T_3 incurred by the $|\Psi|$ sources submitting their carrier bits, and T_4 incurred by the cover nodes returning their IDs to the data sources. Since each h-hop broadcasts incurs a total cost of $\bar{n}l_s$, the $|\Psi|$ sources incur a total cost of $\mathsf{T}_1 = |\Psi|\bar{n}l_v$. Assuming that each reply from a cover node is of $h\lambda$ bits, which can accommodate h node IDs, we have $\mathsf{T}_4 = |\Psi|\bar{n}h\lambda$. Same as before, we can compute

$$\mathsf{T}_2 = (N - |\Psi|) \cdot (1 - (\frac{N - \bar{n} - 1}{N - 1})^{|\Psi|})(l_s + l_c + 1)\bar{L}_{\mathcal{A}} , \quad (9)$$

where l_c denotes the average number of carrier bits, and $\mathsf{T}_3 = |\Psi|(1+l_c)\bar{L}_{\mathcal{A}}$. Finally, we can compute $\mathsf{T} = \mathsf{T}_1+\mathsf{T}_2+\mathsf{T}_3+\mathsf{T}_4$.

IV. Privacy-Preserving Non-additive Aggregation

The slicing technique in Section III cannot be directly applied to non-additive aggregation functions such as Max/Min, Median, Percentile, and Histogram which have wide applications in practice. We observe that those functions are related to count queries which ask for the number of nodes whose data values are above, below, or equal to a certain value. In particular, $d_{\max}$ (respectively, $d_{\min}$) is the result of a Max (respectively, Min) query if at least one node has $d_{\max}$ (respectively, $d_{\min}$), but no other nodes have data larger than $d_{\max}$ (respectively, smaller than $d_{\min}$); d_{med} is the result of a Median query if one half of nodes have data smaller than d_{med}, and the other half have data larger than d_{med}. In addition, Percentile and Histogram queries can be realized by a series of count queries counting the numbers of nodes in specified data ranges. This observation motivates us to enable privacy-preserving non-additive aggregation with a combination of the slicing technique, count query, and binary search.

A. Scheme Description

Given a non-additive aggregation request, $\mathcal{A}$ transforms it into a count query about how many nodes possess data above, below, or equal to a threshold called a *count index*. Each node with the desired data type gives a "yes" or "no" answer by a single bit of value one or zero, respectively. The answers should be sent via the previous slicing technique to preserve their privacy. The sum received by $\mathcal{A}$ is exactly the number of nodes satisfying the count query. If the data returned satisfy the request requirement, the query process stops; otherwise, $\mathcal{A}$ can send another count query with a new count index. This process continues until the non-additive aggregation function of interest is fulfilled.

To further illustrate the use of count queries, we follow the common assumption [16]–[18] that each data value is an integer between $[0, 2^{\omega+1} - 1]$. The choices of count indexes have a large impact on the number of count queries or rounds needed to realize a non-additive aggregation function, which in turn determines the query overhead. Our scheme depends on binary search to reduce the potentially adverse impact of random count indexes. Below we brief how to realize privacy-preserving Max/Min, Median, Histogram, and Percentile aggregation queries. It is easy to extend our technique to other non-additive aggregation functions.

Max/Min

Since the Min operation is opposite to the Max operation, we just illustrate the latter for brevity. Given a Max aggregation request, $\mathcal{A}$ issues a count query about how many nodes have data above or equal to $Q_1 = 2^{\omega}$. With the slicing technique, $\mathcal{A}$ aggregates the received data to get the number of "yes" answers, denoted by a_1. If $a_1 \geq 1$, the maximum value should be in $[2^{\omega}, 2^{\omega+1} - 1]$, so $\mathcal{A}$ will send a new count query with $Q_2 = 2^{\omega} + 2^{\omega-1}$; otherwise, the maximum value should be in $[0, 2^{\omega} - 1]$, so $\mathcal{A}$ will send a new count query with $Q_2 = 2^{\omega-1}$. The suspicion range in which the maximum value is located is reduced by half for each additional count query. This process continues until the suspicion range is reduced to one, in which case the last count index is exactly the maximum value and the last query result refers to the number of nodes with the maximum value.

