Verifiable Privacy-Preserving Aggregation in People-Centric Urban Sensing Systems

Rui Zhang, *Student Member, IEEE,* Jing Shi, *Member, IEEE,* Yanchao Zhang, *Senior Member, IEEE,* and Chi Zhang, *Member, IEEE*

***Abstract*—People-centric urban sensing systems (PC-USSs) refer to using human-carried mobile devices such as smartphones and tablets for urban-scale distributed data collection, analysis, and sharing to facilitate interaction between humans and their surrounding environments. A main obstacle to the widespread deployment and adoption of PC-USSs are the privacy concerns of participating individuals as well as the concerns about data integrity. To tackle this open challenge, this paper presents the design and evaluation of VPA, a novel peer-to-peer based solution to verifiable privacy-preserving data aggregation in PC-USSs. VPA achieves strong user privacy by letting each user exchange random shares of its datum with other peers, while at the same time ensures data integrity through a combination of Trusted Platform Module and homomorphic message authentication code. VPA 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. The efficacy and efficiency of VPA are confirmed by thorough analytical and simulation results.**

***Keywords*-People-centric urban sensing system (PC-USS), peer-to-peer, aggregation, security, privacy.**

I. Introduction

PEOPLE-centric urban sensing systems (PC-USSs) refer to using human-carried mobile devices such as smartphones and tablets with ever-growing capabilities in sensing, computation, storage, and communications for urban-scale distributed data collection, analysis, and sharing to facilitate the interaction between humans and their surrounding environments. Examples of PC-USS applications include environment monitoring [1], [2], traffic measuring and congestion avoidance [3], [4], healthcare monitoring and delivery [5], and many others [6]–[16]. PC-USSs are expected to open a new era of exciting scientific, social, and commercial applications.

PC-USSs differ significantly from traditional wireless sensor networks that focus on environment sensing and data collection. 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 the interactions among humans and between humans and their surroundings instead of only about some physical phenomena of interest. Fifth, but not the last, humans are no longer just passive data users but also active data contributors.

The widespread deployment and adoption of PC-USSs face many obstacles, of which user privacy and data integrity are among the most critical [7], [13], [14]. For instance, in a study of the relationship between air quality and public health, researchers desire some aggregate statistics of personal health data such as heart rates, blood pressure levels, and weights at different sections of an urban area. Individuals may be unwilling to disclose their personal data if there were no guarantee that their data would not be used to invade their privacy. As an example for data-integrity breach, consider applications like CarTel [8] and VTrack [3] that use traffic statistics such as average speed as an indicator of congestion to help system users do route planning. A selfish and malicious driver may prevent other users from choosing his current road by manipulating the aggregation result, i.e., cheating the server into accepting a lower-than-actual average speed that indicates road congestion. These two examples highlight the necessity for verifiable privacy-preserving data aggregation techniques that can ensure strong user privacy and also aggregation integrity.[1]

Designing a verifiable privacy-preserving aggregation scheme for PC-USSs is particulary challenging. On the one hand, ensuring user privacy means that a user's original data cannot be disclosed to any other party. This requirement makes it hard to detect if a user has faithfully participated in data aggregation. On the other hand, ensuring aggregation integrity requires any misbehavior during data aggregation to be detected with overwhelming probability. This requirement is extremely difficult to satisfy without knowing users' original data.

The contribution of this paper is the design and evaluation of VPA, a novel peer-to-peer based solution to verifiable privacy-preserving data aggregation in PC-USSs. VPA consists of the following two components.

- The first component VPA$^+$ aims at additive aggregation functions such as Sum, Average, and Variance. Its basic idea is to divide the aggregation process into two phases. In the first phase, each node submits a commitment to the aggregation server, which is a homomorphic message authentication code of its original data. The homomorphic property of commitments enables the aggregation server

R. Zhang and Y. Zhang are with the School of Electrical, Computer, and Energy Engineering, Arizona State University, Tempe, AZ 85287 (email: {ruizhang, yczhang}@asu.edu).

J. Shi is with the School of Public Administration, Huazhong University of Science and Technology, China (email: shi.jing@mail.hust.edu.cn).

C. Zhang is with the School of Information Science and Technology, University of Science and Technology, China (email: chizhang@ustc.edu.cn).

[1]We use "user privacy" and "data privacy" as well as "aggregation integrity" and "data integrity" interchangeably in this paper.

to compute the aggregate commitment corresponding to the final aggregate, while it is impossible for the aggregation server to recover any node's original datum. In the second phase, the original datum of each node is aggregated in a privacy-preserving manner, in which users first exchange random shares of their data with selected peers and then submit mixed data to the aggregation server. The aggregation server can then verify the aggregation-result integrity using the aggregate commitment derived in the first phase.

- The second component VPA$^{\oplus}$ is a non-trivial combination of the binary search and verifiable privacy-preserving Count queries and can support a wide range of non-additive aggregation functions like Max/Min, Median, Histogram, and Percentile, with accurate aggregation-results.

VPA is the first work of its kind as far as we know. The performance of VPA is thoroughly analyzed and evaluated with detailed simulations.

The rest of this paper is organized as follows. Section II reviews the related work. Section III gives the system and adversary models and the design objectives. Section IV and Section V present the solutions to additive and non-additive aggregation, respectively. Section VI evaluates the performance of VPA using extensive simulation results. This paper is finally concluded in Section VII.

II. Related Work

Although PC-USSs, also known as participatory or opportunistic sensing systems, have received extensive attention (e.g., [1], [6]–[16]), there is relatively little work focusing on their security and privacy aspects. Kapadia *et al.* [14] surveyed the security and privacy challenges in opportunistic sensing systems. Cornelius *et al.* [7] presented the AnonySense architecture for anonymous tasking and reporting in people-centric sensing systems. AnonySense relies on a Mix network like Minimaster [17] to ensure user privacy, which we will not assume in our scheme. Ganti *et al.* [13] proposed PoolView for computing community statistics of time-series data in a privacy-preserving manner without considering aggregation-result integrity. More recently, Cristofaro and Soriente [18] proposed PEPSI to protect data and query privacy from unauthorized subscribers. None of these schemes could achieve the same objectives as our VPA.

Privacy-preserving aggregation in traditional sensor networks has been extensively studied. The work [19]–[22] can support additive aggregation functions such as Sum and Average. GP^{2}S [23] 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 [24] applies a particular class of encryption transformations to compute two aggregation functions, Average and "movement detection" specific to sensor networks. These schemes [19]–[24] do not address aggregation-result integrity, neither could be be directly applied to PC-USSs due to different application scenarios.

There is also a big chunk of work on secure aggregation in sensor networks, see [25]–[31] for example. 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.

To the best of our knowledge, the schemes in [32]–[34] are the only ones that simultaneously address data confidentiality and aggregation-result integrity in different scenarios. VPA differs significantly from them [32]–[34] in many aspects. For example, VPA can support a large family of additive and non-additive aggregation functions, including Sum, Average, Max/Min, Median, Histogram, and Percentile, while [32]–[34] can only support additive aggregation functions. In addition, the scheme in [32] assumes a trustworthy data publisher, which we assume honest-but-curious aggregation servers. Moreover, the scheme in [33] can only detect false aggregation results with certain probability and protect users' data privacy only against other users. In contrast, VPA can detect any false aggregation result with certainty and ensure users' data privacy against both curious users and aggregation servers. Furthermore, PrivStats [34] does not support in-network aggregation and relies on computationally-expensive zero-knowledge proofs to defend against false-data injection attacks, while VPA supports in-network aggregation and realizes data integrity using TPM.

Finally, location/identity privacy of mobile users is another active topic of research, see [35]–[37] for example. This line of work is orthogonal to our work in this paper and can be integrated with our VPA.

III. Models and Design Goals

In this section, we present the system and adversary models as well as our design goals.

A. System Model

There is no universally accepted model for a PC-USS. For ease of illustration, we assume an urban-sensing service provider which deploys a large-scale system similar to a metro-scale wireless mesh network [36], as shown in Fig. 1. Our solution can be easily extended to work with other system models such as cellular networks. The PC-USS 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 therein. Here we use the term "node" to indicate a human who carries a portable device such as a smartphone and tablet. The devices have different communication and computation capabilities as well as various embedded sensors such as accelerator, digital compass, proximity sensors, and humidity sensors [15].

Whenever a node, say i, enters a new cell with an AS, say $\mathcal{A}$, a similar protocol as in [38] based on Identity-Based Cryptography [39] needs to be performed to enable the mutual authentication between i and $\mathcal{A}$. In particular, each node i has an ID ID_i as its public key and an ID-based private key K_i^{-1} obtained from an offline trusted authority (TA). After

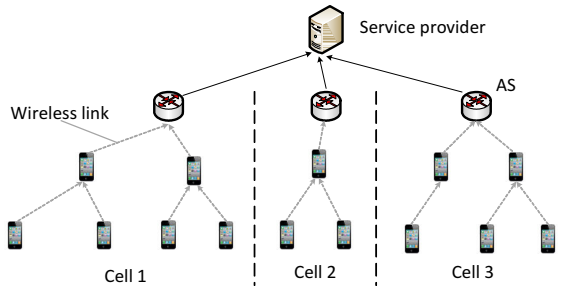


Fig. 1. The abstract architecture of a people-centric urban sensing system (PC-USS).

achieving mutual authentication with a node i, $\mathcal{A}$ also assigns node i a secret key k_i. The use of such ID-based public/private keys and secret key will be illustrated soon.

In addition, we assume an efficient method for $\mathcal{A}$ to keep track of node mobility in its cell. For example, node i needs to periodically notify $\mathcal{A}$ about its existence; otherwise, $\mathcal{A}$ would assume that i has left its cell. $\mathcal{A}$ also includes the number of present nodes via periodically broadcasted beacon messages. We also assume that each node and the AS can directly communicate with each other. In addition, neighboring nodes could communicate with each other through WiFi or Bluetooth interfaces, for which very efficient protocols are available such as in [40].

Without loss of generality, we consider the following scenario throughout. We assume that the service provider, on behalf of a data client, wants to get statistical aggregates of some personal data such as heart rates, blood pressure levels, glucose levels, weights, and moving speeds. A query will be sent to selected ASs which in turn broadcast the query to the nodes inside their respective cells. If some nodes have data satisfying the query, they will participate in aggregation if provided with privacy guarantees. The ASs can then aggregate the returned data and forward the aggregation result to the service provider. The service provider often need provide extra incentives such as credits to motivate participation in data aggregation, but the design of sound incentives is outside the scope of this paper.

B. Adversary Model

This paper focuses on thwarting attacks on breaching nodes' data privacy as well as aggregation-result integrity. Other important issues such as DoS defenses [14] are beyond the scope of this paper.

We assume that ASs are trusted to follow aggregation operations for generating correct aggregation results, but they may be curious about individual user data. A curious AS may collude with other curious nodes to attempt deducing the sensing data of target nodes.

In contrast, a node could be curious, malicious, or both. Like a curious AS, a curious node is interested in discovering other nodes' data but faithfully follow aggregation operations, while a malicious node intends to make the AS derive false aggregation result. More specifically, a malicious node may launch two types of *false-data injection* attacks [28]. First, a malicious node may forge its own datum. Second, a malicious node may forge a false intermediate aggregation result that could significantly affect the final aggregation result. Most recent research [41]–[44] has shown that perhaps the only feasible defense against the former (i.e., ensuring the integrity of sensor readings from human-carried mobile devices) is to use some trusted hardware such as a Trusted Platform Module (TPM). We thus follow this line of research and assume that every participating mobile device has an embedded TPM. To keep the TPM cost as low as possible, we only require the TPM to have a minimal set of functionalities during aggregation, which include collecting sensor readings and generating a message authentication code (MAC). For this purpose, we assume that every TPM has a unique public/private key pair bound to the affiliated mobile device. After achieving mutual authentication with node i, the AS sends another secret key κ_i encrypted with the public key of node i's TPM. The TPM can then decrypt the ciphertext using its private key and store κ_i for later use. Based on the assumption about TPM, we focus on mitigating the forgery of intermediate aggregation results in our solution.

There might also be external eavesdroppers not participating in data aggregation. Since external eavesdroppers are fairly easy to defeat using end-to-end encryption (which we will use), we focus on counteracting internal attackers hereafter, which includes both curious ASs and curious/malicious participating nodes.

C. Design Goals

Given the aforementioned adversary model, VPA is designed with the following objectives.

- *Aggregation accuracy*: VPA should output accurate aggregation results in the absence of malicious attacks.
- *Aggregation/data integrity*: any attempt of injecting false data should be detected with certainty.
- *Data/user privacy*: Each user's datum should be hidden from all the other parties with high probability.
- *Efficiency*: VPA should incur low communication and computation overhead.

IV. VPA$^+$: Verifiable Privacy-Preserving Additive Aggregation

In this section, we present VPA$^+$, a novel scheme to enable verifiable privacy-preserving additive aggregation. Without loss of generality, our discussion focuses on a cell with AS $\mathcal{A}$ and a set of n nodes, denoted by $\mathcal{U}$. We will also use Sum aggregation as an example, based on which other additive aggregation functions such as Average and Variance [20] can be easily realized.

A. Overview and Basic Idea

We observe that either of user privacy and aggregation integrity alone can be easily achieved if we ignore the other. On the one hand, if aggregation integrity is the only concern, a straightforward solution is to let each node submit its datum directly to $\mathcal{A}$ along with a message authentication code

(MAC). The AS can then verify the authenticity of each datum and compute the correct sum. This naive approach, however, offers no data privacy to users. On the other hand, many existing techniques such as [21], [22] can realize privacy-preserving data aggregation, but a malicious node can launch the false-data injection attack without being detected.

Inspired by the above observation, we divide the whole aggregation process into two phases. In the first phase, each node submits to $\mathcal{A}$ a commitment, which is a homomorphic MAC of its datum and has a nice *one-way* property that $\mathcal{A}$ cannot deduce the corresponding datum. The homomorphic property of individual commitments enables $\mathcal{A}$ to compute an aggregate commitment corresponding to the Sum aggregate of all nodes' data. In the second phase, nodes perform privacy-preserving in-network data aggregation for $\mathcal{A}$ to derive the Sum aggregate without disclosing any individual datum with overwhelming probability. Finally, $\mathcal{A}$ can verify the integrity of the Sum aggregate by using the aggregate commitment derived in the first phase. In what follows, we detail the design of VPA$^+$, which includes *aggregation initialization*, *commitment submission*, *privacy-preserving in-network aggregation*, and *aggregation verification*.

B. Aggregation Initialization

The AS $\mathcal{A}$ initializes the aggregation process by selecting a large prime p and a generator g of the group $\mathbb{Z}_p^* = \{1, \ldots, p-1\}$. The parameters p and g should ensure the computational hardness of the *discrete logarithm problem*, that is, given a random $y \in \mathbb{Z}_p^*$, it is computationally infeasible to find the unique integer $x \in [0, p-2]$ such that $g^x = y \mod p$. Assume that each node has reported to $\mathcal{A}$ what kinds of data it could generate when moving into $\mathcal{A}$'s cell. Let $\mathcal{U}$ denote the set of $n = |\mathcal{U}|$ users that $\mathcal{A}$ has selected and motivated to participate in data aggregation.[2] Finally, $\mathcal{A}$ broadcasts an aggregation request $\langle p, g, \mathcal{U}, r \rangle$, where r is a random nonce for message freshness. It is worth noting that the aggregation request can be sent as part of $\mathcal{A}$'s periodic service beacons and need be authenticated properly as in [36] to prevent attackers from sending fake aggregation requests, which we have ignored here for the focus of this paper. In addition, there can be various methods to transmit a condensed version of $\mathcal{U}$, which is also not discussed here for simplicity.

C. Commitment Submission

In this phase, each node $i \in \mathcal{U}$ submits to $\mathcal{A}$ a commitment, which is a homomorphic MAC of its datum d_i after appropriate expansion. In contrast to traditional MAC, a homomorphic MAC function $H(\cdot)$ has the additional property that the given the homomorphic MACs of two messages, say $H(m_1)$ and $H(m_2)$, anyone can derive $H(m_1 + m_2)$ without knowing m_1 or m_2. VPA$^+$ uses a simple homomorphic MAC construction as follows,

$$H(m) = g^m \mod p,$$

where $m \in [0, p-2]$. It is easy to see that $H(\cdot)$ is homomorphic because $\forall m_1, m_2 \in [0, p-2]$,

$$H(m_1 + m_2) = g^{m_1+m_2} = H(m_1)H(m_2) \mod p.$$

Before generating the commitment, each node i first need expand its datum d_i to introduce sufficient randomness. Note that the data range in many PC-USS applications is usually limited. For instance, in a traffic monitoring application, the driving speed is between 0 and 100 miles. If node i directly submits $H(d_i)$ to $\mathcal{A}$, then $\mathcal{A}$ can deduce d_i by exhaustive search. To avoid this situation, each node i expands d_i by adding a random number. In particular, assume that each datum d_i is of l bits. Node i generates a random number r_i of ϕ bits known only to itself and computes

$$e_i = 2^{l+\lfloor \log_2 n \rfloor} \cdot r_i + d_i \ . \tag{1}$$

where ϕ is a system parameter determining the difficulty of exhaustive search. Alternatively, we can view e_i as the concatenation of r_i, $\lfloor \log_2 n \rfloor$ zeros, and d_i as follows

$$e_i = r_i, \overbrace{0, \ldots, 0}^{\lfloor \log_2 n \rfloor}, d_i \ .$$

The reason to separate r_i and d_i by $\lfloor \log_2 n \rfloor$ zeros can be explained as follows. If we perform Sum aggregation over all e_i, then we have

$$\begin{aligned} \sum_{i \in \mathcal{U}} e_i &= 2^{l+\lfloor \log_2 n \rfloor} \cdot \sum_{i \in \mathcal{U}} r_i + \sum_{i \in \mathcal{U}} d_i \\ &\leq 2^{l+\lfloor \log_2 n \rfloor} \cdot \sum_{i \in \mathcal{U}} r_i + n(2^l - 1) \\ &< 2^{l+\lfloor \log_2 n \rfloor} \cdot \sum_{i \in \mathcal{U}} r_i + 2^{l+\lfloor \log_2 n \rfloor} \ . \end{aligned}$$

It follows that

$$\sum_{i \in \mathcal{U}} d_i = \sum_{i \in \mathcal{U}} e_i \mod 2^{l+\lfloor \log_2 n \rfloor} \ . \tag{2}$$

This property will be used later by $\mathcal{A}$ to derive the correct aggregation result without knowing $\{r_i\}_{i \in \mathcal{U}}$.