Median

A median is interpreted as the number separating the higher half of a sample, a population, or a probability distribution, from the lower half. Given a Median aggregation request, $\mathcal{A}$ first sends a count query about how many nodes have data smaller than or equal to $Q_1 = 2^{\omega+1} - 1$. All the nodes with the desired data type will reply with a "yes" answer via the slicing technique. $\mathcal{A}$ can then aggregate all the replies to get the query result, which is exactly the number of qualified nodes, denoted by U. We first assume that U is odd. $\mathcal{A}$ sends the second count query to see how many nodes have data smaller than or equal to $Q_2 = 2^{\omega}$. If the query result is larger than or equal to $(U + 1)/2$, then $\mathcal{A}$ sends the next query with $Q_3 = 2^{\omega-1}$; otherwise, $\mathcal{A}$ sends the next query with $2^{\omega}+2^{\omega-1}$. This process continues until the query result is equal to $\lfloor U/2 \rfloor$ and also the suspicion range of d_{med} is one. The corresponding count index will be d_{med}. If U is even, d_{med} should be the mean of the two values in the middle. The above process can be simply modified to find two query results equal to $(U + 2)/2$ and $(U - 2)/2$, respectively, both with the suspicion range reduced to one. d_{med} is the average of the two corresponding count indexes.

Histogram

In statistics, a histogram is a graphical display of tabulated frequencies, shown as bars. It shows what proportion of cases fall into each of several categories. Given a Histogram aggregation request, $\mathcal{A}$ partitions $[0, 2^{\omega+1} - 1]$ into a certain number of consecutive, non-overlapping intervals according to the aggregation request. It then sends a count query for each interval, and the corresponding query result will be the number of nodes with data falling into that interval.

Percentile

A percentile (or centile) is the value of a variable below which a certain percent of observations fall. Given a Percentile aggregation request with a desired percent $0 < \sigma < 100$, $\mathcal{A}$ first sends a count query about how many nodes have data smaller than or equal to $Q_1 = 2^{\omega+1} - 1$, and the query result denoted by U is exactly the number of qualifying nodes with desired observations. $\mathcal{A}$ then sends the second query with $Q_2 = 2^{\omega}$. If the query result U is larger than or equal to $\lfloor \sigma U \rfloor$, $\mathcal{A}$ sends the next query with $Q_3 = 2^{\omega-1}$ and with $Q_3 = 2^{\omega} + 2^{\omega-1}$ otherwise. This binary search continues until the query result is equal to $\lfloor \sigma U \rfloor$ and the suspicion range of the percentile is reduced to one. The final count index is the desired percentile.

B. Performance Analysis

The hidden probability P used to analyze the performance of additive aggregation can no longer precisely measure the privacy provision of non-additive aggregation. For example, even if the answer of node j to a count query is disclosed, the adversary can only narrow down the search of D_j to a certain range instead of precisely determining D_j. Assume that the adversary knows that D_j is in a range of length R after the whole query process. It is clear that $\mathsf{R} \leq 2^{\omega+1}$ and can be used to analyze the privacy performance of the non-additive aggregation process: the larger R, the higher level of privacy provision, and vice versa. In particular, when

$\mathsf{R} = 2^{\omega+1}$, the adversary has no clue about what D_j is; when $\mathsf{R} = 1$, the adversary has precisely located D_j. We call R the *suspicion width* of D_j hereafter. Without loss of generality, we use Max as an example to evaluate the performance of the non-additive aggregation process. The studies about other non-additive aggregation functions can be conducted similarly. Before proceeding, we want to mention that the Max/Min aggregation functions naturally disclose some information: any user's data will be smaller or equal to $d_{\max}$ and larger or equal to $d_{\min}$. No scheme can prevent this kind of privacy breach which is due to the aggregate functions themselves. In the following, we will ignore such natural privacy breach and focus on the loss of privacy occurring in the query process.

Suspicion Width Analysis

For clarity, we consider a special case where the maximum value $d_{\max} = 2^{\omega+1} - 1$. The similar process can be used to analyze the more general case that $d_{\max}$ may be any value in $[0, 2^{\omega+1} - 1]$, which is not reported here due to space limitations. Let $\{a_{j,i}\}_{i=1}^{\omega+1}$ denote the sequences of answers from node j, where $a_{j,i} = 1$ means "yes" and $a_{j,i} = 0$ means "no". We then have the following observation from the binary search process.