To prevent malicious nodes from submitting arbitrary data, we require that d_i and e_i be generated and authenticated by node i's TPM. Recall that $\mathcal{A}$ has assigned a secret key k_i to node i and another secret key κ_i to node i's TPM after mutual authentication (see Section III-A). Node i submits to $\mathcal{A}$ the following message.

$$i \rightarrow \mathcal{A} : i, \langle H(e_i), h(\kappa_i || H(e_i)) \rangle_{k_i} \ ,$$

where $h(\cdot)$ denotes a good hash function, and $\langle \cdot \rangle_*$ denotes a symmetric-key encryption operation using the key on the subscript.

On receiving the message, the AS locates k_i and κ_i using node ID i. It uses k_i to decrypt the message, and then verifies $h(\kappa_i || H(e_i))$ using κ_i. If the verification succeeds, $\mathcal{A}$ considers $H(e_i)$ an authentic commitment and drops it otherwise. If $\{H(e_i)\}_{i=1}^n$ are all authentic, the AS proceeds to derive the aggregate commitment corresponding to $\sum_{i=1}^n e_i$

[2] How $\mathcal{A}$ selects $\mathcal{U}$ from candidate users and appropriately stimulate their participation is an orthogonal topic deserving independent investigation.

by computing

$$H(\sum_{i\in\mathcal{U}} e_i) = \prod_{i\in\mathcal{U}} H(e_i) \mod p$$
$$= \prod_{i\in\mathcal{U}} g^{e_i} \mod p$$
$$= g^{\sum_{i\in\mathcal{U}} e_i} \mod p .$$

D. Privacy-Preserving In-Network Data Aggregation

In this phase, nodes jointly perform in-network aggregation over their expanded data without disclosing them. This phase requires the establishment of an on-demand temporary aggregation tree. In particular, the AS $\mathcal{A}$ broadcast an *aggregation-tree formation* request, which specifies any node, say $v \in \mathcal{U}$, as the root of the aggregation tree. On receiving the request, node v rebroadcasts it via its Bluetooth or WiFi interface, depending on the particular method (e.g., [40]) it uses to communicate with neighboring nodes. Upon receiving the request for the first time, each node further rebroadcasts it and records the parent node from which this request came from. In this way, an aggregation tree is formed and rooted at node v which can directly communicate with $\mathcal{A}$.

In what follows, we present two techniques for privacy-preserving in-network Sum aggregation over all expanded data with different user-privacy guarantees and communication overhead. To facilitate presentation, we define node i's *aggregation neighbors* as i's neighboring nodes on the aggregation tree, denoted by $\mathcal{T}_i$.

*1) Method 1: Data Perturbation (*DP*):* In this method, each node i perturbs its expanded datum e_i before actual aggregation. Since e_i is of $l + \lfloor \log_2 n \rfloor + \phi$ bits, we have

$$\sum_{i\in\mathcal{U}} e_i \le n \cdot (2^{l+\lfloor \log_2 n \rfloor+\phi} - 1)$$
$$< 2^{l+2\lfloor \log_2 n \rfloor+\phi} ,$$

i.e., that $\sum_{i\in\mathcal{U}} e_i$ is at most of $l + 2\lfloor \log_2 n \rfloor + \phi$ bits.

Denote by $h_1(\cdot)$ a good hash function of $l + 2\lfloor \log_2 n \rfloor + \phi$ bits. Each node i generates a perturbed datum α_i by computing

$$\alpha_i = h_1(k_i||r) + e_i \mod 2^{l+2\lfloor \log_2 n \rfloor+\phi} , \quad (3)$$

where k_i is the secret key shared between node i and the AS and r is the nonce broadcasted by $\mathcal{A}$.

Each node then performs in-network aggregation over its perturbed datum by adding it to the values received from its children on the aggregation tree and then transmitting the result to its parent. Finally, the AS $\mathcal{A}$ can obtain $\sum_{i\in\mathcal{U}} \alpha_i$ by summing the values received from the root of the aggregation tree, i.e., node v. Since $\mathcal{A}$ knows k_i for each $i \in \mathcal{U}$, it can compute all $h_1(k_i||r)$ and derive $\sum_{i\in\mathcal{U}} e_i$ by computing

$$\sum_{i\in\mathcal{U}} e_i = \sum_{i\in\mathcal{U}} \alpha_i - \sum_{i\in\mathcal{U}} h_1(k_i||r) \mod 2^{l+2\lfloor \log_2 n \rfloor+\phi} . \quad (4)$$

Since e_i is completely concealed by $h_1(k_i||r)$, which is only known by $\mathcal{A}$, other curious nodes, e.g., node i's neighbors on the aggregation tree, cannot derive e_i by monitoring i's incoming and outgoing transmission. Unfortunately, node i's data privacy can still be breached if $\mathcal{A}$ colludes with node i's aggregation neighbors.

2) Method 2: peer-to-peer slicing and mixing: To defend against $\mathcal{A}$ colluding with other curious nodes, we further propose another approach based on peer-to-peer data slicing and mixing. In this approach, before participating in in-network aggregation, each node i randomly divides its expanded datum e_i into multiple slices and mixes them with those from selected peers, such that data privacy can be preserved without affecting the correctness of the final aggregation result.

Specifically, before answering the query, each node i slices e_i into $t+1$ random slices $\{s_{i,j}\}_{j=1}^{t+1}$ with $t \le n-1$, such that

$$e_i = \sum_{j=1}^{t+1} s_{i,j} \mod 2^{l+2\lfloor \log_2 n \rfloor+\phi} .$$

Then node i keeps $s_{i,t+1}$ to itself while sending each other slice to a unique peer called a *cover node*. Next, each node i adds the slices received from other nodes to its remained slice $s_{i,t+1}$ and conducts in-network aggregation as in Method 1. Finally, $\mathcal{A}$ adds up all the received values. It is easy to see that the result is exactly the Sum aggregate of interest.

This slicing technique shares the similar idea as PDA [21], while our application scenario is totally different. In particular, PDA is designed for sensor networks with relatively static network topology, where all the nodes 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 holds in our target scenarios, 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. This significant difference necessitates novel cover-selection strategies. In what follows, we detail two cover-selection approaches that specially tailored for our target scenario.

*a) **Random cover selection** (*RCS*) :* In this method, each node randomly chooses t cover nodes from $\mathcal{U}$ and sends a data slice to each of them. VPA$^+$ uses the following method for choosing cover nodes. Consider node i as an example with data e_i to share. Node i first slices e_i into $\{s_{i,j}\}_{j=1}^{t+1}$ and broadcasts a *cover-discovery request* within the cell as follows.

$$i \rightarrow * : ID_i, r, x, y ,$$

where r is a random nonce, and x and y ($x > y$) are two integers jointly determining the ratio of cover nodes in the cell. On receiving the request, each node, say j, rebroadcasts the request and also checks if the following condition holds.

$$h(r||ID_j) \mod x < y \quad (5)$$

If so, node j considers itself as a candidate cover node chosen for node i and returns a *cover-discovery response* containing ID_j to node i via multi-hop communication, during which each intermediate node appends its ID to the response. By doing so, node i can discover the route to every candidate cover node. Suppose that there are currently z other nodes in the same cell. The expected number of candidate cover nodes with IDs satisfying Eq. (5) is zx/y. In practice, node i need

choose x and y such that zy/x is slightly larger than t to ensure that sufficient candidates can be selected.

On receiving the response from node j, node i first verifies if ID_j satisfies Eq. (5). If so, node i considers it as a candidate cover node. After receiving all the responses, node i randomly selects t cover nodes from all the candidate nodes, denoted by $\mathcal{C}_i \subseteq \mathcal{U}$. For any cover node $j \in \mathcal{C}_i$, node i computes a shared key $k_{i,j}$ based on its public/private keys ID_i/K_i^{-1} and ID_j by using the method in our previous work [38] and then sends an encrypted unique slice $s_{i,\tau_{i,j}}$ to node j as follows.

$$i \rightarrow j : ID_i, \langle s_{i,\tau_{i,j}}, h(s_{i,\tau_{i,j}}) \rangle_{k_{i,j}}$$

On receiving the message, node j can derive the same key $k_{i,j}$ using its public/private keys ID_j/K_j^{-1} and ID_i according to [38] and then decrypts the packet to get $s_{i,\tau_{i,j}}$. Node i repeats this process for all its cover nodes, and so does every other node in $\mathcal{U}$.

Each node waits for sufficient time to receive all the slices from other nodes choosing it as cover node. Let $\mathcal{S}_i \subset \mathcal{U}$ denote the set of nodes selecting i as a cover node. Each node i computes its share as

$$\beta_i = s_{i,t+1} + \sum_{j \in \mathcal{S}_i} s_{j,\tau_{j,i}} \mod 2^{l+2\lfloor \log_2 n \rfloor + \phi} . \quad (6)$$

Finally, all the nodes perform in-network aggregation over there shares as in Method 1 so that the AS $\mathcal{A}$ finally receives $\sum_{i \in \mathcal{U}} \beta_i$ which equals $\sum_{i \in \mathcal{U}} e_i$.

b) μ-hop cover selection (μCS): Random cover selection may not be efficient because cover nodes are randomly chosen regardless of their locations. As a result, a cover node discovery process is needed to find the route to each of the chosen cover nodes multi-hop away, which may cause unnecessarily high energy consumption.

Moreover, we observe that it is unnecessary for each node i to predetermine the slices $\{s_{i,j}\}_{j=1}^{t+1}$ and send each of them to a cover node. Instead, node i can broadcast a random seed within its μ-hop neighborhood, in which every node is chosen as a cover node and can compute a slice using their shared key.