Observation 1: For $d_{\max} = 2^{\omega+1} - 1$, if $a_{j,i} = 1$, then $a_{j,k} = 1$, $\forall k < i$; if $a_{j,i} = 0$, then $a_{j,k} = 0$, $\forall k > i$.

The above observation indicates that all the yes (or no) answers if any are consecutive. Denote by $a_{j,y}$ and $a_{j,n}$ the last yes answer and the first no answer in $\{a_{j,i}\}_{i=1}^{\omega+1}$, respectively. By Observation 1, we know that $\{a_{j,i}\}_{i=1}^{y}$ are all answers of yes, that $\{a_{j,i}\}_{i=n}^{\omega+1}$ are all answers of no, and that $n = y + 1$. Let $\mathsf{R}_i, i \in [1, \omega + 1]$, be the suspicious width after the ith count query, where $\mathsf{R}_0 = 2^{\omega+1}$ and $\mathsf{R}_{\omega+1} = \mathsf{R}$. We then have the following observation.

Observation 2: Assume that $a_{j,i}$ is the first answer recovered by $\mathcal{A}$. If the answer is yes, then $\mathsf{R}_i = 2^{\omega-i+1}$; if the answer is no, then $\mathsf{R}_i = \mathsf{R}_0 - 2^{\omega-i+1}$.

The above observation means that the disclosure of a yes answer or a no answer has different impact on the suspicion width. More specifically, the disclosure of a yes answer for the ith query tells $\mathcal{A}$ that $D_j \in [2^{\omega+1} - 2^{\omega-i+1}, 2^{\omega+1} - 1]$, while the disclosure of a no answer for the ith query tells $\mathcal{A}$ that $D_j \notin [2^{\omega+1} - 2^{\omega-i+1}, 2^{\omega+1} - 1]$.

During the whole query process, each $a_{j,i}, 1 \le i \le \omega + 1$, is disclosed to $\mathcal{A}$ with probability $\mathsf{P}_c = 1 - \mathsf{P}$, where P is determined by the chosen cover-selection scheme introduced before. $\mathcal{A}$ may recover multiple answers in $\{a_{j,i}\}_{i=1}^{\omega+1}$, for which we have the following observation.

Observation 3: Let a_{j,c_y} and a_{j,c_n} be the last disclosed yes answer in $\{a_{j,i}\}_{i=1}^{y}$ and the first disclosed no answer in $\{a_{j,i}\}_{i=n}^{\omega+1}$ $\mathcal{A}$, respectively, where $c_y \le y < n \le c_n$. The suspicion width after the whole query process is $\mathsf{R}_{\omega+1} = 2^{\omega-c_y+1} - 2^{\omega-c_n+1}$.

The above observation indicates that what really matters is the last disclosed yes answer and the first disclosed no answer.

Assume that D_j is uniformly distributed in $[0, 2^{\omega+1} - 1]$. Note that D_j will determine the final suspicion width $\mathsf{R}_{\omega+1}$. To derive $\mathsf{R}_{\omega+1}$, we partition $[0, 2^{\omega+1} - 1]$ into $\omega + 2$ *equivalent classes* denoted by $\{\mathsf{C}_i\}_0^{\omega+1}$, where $\mathsf{C}_i = [2^{\omega+1} - 2^{\omega-i+1}, 2^{\omega+1} - 2^{\omega-i} - 1]$ for $i \in [0, \omega]$, and $\mathsf{C}_{\omega+1} = [2^{\omega+1} - 1, 2^{\omega+1} - 1]$. We then have the final observation from the binary search process.

Observation 4: For all D_{j_1}, D_{j_2} in $\mathsf{C}_i, i \in [0, \omega + 1]$, $\{a_{j_1,k} = a_{j_2,k}\}_{k=1}^{\omega+1}$. In particular, the answers $\{a_{j_1,k}\}_{k=1}^{i}$ and $\{a_{j_2,k}\}_{k=1}^{i}$ are all yes, and $\{a_{j_1,k}\}_{k=i+1}^{w+1}$ and $\{a_{j_2,k}\}_{k=i+1}^{w+1}$ are all no.

The above observation shows that if two data items belong to the same equivalent class, they will have the same sequence of yes/no answers for the $\omega+1$ count queries. Now we estimate the final suspicion width $\mathsf{R}_{\omega+1}$ for a random D_j. Assuming that $D_j \in \mathsf{C}_i$, we have $c_y \le i$ and $c_n \ge i+1$. For a given i, the probability density functions of c_y and c_n can be computed as

$$\Pr(c_y = t) = \begin{cases} (1-\mathsf{P})\mathsf{P}^{i-t} & \text{if} \quad 1 \le t \le i \ , \\ \mathsf{P}^i & \text{if} \quad t = 0 \ , \end{cases} \tag{10}$$

and

$$\Pr(c_n = t) = \begin{cases} (1-\mathsf{P})\mathsf{P}^{t-i+1} & \text{if} \quad i+1 \le t \le \omega+1 \ , \\ \mathsf{P}^{\omega-i+1} & \text{if} \quad t = \omega+2 \ , \end{cases} \tag{11}$$

where we define $c_y = 0$ and $c_n = \omega + 2$ as the cases that no answers of yes or no are disclosed to $\mathcal{A}$, respectively. The suspicion width for given c_y and c_n can then be computed as

$$\mathsf{R}(c_y, c_n) = \begin{cases} 2^{\omega-c_y+1} - 2^{\omega-c_n+1}, & \text{if} \quad i \le c_n \le \omega+1 \ , \\ 2^{\omega-c_y+1}, & \text{if} \quad c_n = \omega+2 \ . \end{cases} \tag{12}$$

Then the expected R for class i is the average over all possible c_y and c_n, which can be computed as

$$E[\mathsf{R}(i)] = \sum_{t_1=0}^{i} \sum_{t_2=i+1}^{\omega+2} \mathsf{R}(t_1, t_2) \cdot \Pr(c_y = t_1) \cdot \Pr(c_y = t_2) \ . \tag{13}$$

The overall expected R for a random D_j uniformly distributed in $[0, 2^{\omega+1} - 1]$ is then given by

$$E[\mathsf{R}] = \sum_{j=0}^{\omega+1} \mathsf{R}(i) \Pr(D_j \in \mathsf{C}_i) \ , \tag{14}$$

where $\Pr(D_j \in \mathsf{C}_i)$ is equal to the length of C_i divided by $2^{\omega+1}$.

Communication Cost Analysis

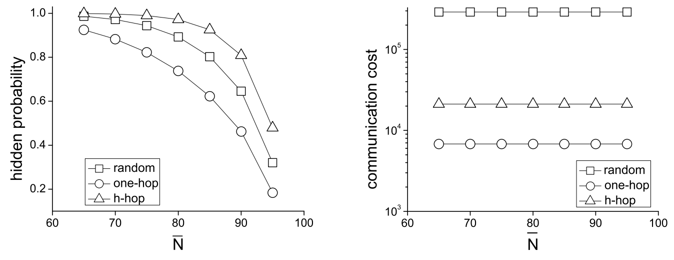
Assume that $\mathcal{A}$ need $\omega + 1$ count queries to get the final results. Since each count query incurs communication cost T, the total communication cost is thus $\mathsf{T}_n = (\omega + 1)\mathsf{T}$.

V. Performance Evaluation

In this section, we thoroughly evaluate the performance of PriSense. We simulate a cell with $N = 100$ nodes, among which 50 are malicious, and a malicious AS at the center of the cell. Table I summarizes the default evaluation parameters unless specified otherwise. Figs. 2 to 5 are for additive aggregation, while Fig. 7 is for non-additive aggregation.