Specifically, in μ-hop cover selection, each node i initiates the slicing process by broadcasting a *slicing request* with a random seed r_i and a TTL value set to μ. 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 from. In this way, a routing tree of depth μ 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 *slicing response* to its parent node which in turn forwards the response via the routing tree back to node i after appending its ID.

Each node waits sufficient time and then updates its share as follows. Consider node i as an example. Suppose that node i has received slicing responses from the set of nodes $\mathcal{C}_i$ and slicing requests from the set of nodes $\mathcal{S}_i$, i.e., the set of nodes choosing i as covers. Node i derives a shared key $k_{i,j}$ for each $j \in \mathcal{C}_i \bigcup \mathcal{S}_i$ according to [36] and updates its share by computing

$$\beta_i = e_i - \sum_{j \in \mathcal{C}_i} h_1(r_i || k_{i,j}) + \sum_{j \in \mathcal{S}_i} h_1(r_j || k_{i,j}) \mod 2^{l+2\lfloor \log_2 n \rfloor + \phi} . \quad (7)$$

Finally, all the nodes perform in-network aggregation over their shares so that the AS $\mathcal{A}$ finally obtains $\sum_{i \in \mathcal{U}} \beta_i$ which equals $\sum_{i \in \mathcal{U}} e_i$.

Unlike in random cover selection, the number of cover nodes in μ-hop cover selection is a random variable which cannot be determined before the process is completed. Intuitively, the larger μ, the more cover nodes discovered, the higher privacy and the communication cost, and vice versa.

E. Aggregation-Result Verification

After in-network aggregation via Method 1 or 2, the AS obtains $\sum_{i \in \mathcal{U}} e_i$. It first verifies its integrity by checking if

$$g^{\sum_{i \in \mathcal{U}} e_i} = \prod_{i \in \mathcal{U}} H(e_i) \mod p.$$

If so, $\mathcal{A}$ considers $\sum_{i \in \mathcal{U}} e_i$ authentic and proceeds to derive $\sum_{i \in \mathcal{U}} d_i$ by computing

$$\sum_{i \in \mathcal{U}} d_i = \sum_{i \in \mathcal{U}} e_i \mod 2^l ,$$

which should hold according to Eq. (2).

F. Performance Analysis

Now we analyze the performance of VPA$^+$ with regard to its aggregation-integrity provision, data-privacy guarantee, and the associated overhead.

1) Aggregation integrity: We first have the following theorem regarding the aggregation integrity of VPA$^+$.

Theorem 1: Assume that each node's datum is generated and authenticated by TPM and that $p > 2^{l+2\lfloor \log_2 n \rfloor + \phi}$. VPA$^+$ allows the AS to detect any false-data injection attack.

Proof: Assume that the AS receives $\{H(e_i) : i \in \mathcal{U}\}$ during the commitment submission phase, whereby it derives $H(\sum_{i \in \mathcal{U}} e_i) = g^{\sum_{i \in \mathcal{U}}} \mod p$. Suppose that some malicious nodes injected false data during the data-aggregation phase so that the AS receives $e' \neq \sum_{i \in \mathcal{U}} e_i$. The AS cannot detect the false-data injection attack if and only if

$$g^{e'} = g^{\sum_{i \in \mathcal{U}} e_i} \mod p .$$

However, for any $y \in \mathbb{Z}_p^*$, there is a unique $x \in [0, p-2]$ such that $g^x = y \mod p$. Since $p > 2^{l+2\lfloor \log_2 n \rfloor + \phi} > \sum_{i \in \mathcal{U}} e_i$, there is no $e' \neq \sum_{i \in \mathcal{U}} e_i$ can satisfy the above equation. It is thus impossible for the adversary to inject false data without being detected under VPA$^+$. ■

2) Data privacy: To evaluate the data-privacy provision of VPA$^+$, we define *exposure probability*, denoted by P_{exp}, as the probability that a node i's data d_i is disclosed during aggregation. To enable quantitative analysis, we assume that

each node has N_{tree} aggregation neighbors on average. We also assume that there are M_c out of M curious ASs and n_c out of n curious nodes.

We then have the following theorems regarding the exposure probability under VPA$^+$.

Theorem 2: The exposure probability under DP *is given by*

$$P_{\text{exp}} = \frac{M_c}{M} \cdot \frac{\binom{n-n_c}{n_c-N_{\text{tree}}}}{\binom{n}{n_c}} . \tag{8}$$

We give the proof in Appendix A.

Theorem 3: The exposure probabilities under RCS *and* μCS *are bounded by*

$$P_{\text{exp}} \leq \frac{\binom{n-n_c}{n_c-w}}{\binom{n}{n_c}} , \tag{9}$$

where

$$w = \begin{cases} \max(N_{\text{tree}}, t) & \text{for RCS,} \\ \max(N_{\text{tree}}, \sum_{x=1}^{\mu} N_x) & \text{for } \mu\text{CS,} \end{cases} \tag{10}$$

is the minimum number of nodes colluding with $\mathcal{A}$.
We give the proof in Appendix B.

3) Overhead Analysis: Now we analyze the computation and communication overhead incurred by VPA$^+$.

For computation overhead, each node need perform one exponentiation to generate one commitment of its data. Assuming that each commitment is of 1024 bits, it takes about 159 ms on a smartphone with 1 GHz CPU according to our implementation result. In addition, each node i need compute the shared key $k_{i,j}$ for each node $j \in \mathcal{C}_i \bigcup \mathcal{S}_i$ under RCS and μCS. Each key generation requires one pairing operation over the underlying group, which takes about 491 ms on a smartphone with 1 GHz CPU, according to the latest Java implementation benchmark [45]. We do not expect $|\mathcal{C}_i \bigcup \mathcal{S}_i|$ to be too large in practice, say less than 10, so the computation overhead incurred by VPA$^+$ is quite acceptable.

Assume that the average distance two random chosen nodes is L hops. Also denote by l_{tree}, l_{seed}, l_{hmac}, l_{h} the length of a aggregation tree formation request, a slicing request in μCS, a homomorphic MAC, and $h(\cdot)$, respectively. We then have the following theorem regarding the communication overhead incurred by VPA$^+$.

Theorem 4: The communication overhead incurred by VPA$^+$ in bits is given by

$$\mathsf{T}_{\text{VPA}^+} = nl_{\text{tree}} + \mathsf{T}_{\text{commit}} + \mathsf{T}_{\text{agg}} , \tag{11}$$

where

$$\mathsf{T}_{\text{commit}} = n(\lambda + l_{\text{hmac}} + l_{\text{h}}) \tag{12}$$

is the overhead incurred by transmitting commitments to $\mathcal{A}$*, and*

$$\mathsf{T}_{\text{agg}} = \begin{cases} nl_{\text{data}} & \text{for DP,} \\ n(nl_{\text{req}} + tL(l_{\text{rsp}} + l_{\text{data}} + \lambda)) & \\ \quad + nl_{\text{data}} & \text{for RCS,} \\ n((1 + \sum_{x=1}^{\mu-1})N_x(\lambda + l_{\text{seed}}) & \\ \quad + \sum_{x=1}^{\mu} N_x\lambda) + nl_{\text{data}} & \text{for } \mu\text{CS,} \end{cases} \tag{13}$$

is the overhead incurred by in-network aggregation, and $l_{\text{data}} = l + 2\lfloor \log_2 n \rfloor + \phi$.

We give the proof in Appendix C.

V. VPA$^{\oplus}$: Verifiable Privacy-Preserving Non-additive Aggregation

VPA$^+$ cannot be directly applied to non-additive aggregation functions such as Max/Min, Median, Percentile, and Histogram, which have wide applications in practice. In this section, we propose VPA$^{\oplus}$ as an extension of VPA$^+$ to support non-additive aggregation.

A. Basic Idea

Our key observation is that all the above non-additive aggregation functions are closely related to Count aggregation that ask for the number of nodes whose values are above, below, or equal to a certain value. In particular, let $\mathsf{Count}[Q]$ be the number of nodes with data satisfying the condition Q. Also denote by $d_{\max}, d_{\min}, d_{\text{med}}, d_{\sigma-\text{per}}$, the Max, Min, Median, and σ-percentile of a data set, respectively. It is easy to see that the following conditions hold.

- *Max*:
$$\begin{cases} \mathsf{Count}[d > d_{\max}] = 0, \\ \mathsf{Count}[d = d_{\max}] > 0 . \end{cases} \tag{14}$$
- *Min*:
$$\begin{cases} \mathsf{Count}[d < d_{\min}] = 0, \\ \mathsf{Count}[d = d_{\min}] > 0 . \end{cases} \tag{15}$$
- *Median*:
  - If n is odd, then
  $$\begin{cases} \mathsf{Count}[d \leq d_{\text{med}}] \geq \lfloor n/2 \rfloor, \\ \mathsf{Count}[d \geq d_{\text{med}}] \geq \lfloor n/2 \rfloor . \end{cases} \tag{16}$$
  - If n is even, then there exists $i, j \in \mathcal{U}$, such that $d_i \leq d_j$ and
  $$\begin{cases} \mathsf{Count}[d \leq d_i] \geq n/2, \\ \mathsf{Count}[d < d_i] < n/2, \\ \mathsf{Count}[d \geq d_j] \geq n/2, \\ \mathsf{Count}[d > d_j] < n/2 , \end{cases} \tag{17}$$
  and $d_{\text{med}} = (d_i + d_j)/2$.
- *σ-percentile*: we only show the simplest case here
$$\begin{cases} \mathsf{Count}[d \leq d_{\sigma-\text{per}}] \geq \lfloor \sigma n/100 \rfloor, \\ \mathsf{Count}[d \geq d_{\sigma-\text{per}}] \geq \lfloor (100-\sigma) n/100 \rfloor . \end{cases} \tag{18}$$

Conversely, if we can find d^* such that the conditions in Eq. (14) (respectively, (15), (17), (18)) hold, then we have $d^* = d_{\max}$ (respectively, $d_{\min}$, d_{med}, $d_{\sigma-\text{per}}$). Since Count is an additive aggregation function, it can be realized by VPA$^+$. Built on the above observation, VPA$^{\oplus}$ combines VPA$^+$ with binary search to realize non-additive aggregation functions through a series of verifiable privacy-preserving Count aggregations.

B. Scheme Description

Given a non-additive aggregation request, $\mathcal{A}$ transforms it into a series of Count queries with conditions $Q_1, Q_2, \ldots$, until the desired d^* is found, where Q_x is determined by the result of the previous Count query with condition Q_{x-1}. Each Count query Q_x asks how many nodes possess data above, below, or equal to a threshold, called a *count index*. Each node i with datum d_i satisfying condition Q_x gives an answer "yes", or "no" otherwise, by a single bit of value one or zero, respectively. The answers are then aggregated via VPA$^+$ to let $\mathcal{A}$ get $\mathsf{Count}(Q_x)$ with both user-privacy and aggregation-integrity guarantees.

Below we brief how to realize privacy-preserving Max/Min, Median, Histogram, and Percentile aggregation queries under the assumption that each data value d_i is an integer between $[0, 2^l - 1]$. It is easy to extend our technique to other non-additive aggregation functions.

1) Max/Min: Since the Min operation is opposite to the Max operation, we just illustrate the latter for brevity. Given a Max aggregation request, $\mathcal{A}$ first issues a Count query with $Q_1 = [d \geq 2^{l-1}]$ and then aggregates the received data via VPA$^+$ to get the number of "yes" answers, denoted by θ_1. If $\theta_1 \geq 1$, the maximum value should be in $[2^{l-1}, 2^l - 1]$, so $\mathcal{A}$ will send a new Count query with $Q_2 = [d \geq 2^{l-1} + 2^{l-2}]$; otherwise, the maximum value should be in $[0, 2^{l-1} - 1]$, so $\mathcal{A}$ will send a new Count query with $Q_2 = [d > 2^{l-2}]$. 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 equals the number of nodes with the maximum value.

2) Median/Percentile: Since Median is a special case of Percentile, we illustrate the former for simplicity, which can be easily extended to the latter. A median value is described as the number separating the higher half of a sample, a population, or a probability distribution, from the lower half. Median aggregation can be realized in a similar fashion as Max. Here we present the case for n being odd for simplify, while the case of n being even can be realized accordingly.

Given a Median aggregation request, $\mathcal{A}$ first issues a Count query with $Q_1 = [d \geq 2^{l-1}]$ and obtains θ_1 via VPA$^+$. If $\theta_1 \geq (n+1)/2$, $\mathcal{A}$ sends the second Count query with $Q_2 = [d \geq 2^{l-1} + 2^{l-2}]$; otherwise, $\mathcal{A}$ sends the next query with $Q_2 = [d \geq 2^{l-2}]$. This process continues until the suspicion range of d_{med} is one, which takes total l queries. Suppose that the last two queries are $Q_{l-1} = [d \geq q_{l-1}]$ and $Q_l = [d \geq q_l]$ whereby $\mathcal{A}$ receives θ_{l-1} and θ_l, respectively. It follows that q_{l-1} and q_l differ by one. There are four cases.

- Case 1: if $q_{l-1} < q_l$ and $\theta_l \geq \lfloor n/2 \rfloor$, then we have $d_{\text{med}} = q_l$. The reasons are as follows. First, we must have $\theta_{l-1} < \lfloor n/2 \rfloor$, as otherwise $d_{\text{med}} \leq q_l - 1$, and q_l should not be queried. Second, there must exist a query $Q_x = [d \geq q_l + 1]$ with $x \in [1, l-2]$, due to the property of binary search. Third, it must hold that $\theta_x < \lfloor n/2 \rfloor$, as otherwise $d_{\text{med}} > q_l + 1$ and neither q_{l-1} nor q_l should be queried.
- Case 2: if $q_{l-1} > q_l$ and $\theta_l \geq \lfloor n/2 \rfloor$, then we have $d_{\text{med}} = q_l$.
- Case 3: if $q_{l-1} < q_l$ and $\theta_l < \lfloor n/2 \rfloor$, then $d_{\text{med}} = q_l + 1$.
- Case 4: if $q_{l-1} > q_l$ and $\theta_l < \lfloor n/2 \rfloor$, then $d_{\text{med}} = q_l + 1$.

The reasoning for Cases 2$\sim$4 are similar to that of Case 1 and is thus omitted.

3) Histogram: In statistics, a histogram is a graphical display of tabulated frequencies, shown as bars, and shows the proportion of cases falling into each of several categories. Using Count query to realize Histogram is straightforward. In particular, given a Histogram aggregation request, $\mathcal{A}$ partitions the data range $[0, 2^l - 1]$ 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 equal the number of nodes with data in that interval.

C. Performance Analysis

Since VPA$^\oplus$ is built upon VPA+, it can also ensure perfect aggregation integrity. We thus focus on analyzing the user-privacy provision and overhead of VPA$^\oplus$.

1) Data privacy: The exposure probability P_{exp} used to analyze the performance of VPA$^+$ can no longer precisely measure the privacy provision of non-additive aggregation. For example, even if the answer of node i to a Count query Q_x is disclosed, the adversary can only narrow down the search of d_i to a certain range instead of precisely determining d_i. Assume that the adversary knows that d_j is in a range of length ϵ after the whole query process. It is clear that the ratio $\rho = \epsilon / 2^l$ can be used to analyze the privacy performance of the non-additive aggregation process: the larger ρ, the higher level of privacy provision, and vice versa.

In particular, when $\rho = 1$, the adversary has no clue about what d_i is; when $\rho = 2^{-l}$, i.e., $\epsilon = 1$, the adversary has precisely located d_i. We call ρ the *suspicion ratio* of d_i 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.

We make the following assumptions for analytical tractability. We assume that the aggregation tree is static for the entire sequence of l Count queries. For clarity, we consider a special case where the maximum value $d_{max} = 2^l - 1$. The similar process can be used to analyze the more general case that d_{max} may be any value in $[0, 2^l - 1]$, which is not reported here due to space limitations. We then have the following theorems regarding the expected suspicion ratio of VPA$^\oplus$.

Theorem 5: Assume that $d_{max} = 2^l - 1$, the expected

suspicion ratio of VPA$^{\oplus}$ under DP *or* μCS *is given by*

$$\mathsf{E}[\rho] = 1 - P_{\text{exp}} + (2^{-2l} + \sum_{x=1}^{l} 2^{-2x}) P_{\text{exp}} \ , \tag{19}$$

where P_{exp} is the exposure probability of VPA^{+} that given in Eq. (8) for DP *and Eq. (9) for* μCS.

We give the proof in Appendix C.

Theorem 6: Assume that $d_{max} = 2^l - 1$, the expected suspicion ratio under of VPA$^{\oplus}$ under RCS *is given by*

$$\mathsf{E}[\rho] = \sum_{x=0}^{l-1} 2^{-x-1} \mathsf{E}[\rho_x] + 2^{-l} \mathsf{E}[\rho_l] \ , \tag{20}$$

where

$$\mathsf{E}[\rho_x] = \sum_{k_1=0}^{x} Pr(y_e = k_1) \sum_{k_2=x+1}^{l+1} Pr(n_e = k_2) \rho[y_e, n_e] \ , \tag{21}$$

$$Pr(y_e = k) = \begin{cases} (1 - P_{\text{exp}})^x & \text{if } k = 0, \\ P_{\text{exp}}(1 - P_{\text{exp}})^{k-1} & \text{if } 1 \le k \le x \ , \end{cases} \tag{22}$$

$$Pr(n_e = k) = \begin{cases} P_{\text{exp}}(1 - P_{\text{exp}})^{k-x-1} & \text{if } x+1 \le k \le l, \\ (1 - P_{\text{exp}})^{l-x} & \text{if } k = l+1 \ , \end{cases} \tag{23}$$

$$\rho[y_e, n_e] = \begin{cases} 2^{-y_e} - 2^{-n_e} & \text{if } y_e + 1 \le n_e \le l, \\ 2^{-y_e} & \text{if } n_e = l+1 \ , \end{cases} \tag{24}$$

P_{exp} is the exposure probability of VPA^{+} that given in Eq. (9) for RCS.

We give the proof in Appendix D.

2) Overhead Analysis: VPA$^{\oplus}$ differs from VPA+ mainly in the communication overhead. Since it takes l queries to complete the aggregation process, each of which incurs communication overhead of $\mathsf{T}_{\text{VPA}^+}$, we thus have

$$\mathsf{T}_{\text{VPA}^{\oplus}} = l \cdot \mathsf{T}_{\text{VPA}^+} \ ,$$

where $\mathsf{T}_{\text{VPA}^+}$ is given in Eq. (11).

VI. Performance Evaluation

In this section, we evaluate VPA+ and VPA$^{\oplus}$ using extensive simulations.