TABLE I
DEFAULT EVALUATION PARAMETERS

Para.	**Val.**	**Para.**	**Val.**	**Para.**	**Val.**	**Para.**	**Val.**
N	100	$\bar{N}$	50	M	10	$\bar{M}$	10
$\|\Psi\|$	50	l_r	5	l_s	10	l_c	8
l_v	5	$\bar{L}$	5	$\bar{L}_{\mathcal{A}}$	5	n	10
μ	4	μ'	2	h	2	R	2
$\mathcal{R}$	10						



(a) hidden probability (b) communication cost

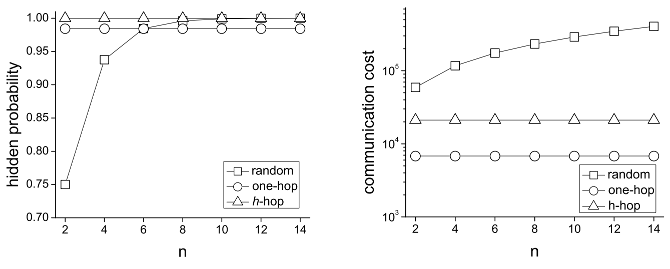
Fig. 2. Impact of $\bar{N}$, the number of malicious nodes.

Fig. 2(a) shows the impact of $\bar{N}$, the number of malicious nodes, on the hidden probabilities of three cover-selection schemes. We can see that the larger $\bar{N}$, the lower the hidden probabilities of all three schemes, and vice versa. This coincides with the intuition: the more malicious nodes in the cell, the higher the probability of the data being disclosed. In addition, under the default configuration, h-hop cover selection has the highest hidden probability, followed by random cover selection and then one-hop cover selection. This is not surprising since h is chosen such that the probability that there are at least n nodes in the h-hop neighborhood is large enough, leading to more than n cover nodes in most cases.

Moreover, we can see from Fig. 2(b) that the communication cost of the three schemes are independent of $\bar{N}$. From the previous analysis, we know that since what really matter are N and $|\Phi|$. Under the default configuration, random cover selection has the highest communication cost while one-hop cover selection has the lowest. The reason is that in one-hop cover selection, nodes only perform local broadcast, while other two schemes involve multi-hop routing.

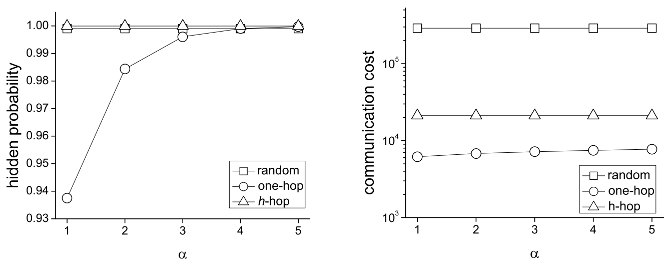
Fig. 3(a) and Fig. 3(b) show the hidden probability and communication cost of random cover selection versus the number n of cover nodes. We can see that the larger n, i.e., the more cover nodes chosen, the higher the hidden probability and also the communication cost. Since other two schemes are not affected by n, they are drawn only for comparison.

Fig. 4 shows the impact of α on one-hop cover selection. We can see from Fig. 4(a) that both the hidden probability and the communication cost of one-hop cover selection increase as α increases. The reason is that the larger α, the more cover nodes will be selected, leading to the increase in the hidden probability and also the communication cost. Similarly, other two schemes are not affected by α and are shown only for illustrative purpose.



(a) hidden probability (b) communication cost

Fig. 3. Impact of n on random cover selection.



(a) hidden probability (b) communication cost

Fig. 4. Impact of α on one-hop cover selection.

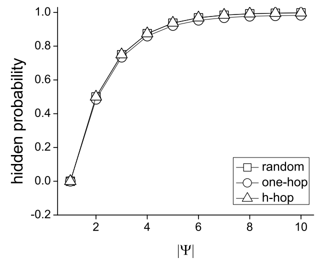
Fig. 6 shows the impact of h on h-hop cover selection, in which we can observe a similar trend as in Fig. 3 and Fig. 4. In general, the larger h, the more cover nodes, the higher hidden probability and communication cost, and vice versa.

Fig. 5 shows the impact of $|\Psi|$ on all three schemes. In general, the larger $|\Psi|$, the less likely that all the other $|\Psi| - 1$ nodes are malicious, and the higher hidden probabilities for all three schemes. It is worth noticing that P increases most drastically when $|\Psi|$ increases from 1 to 4. In the extreme case, when $|\Psi| = 1$, the data will always be recovered by a malicious AP. In addition, a larger $|\Psi|$ leads to higher communication cost, since more nodes have data to submit.