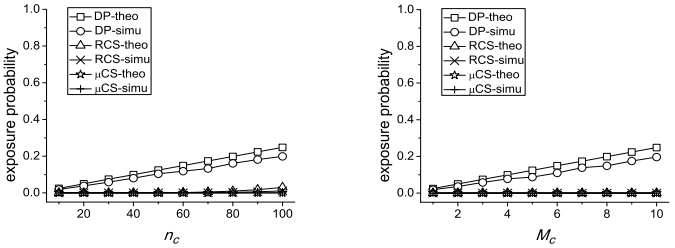
A. Simulation Setting

We simulate 10 cells of 1 km^2, each with an AS located at the center and 200 nodes randomly distributed within the cell. The transmission range of each node is 200m. This gives the average hop distance between two random nodes $L = 3.39$.

For our purpose, the simulation code is written in C++ and each data point represents an average of 50 simulation runs with different random seeds. Table I summarizes the default setting used in our simulation if not mentioned otherwise.

B. Evaluation of VPA^{+}

Fig. 2(a) shows both the theoretical and simulation results of the exposure probabilities of DP, RCS and μCS varying with n_c, the number of curious nodes. We can see that the exposure probabilities of all three schemes decrease as n_c increases. Among three schemes, DP has the highest exposure probability, followed by RCS and μCS. The reason is that on average, each node has only less than two neighbors on the aggregation tree (i.e., a spanning tree), making it easier for the adversary to compromise (or collude with) all the neighbors of a target node under DP. In contrast, it is much more difficult to compromise all the cover nodes under RCS and μCS. In addition, we can see that the P_{exp} of DP obtained via theoretical analysis is slightly higher than that obtained by simulations. The reason is that we round N_{tree} to $\lfloor N_{\text{tree}} \rfloor$ when computing $\binom{n-n_c}{n_c-N_{\text{tree}}}$ in Eq. (8), leading to higher P_{exp}.



(a) exposure probability (b) exposure probability

Fig. 2. Impact of n_c and M_c.

TABLE I
Default Simulation Settings

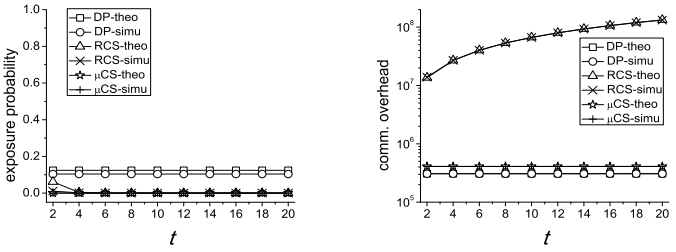
Para.	Val.	Para.	Val.	Para.	Val.	Para.	Val.
M	10	M_c	5	n	200	n_c	50
λ	8	ϕ	160	μ	1	t	5
N_1	20.9	N_2	39.3	N_3	48.1	N_4	48.2
l	10	L	3.39	l_{tree}	160	l_{seed}	160
l_{req}	160	l_{rsp}	160	l_{hmac}	1024	N_{tree}	1.86

Fig. 2(b) shows the impact of M_c, the number of curious ASs on the exposure probability of DP. Since curious ASs has no impact on RCS and μCS, there exposure probabilities are shown only for references. We can see that the exposure probability of DP increases linearly with the number of curious AS increases, which is expected.

Fig. 3(a) shows the impact of t, the number of cover nodes, on the exposure probability of RCS, where the P_{exp}s of DP and μCS are shown only for reference. We can see that the P_{exp} of RCS decreases as t increases, and quickly drops to zero when $t > 4$. The reason is that the probability of all the t cover nodes being compromised decreases exponentially as t increases.

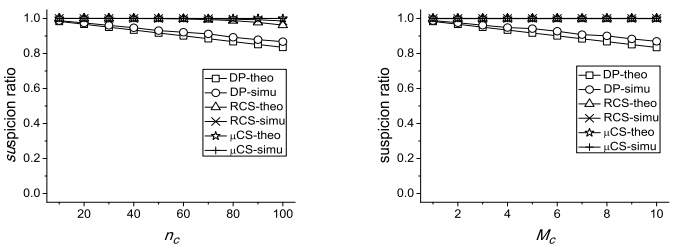
Fig. 3(b) shows the communication overhead of RCS varying with t. We can see that under the default settings, RCS incurs significantly higher communication overhead than that of DP and RCS. This is anticipated since finding a cover node under RCS requires an AODV-like route discovery that involves a network-wide flooding.

Fig. 4 shows the impact of μ on the exposure probability and communication overhead of μCS, where the results of DP and RCS are only shown for reference. We can see from Fig. 4(a) that the exposure probability of μCS is not much affected by μ because P_{exp} is already close to zero when $\mu = 1$. In addition,



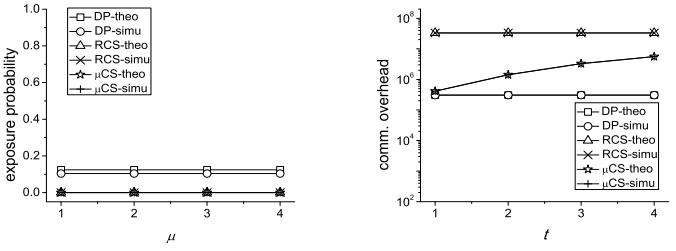
(a) exposure probability (b) communication overhead

Fig. 3. Impact of t, the number of cover nodes on RCS.



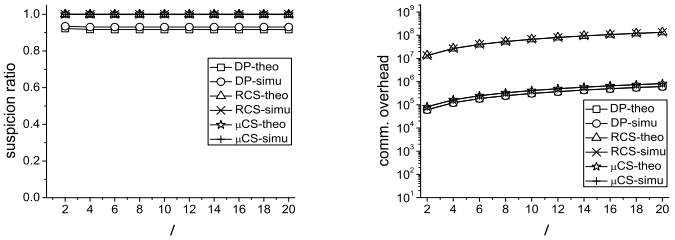
(a) suspicion ratio (b) suspicion ratio

Fig. 5. Impact of n_c and M_c on suspicion ratio.



(a) exposure probability (b) exposure probability

Fig. 4. Impact of μ.



(a) suspicion ratio (b) communication overhead

Fig. 6. Impact of l.

We can see that the communication overhead of μCS increases moderately as μ increases, which is of no surprise.

1) Evaluation of VPA$^{\oplus}$: Fig. 5(a) shows the suspicion ratios of DP, RCS and μCS, varying with n_c. We can see that the suspicion ratios of all three schemes decrease as n_c increases. The reason is that the higher n_c, the lower P_{exp}, and the lower suspicion ratio, and vice versa. In addition, under the default setting, μCS has the highest suspicion ratio, followed by that of RCS and DP.

Fig. 5(b) shows the impact of M_c on the suspicion ratio of DP, where the performance of RCS and μCS are only shown for reference. We cans see that the larger M_c, the lower suspicion ratio, and vice versa, which is easy to understand.

Fig. 6 shows the impact of l on the suspicion ratio and communication overhead of VPA$^{\oplus}$. We can see from Fig. 6(a) that the change in data range has negligible impact on the suspicion ratio of VPA$^{\oplus}$. The reason is that the suspicion ratio is determined by the last disclosed yes answer and the first disclosed no answer (cf. the proof of Theorem 6 in the Appendix D). Under the default setting, DP has the lowest suspicion ratio due to its highest P_{exp} among the three schemes (cf. Fig. 2(a)), while the suspicion ratios of both RCS and μCS are close to one. In addition, we can see from Fig. 6(b) that the communication overhead of VPA$^{\oplus}$ increases linearly as l increases, as it takes l Count queries to locate the desired aggregate.

C. Discussion

We summarize the evaluation results as follows.

- All three variants of VPA^{+} (i.e., DP, RCS, and μCS) can ensure aggregation integrity by detecting any false-data injection attempt.
- DP can provide user/data privacy with high probability while incurring the minimum communication overhead.
- RCS can provide user/data privacy against curious ASs with overwhelming probability while incurring the highest communication overhead.
- μCS can provide user/data privacy against curious ASs with overwhelming probability while incurring relatively low communication overhead.
- Built upon VPA^{+} and binary search, VPA$^{\oplus}$ can ensure both aggregation integrity and user/data privacy with communication overhead linear to the bit length of data.

In practice, μCS and the resulting VPA$^{\oplus}$ may be the best choices whose performance can be adjusted as needed.

VII. Conclusion

In this paper, we have presented the design and evaluation of VPA, a novel peer-to-peer approach to verifiable privacy-preserving aggregation for people-centric urban sensing systems. VPA can support a wide range of additive and non-additive aggregation functions with strong user-privacy and aggregation-integrity guarantees. The high efficacy and efficiency of VPA are confirmed by thorough theoretical analysis and simulation results.

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] J. Froehlich, T. Dillahunt, P. Klasnja, J. Mankoff, S. Consolvo, B. Harrison, and J. Landay, "UbiGreen: investigating a mobile tool for tracking and supporting green transportation habits," in *CHI'09*, Boston, MA, 2009, pp. 1043–1052.

[3] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, "VTrack: accurate, energy-aware road traffic delay estimation using mobile phones," in *ACM SenSys'08*, Berkeley, CA, Nov. 2009, pp. 85–98.
[4] S. Mathur, T. Jin, N. Kasturirangan, J. Chandrasekaran, W. Xue, M. Gruteser, and W. Trappe, "ParkNet: drive-by sensing of road-side parking statistics," in *MobiSys'10*, San Francisco, CA, 2010, pp. 123–136.
[5] P. Leijdekkers and V. Gay, "Personal heart monitoring and rehabilitation system using smart phones," in *ICMB'06*, Washington, DC, USA, 2006, pp. 29–36.
[6] 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.
[7] 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.
[8] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden, "CarTel: A distributed mobile sensor computing system," in *ACM SENSYS'06*, Boulder, CO, Oct. 2006, pp. 125–138.
[9] A. Parker, S. Reddy, T. Schmid, K. Chang, G. Saurabh, M. Srivastava, M. Hansen, J. Burke, D. Estrin, M. Allman, and V. Paxson, "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.
[10] 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.
[11] T. Abdelzaher, Y. Anokwa, P. Boda, J. Burke, D. Estrin, L. Guibas, A. Kansal, S. Madden, and J. Reich, "Mobiscopes for human spaces," *IEEE Pervasive Computing*, vol. 6, no. 2, pp. 20–29, 2007.
[12] V. Tuulos, J. Scheible, and H. Nyholm, "Combining web, mobile phones and public displays in large-scale: Manhattan story mashup," in *the Fifth International Conference on Pervasive Computing*, Toronto, Canada, May 2007, pp. 37–54.
[13] 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.
[14] A. Kapadia, D. Kotz, and N. Triandopoulos, "Opportunistic sensing: Security challenges for the new paradigm," in *COMSNETS'09*, Bangalore, India, Jan 2009.
[15] 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.
[16] R. N. Murty, G. Mainland, I. Rose, A. R. Chowdhury, A. Gosaint, J. Berst, and M. Welsh, "CitySense: An urban-scale wireless sensor network and testbed," in *IEEE HST'08*, Waltham, MA, May 2008, pp. 583–588.
[17] U. Moller, L. Cottrell, P. Palfrader, and L. Sassaman, "Mixmaster protocol ł version 2," IETF Internet Draft, July 2003.
[18] E. Cristofaro and C. Soriente, "PEPSI: privacy enhancing participatory sensing infrastructure," in *WiSec'11*, Hamburg, Germany, June 2011.
[19] J. Girao, M. Schneider, and D. Westhoff, "CDA: Concealed data aggregation in wireless sensor networks," in *ACM WiSe'04*, Philadelphia, PA, Oct. 2004.
[20] 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.
[21] 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, USA, May 2007, pp. 2045–2053.
[22] 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.
[23] 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.
[24] "Concealed data aggregation for reverse multicast traffic in sensor networks: Encryption, key distribution, and routing adaptation," *IEEE Transactions on Mobile Computing*, vol. 5, no. 10, pp. 1417–1431, Oct. 2006, dirk Westhoff and Joao Girao and Mithun Acharya.
[25] 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.
[26] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," in *ACM CCS'06*, Alexandria, Virginia, USA, Oct. 2006, pp. 278–287.
[27] 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.
[28] Y. Yang, X. Wang, S. Zhu, and G. Cao, "SDAP: a secure hop-by-hop data aggregation protocol for sensor networks," in *ACM MobiHoc'06*, Florence, Italy, May 2006, pp. 356–367.
[29] S. Roy, M. Conti, S. Setia, and S. Jajodia, "Securely computing an approximate median in wireless sensor networks," in *SecureComm'08*, Istanbul, Turkey, 2008, pp. 6:1–6:10.
[30] H. Yu, "Secure and highly-available aggregation queries in large-scale sensor networks via set sampling," in *IPSN'09*, Washington, DC, Apr. 2009, pp. 1–12.
[31] B. Chen and H. Yu, "Secure aggregation with malicious node revocation in sensor networks," in *ICDCS'11*, June 2011, pp. 581–592.
[32] K. Minami, A. J. Lee, M. Winslett, and N. Borisov, "Secure aggregation in a publish-subscribe system," in *WPES'08*, Alexandria, Virginia, USA, 2008, pp. 95–104.
[33] C. Wang, G. Wang, W. Zhang, and T. Feng, "Reconciling privacy preservation and intrusion detection in sensory data aggregation," in *INFOCOM'11*, Shanghai, China, Apr. 2011, pp. 336–340.
[34] R. A. Popa, A. J. Blumberg, H. Balakrishnan, and F. H. Li, "Privacy and accountability for location-based aggregate statistics," in *CCS'11*, Chicago, Illinois, USA, 2011, pp. 653–666.
[35] G. Ateniese, A. Herzberg, H. Krawczyk, and G. Tsudik, "Untraceable mobility or how to travel incognito," *Computer Networks*, vol. 31, no. 8, pp. 871–884, Apr. 1999.
[36] Y. Zhang and Y. Fang, "ARSA: an attack-resilient security architecture for multi-hop wireless mesh networks," *IEEE J. Select. Areas Commun., Special Issue on High-Speed Network Security - Architecture, Algorithms, and Implementation*, vol. 24, no. 10, pp. 1916–1928, Oct. 2006.
[37] X. Fu, N. Zhang, A. Pingley, W. Yu, J. Wang, and W. Zhao, "The digital Marauder's map: A new threat to location privacy," in *ICDCS'09*, Montreal, Quebec, Canada, June 2009, pp. 589–596.
[38] Y. Zhang, W. Liu, W. Lou, and Y. Fang, "Securing mobile ad hoc networks with certificateless public keys," *IEEE Transactions on Dependable and Secure Computing,*, vol. 3, no. 4, pp. 386–399, Oct.-Dec. 2006.
[39] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *CRYPTO'01*, Santa Barbara, CA, Aug. 2001, pp. 213–229.
[40] Z. Yang, B. Zhang, J. Dai, A. Champion, D. Xuan, and D. Li, "E-SmallTalker: A distributed mobile system for social networking in physical proximity," in *ICDCS'10*, Genoa, Italy, June 2010, pp. 468–477.
[41] A. Dua, N. Bulusu, W. chang Feng, and W. Hu, "Towards trustworthy parcitipatory sensing," in *USENIX HotSec*, Montreal, Canada, Aug. 2009.
[42] S. Saroiu and A. Wolman, "I am a sensor, and I approve this message," in *HotMobile'10*, Annapolis, Maryland, Mar. 2010, pp. 37–42.
[43] P. Gilbert, L. Cox, J. Jung, and D. Wetherall, "Toward trustworthy mobile sensing," in *ACM HotMobile'10*, Annapolis, MA, Feb. 2010, pp. 31–36.
[44] P. Gilbert, J. Jung, K. Lee, H. Qin, D. Sharkey, A. Sheth, and L. Cox, "Youprove: Authenticity and fidelity in mobile sensing," in *SenSys'11*, Seattle, WA, Nov. 2011.
[45] "jPBC library," http://gas.dia.unisa.it/projects/jpbc/benchmark.html.

APPENDIX A
PROOF OF THEOREM 2

Proof: Consider node i as an example. Under DP, node i's data e_i is exposed if the AS is curious and colludes with all the neighbors of node i in the aggregation tree, which are denoted by $\mathcal{T}_i$. The probability that the AS being curious is M_c/M. Assume that $|\mathcal{T}_i| = N_{\text{tree}}$ and that there are n_c curious nodes. The probability of all nodes in $\mathcal{T}_i$ being curious is $\binom{n_c}{N_{\text{tree}}}\binom{n-n_c}{n_c-N_{\text{tree}}}/\binom{n}{n_c}$. We therefore have

$$P_{\text{exp}} = \frac{M_c}{M} \cdot \frac{\binom{n-n_c}{n_c-N_{\text{tree}}}}{\binom{n}{n_c}} \ . \tag{25}$$

■

APPENDIX B
PROOF OF THEOREM 3

Proof: Under RCS and μCS, node i's data e_i is exposed if all the nodes in $\mathcal{C}_i$, $\mathcal{S}_i$ and $\mathcal{T}_i$ are curious, where $\mathcal{S}_i$ is the set of nodes that choose node i as a cover. For RCS, we have $|\mathcal{C}_i| = t$ and for μCS, we have $|\mathcal{C}_i| = \sum_{x=1}^{\mu} N_x$, where N_x is the number of the x-hop neighbors of node i. Since $\max(|\mathcal{C}_i|, |\mathcal{S}_i|, |\mathcal{T}_i|) \leq |\mathcal{C}_i \bigcup \mathcal{S}_i \bigcup \mathcal{T}_i|$, we have

$$P_{\text{exp}} \leq \frac{\binom{n-n_c}{n_c-w}}{\binom{n}{n_c}}$$

where

$$w = \begin{cases} \max(N_{\text{tree}}, t) & \text{for RCS,} \\ \max(N_{\text{tree}}, \sum_{x=1}^{\mu} N_x) & \text{for } \mu\text{CS .} \end{cases}$$

■

APPENDIX C
PROOF OF THEOREM 4

Proof: The communication overhead incurred by VPA$^+$ consists of three parts: T_{tree}, the overhead incurred by forming aggregation tree, $\mathsf{T}_{\text{commit}}$, the overhead incurred by submitting commitment to $\mathcal{A}$, and T_{agg}, the overhead incurred by privacy-preserving aggregation.

We first estimate T_{tree}. During the aggregation-tree formation process, each node broadcasts the tree formation request once. Therefore we have

$$\mathsf{T}_{\text{tree}} = n l_{\text{tree}} ,$$

where l_{tree} is the length of the tree formation request in bits.