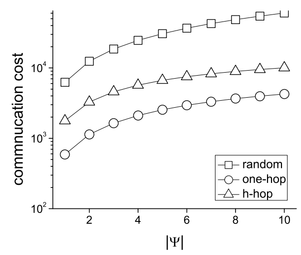
Fig. 7 shows the performance of PriSense for non-additive aggregation with different ω and P. Fig. 7(a) shows the ratio of the suspicion width to the whole data range, i.e, $2^{\omega+1}$. We can see that the larger ω, the higher the ratio, and vice versa, since the number of queries increases with ω, and it becomes more difficult for malicious AP to precisely recover the user data. Furthermore, the higher P, the lower the probability that the answer for one count query can be recovered, and thus the larger suspicion width. Fig. 7(b) shows the communication cost of non-additive query varying with ω with different cover selection schemes. It is of no surprise that there is a linear relationship between the communication cost and ω, since the larger ω, the larger the data length.

We summarize the evaluation results as follows.

- Random cover selection can select a deterministic number of cover nodes and thus can provide privacy guarantee at the cost of higher communication cost.
- One-hop cover selection has the lowest communication

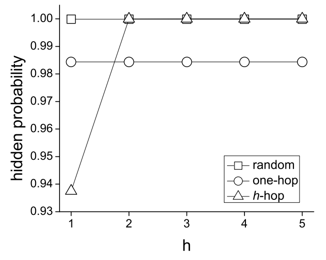


(a) hidden probability

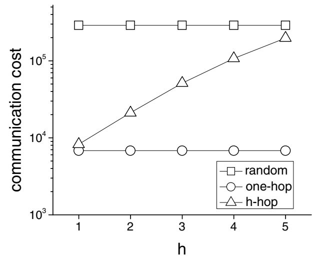


(b) communication cost

Fig. 5. Impact of $|\Psi|$ on cover selection.

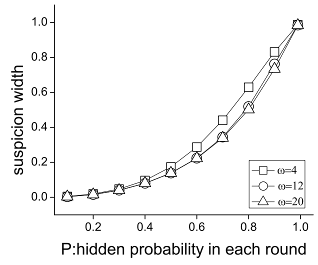


(a) hidden probability

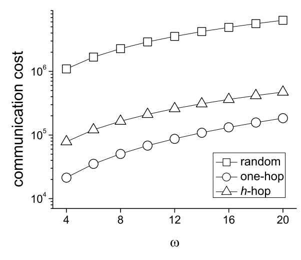


(b) communication cost

Fig. 6. Impact of h on h-hop cover selection.



(a) hidden probability



(b) communication cost

Fig. 7. Impact of ω and P on non-additive aggregation.

cost, but the privacy provision depends on node mobility and network density.

- h-hop cover selection strikes a good balance between privacy and the communication cost.
- Realizing non-additive query with multiple count queries can effectively protect user data privacy with communication cost linear to the length of the user data.

The evaluation results demonstrate that PriSense is very suitable and practical as a solution to privacy-preserving data aggregation in people-centric urban sensing systems. We seek to evaluate PriSense in more realistic and comprehensive settings in our future work.

Acknowledge

This work was supported in part by the US National Science Foundation under grants CNS-0716302 and CNS-0844972 (CAREER). We would also like to thank anonymous reviewers for their constructive comments and helpful advice.

References

[1] E. Paulos and T. Jenkins, "Urban Probes: encountering our emerging urban atmospheres," in *ACM CHI'05*, Portland, OR, April 2005, pp. 341–350.
[2] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, and R. A. Peterson, "People-centric urban sensing," in *WICON'06*, Boston, Massachusetts, Aug. 2006.
[3] C. Cornelius, A. Kapadia, D. Kotz, D. Peebles, M. Shin, and N. Triandopoulos, "Anonysense: privacy-aware people-centric sensing," in *ACM MobiSys'08*, Breckenridge, CO, June 2008, pp. 211–224.
[4] B. Hull, *et al.*, "CarTel: A distributed mobile sensor computing system," in *ACM SenSys'06*, Boulder, CO, Oct. 2006, pp. 125–138.
[5] A. Parker, *et al.*, "Network system challenges in selective sharing and verification for personal, social, and urban-scale sensing applications," in *HotNets-V'06*, Irvine, CA, Nov. 2006, pp. 37–42.
[6] A. Kansal, S. Nath, J. Liu, and F. Zhao, "SenseWeb: An infrastructure for shared sensing," *IEEE MultiMedia*, vol. 14, no. 4, pp. 8–13, 2007.
[7] T. Abdelzaher, *et al.*, "Mobiscopes for human spaces," *IEEE Pervasive Computing*, vol. 6, no. 2, pp. 20–29, 2007.
[8] V. Tuulos, J. Scheible, and H. Nyholm, "Combining web, mobile phones and public displays in large-scale: Manhattan story mashup," in *Pervasive'07*, Toronto, Canada, May 2007, pp. 37–54.
[9] R. K. Ganti, N. Pham, Y.-E. Tsai, and T. F. Abdelzaher, "Poolview: stream privacy for grassroots participatory sensing," in *ACM SenSys'08*, Raleigh, NC, Nov. 2008, pp. 281–294.
[10] A. Kapadia, *et al.*, "Opportunistic sensing: Security challenges for the new paradigm," in *COMSNETS'09*, Bangalore, India, Jan 2009.
[11] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell, "Soundsense: Scalable sound sensing for people-centric applications on mobile phones," in *ACM MobiSys'09*, Wroclaw, Poland, June 2009, pp. 165–178.
[12] R. N. Murty, *et al.*, "CitySense: An urban-scale wireless sensor network and testbed," in *IEEE HST'08*, Waltham, MA, May 2008, pp. 583–588.
[13] U. Moller, L. Cottrell, P. Palfrader, and L. Sassaman, "Mixmaster protocol ł version 2," IETF Internet Draft, July 2003.
[14] J. Girao, *et al.*, "CDA: Concealed data aggregation in wireless sensor networks," in *ACM WiSe'04*, Philadelphia, PA, Oct. 2004.
[15] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks," in *MobiQuitous'05*, San Diego, CA, July 2005, pp. 109–117.
[16] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. F. Abdelzaher, "PDA: Privacy-preserving data aggregation in wireless sensor networks," in *IEEE INFOCOM'07*, Anchorage, Alaska, May 2007, pp. 2045–2053.
[17] T. Feng, C. Wang, W. Zhang, and L. Ruan, "Confidentiality protection for distributed sensor data aggregation," in *IEEE INFOCOM'08*, Phoneix, AZ, Apr. 2008, pp. 475–483.
[18] W. Zhang, C. Wang, and T. Feng, "Gp^{2}s: Generic privacy-preservation solutions for approximate aggregation of sensor data (concise contribution)," in *PerCom'08*, Hong Kong, China, Mar. 2008, pp. 179–184.
[19] D. Westhoff, J. Girao, and M. Acharya, "Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution, and routing adaptation," *IEEE Trans. on Mobile Computing*, vol. 5, no. 10, pp. 1417–1431, Oct. 2006.
[20] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure information aggregation in sensor networks," in *ACM SenSys'03*, Los Angeles, CA, Nov. 2003, pp. 255–265.
[21] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," in *ACM CCS'06*, Alexandria, Virginia, Oct. 2006, pp. 278–287.
[22] S. Roy, S. Setia, and S. Jajodia, "Attack-resilient hierarchical data aggregation in sensor networks," in *SASN'06*, Alexandria, Virginia, USA, Oct. 2006, pp. 71–82.
[23] Y. Zhang and Y. Fang, "ARSA: an attack-resilient security architecture for multi-hop wireless mesh networks," *IEEE J. Select. Areas Commun.*, vol. 24, no. 10, pp. 1916–1928, Oct. 2006.
[24] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Securing mobile ad hoc networks with certificateless public keys," *IEEE Trans. on Dependable and Secure Computing,*, vol. 3, no. 4, pp. 386–399, Oct.-Dec. 2006.
[25] P. Rogaway, M. Bellare, and J. Black, "OCB: A block-cipher mode of operation for efficient authenticated encryption," *ACM Trans. Inf. Syst. Secur.*, vol. 6, no. 3, pp. 365–403, Aug. 2003.
[26] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," RFC 3561, July 2003.