Now we estimate $\mathsf{T}_{\text{commit}}$. Suppose that $H(e)$ and $h(\cdot)$ are of l_{hmac} and l_{h} bits, respectively. The length of each commitment message is $l_{\text{hmac}} + l_{\text{h}}$. Recall that each node ID is of λ bits. We then have

$$\mathsf{T}_{\text{commit}} = n(\lambda + l_{\text{hmac}} + l_{\text{h}}) .$$

Finally, we estimate T_{agg} for DP, RCS and μCS.

- DP: during the aggregation phase, each node need transmit the intermediate aggregation result to its parent node. We therefore have

  $$\mathsf{T}_{\text{agg}} = n l_{\text{data}} ,$$

  where $l_{\text{data}} = l + 2\lfloor \log_2 n \rfloor + \phi$.
- RCS: before in-network aggregation, each node need select t cover nodes and discover the route to each of them. For simplicity, we assume that the number of candidate cover nodes is t. The route discovery incurs a communication overhead of $n l_{\text{req}} + tL l_{\text{rsp}}$, where l_{req} and l_{rsp} denote the length of a cover discovery request and a response, respectively. In addition, transmitting one slice incurs a communication overhead of $L(l_{\text{data}} + \lambda)$. Therefore we have

  $$\mathsf{T}_{\text{agg}} = n(n l_{\text{req}} + tL(l_{\text{rsp}} + l_{\text{data}} + \lambda))$$

- μCS: For μCS, during slicing and mixing, each node i and all the nodes within its $\mu-1$-hop neighborhood need broadcast a random seed along with ID_i once, which incurs communication overhead of $(1 + \sum_{x=1}^{\mu-1}) N_x(\lambda + l_{\text{seed}})$. In addition, each nodes in its $\mu - 1$-hop neighborhood need return its node ID to node i, which leads to communication overhead $\sum_{x=1}^{\mu} N_x \lambda$. We therefore have

  $$\mathsf{T}_{\text{agg}} = n((1 + \sum_{x=1}^{\mu-1} N_x)(\lambda + l_{\text{seed}}) + \sum_{x=1}^{\mu} N_x x \lambda) + n l_{\text{data}} .$$

■

APPENDIX D
PROOF OF THEOREM 5

Before we prove Theorem 5, we first have the following lemmas.

Lemma 1: Assume that $d_{max} = 2^l - 1$. Let $\{\chi_{i,j}\}_{j=1}^{l}$ be the sequence of answers from a node i, where $\chi_{i,j} \in \{0, 1\}$ for all $j \in [1, l]$. If $\chi_{i,j} = 1$, then $\chi_{i,j'} = 1$ for all $j' < j$; likewise, if $\chi_{i,j} = 0$, then $\chi_{i,j'} = 0$ for all $j' > j$.

Proof: Assuming that $d_{\max} = 2^l - 1$, we have $Q_j = [d \geq \sum_{i=1}^{j} 2^{l-i}]$ for all $j \in [1, l]$. It follows that $Q_{j+1} \subset Q_j$ for all $j \in [1, l-1]$. Therefore, if $\chi_{i,x} = 1$, then $d_i \in Q_j$. It follows that $d_i \in Q_{j'}$ and $\chi_{i,j'} = 1$ for all $j' < j$. The second part can be proved similarly and is thus omitted. ■

Lemma 2: Assume that $d_{max} = 2^l - 1$. Let χ_{i,y_e} and χ_{i,n_e} be the first exposed yes answer and last exposed no answer, where $0 \leq y_e < n_e \leq l + 1$, $y_e = 0$ and $n_e = l + 1$ denote the case that no yes answer is disclosed and the case that no no answer is exposed, respectively. The suspicion ratio after l count queries is given by

$$\rho[y_e, n_e] = \begin{cases} 2^{-y_e} - 2^{-n_e} & \text{if } y_e + 1 \leq n_e \leq l, \\ 2^{-y_e} & \text{if } n_e = l + 1. \end{cases} \tag{26}$$

Proof: Suppose $Q_{y_e} = [d \geq \sum_{j=1}^{y_e} 2^{l-j}]$ and $\chi_{i,y_e} = 1$. The disclosure of $\chi_{i,y_e} = 1$ let the adversary know that $d_i \geq \sum_{j=1}^{y_e} 2^{l-j}$. Similarly, suppose that n_e exists, i.e., at least one no answer is exposed. The disclosure of $\chi_{i,n_e} = 0$ let the adversary know that $d_i < \sum_{j=1}^{n_e} 2^{l-j}$. It follows that

$$\sum_{j=1}^{y_e} 2^{l-j} \leq d_i < \sum_{j=1}^{n_e} 2^{l-j} .$$

Therefore, we have

$$\begin{aligned} \rho[y_e, n_e] &= 2^{-l}(\sum_{j=1}^{n_e} 2^{l-j} - \sum_{j=1}^{y_e} 2^{l-j}) \\ &= \sum_{j=1}^{n_e} 2^{-j} - \sum_{j=1}^{y_e} 2^{-j} \\ &= \sum_{j=y_e+1}^{n_e} 2^{-j} \\ &= 2^{-y_e} - 2^{-n_e} . \end{aligned}$$

Note that the disclosure of other answers does not give the adversary additional information, since $Q_{j+1} \subset Q_j$ for all $j \in [1, l-1]$.

In addition, if no "no" answer is disclosed, we have

$$\rho = 2^{-y_e} .$$

■

Now we prove Theorem 4.

Proof: We first consider DP and μCS in which each node i directly interacts with the same set nodes during the whole sequence of l count queries. This means that if $\chi_{i,1}$ is exposed, which happens with probability P_{exp}, then all the subsequent answers $\{\chi_{i,1}\}_{j=2}^{l}$ are also exposed. On the other hand, if $\chi_{i,1}$ is kept secret, which happens with probability $1 - P_{\text{exp}}$, so are $\{\chi_{i,1}\}_{j=2}^{l}$.

We further partition the data range $[0, 2^l - 1]$ into $l + 1$ *equivalent classes*, denoted by $\{C_x\}_{x=0}^{l}$, where $C_x = [2^l - 2^{l-x}, 2^l - 2^{l-x-1} - 1]$ for $x \in [0, l-1]$, and $C_l = [2^l - 1, 2^l - 1]$. It follows if $d_i \in C_x$, then $\chi_{i,j} = 1$ for all $j \in [1, x]$, and $\chi_{i,j} = 0$ for all $j \in [x + 1, l]$. In other words, different nodes have the same sequence of answers if their data are in the same equivalent classes.

Assume that $d_i \in C_x$. If $\{\chi_{i,1}\}_{j=1}^{l}$ are exposed, which happens with probability P_{exp}, then we have $y_e = x$ and $n_e = x + 1$. According to Eq. (26), we have

$$\rho = \begin{cases} 2^{-x-1} & \text{if } 0 \le x \le l-1, \\ 2^{-l} & \text{if } x = l, \end{cases}$$

in this case. If $\{\chi_{i,1}\}_{j=1}^{l}$ are kept secret, which happens with probability $1 - P_{\text{exp}}$, then we have $\rho = 1$.

Combining the above two cases, we can compute the expected suspicion ratio as

$$\begin{aligned} \rho &= \sum_{x=0}^{l-1} Pr(d_i \in C_x)((1 - P_{\text{exp}}) + 2^{-x-1} P_{\text{exp}}) \\ &\quad + Pr(d_i \in C_l)((1 - P_{\text{exp}}) + 2^{-l} P_{\text{exp}}) \\ &= \sum_{x=0}^{l-1} 2^{-x-1}((1 - P_{\text{exp}}) + 2^{-x-1} P_{\text{exp}}) \\ &\quad + 2^{-l}((1 - P_{\text{exp}}) + 2^{-l} P_{\text{exp}}) \\ &= 1 - P_{\text{exp}} + (2^{-2l} + \sum_{x=1}^{l} 2^{-2x}) P_{\text{exp}} \, . \end{aligned} \tag{27}$$

Now we consider RCS, in which each node i chooses the cover nodes independently for each query. This means that each $\chi_{i,j}$ is exposed independently with probability P_{exp}. Suppose $d_i \in C_x$. So node i returns x yes answers and $l - x$ no answers. The p.d.f. of y_e is then given by

$$Pr(y_e = k) = \begin{cases} (1 - P_{\text{exp}})^x & \text{if } k = 0, \\ P_{\text{exp}}(1 - P_{\text{exp}})^{k-1} & \text{if } 1 \le k \le x, \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

Similarly, p.d.f. of n_e can be computed as

$$Pr(n_e = k) = \begin{cases} P_{\text{exp}}(1 - P_{\text{exp}})^{k-x-1} & \text{if } x + 1 \le k \le l, \\ (1 - P_{\text{exp}})^{l-x} & \text{if } k = l + 1, \\ 0 & \text{otherwise.} \end{cases} \tag{29}$$

For all data in $d_i \in C_x$, the expected suspicion ratio can be computed as

$$\mathsf{E}[\rho_x] = \sum_{k_1=0}^{x} Pr(y_e = k_1) \sum_{k_2=x+1}^{l+1} Pr(n_e = k_2) \rho[y_e, n_e] \, . \tag{30}$$

Finally, the expected suspicion ratio can be computed as

$$\begin{aligned} \mathsf{E}[\rho] &= \sum_{x=0}^{l} Pr(d_i \in C_x) \mathsf{E}[\rho_x] \\ &= \sum_{x=0}^{l-1} 2^{-x-1} \mathsf{E}[\rho_x] + 2^{-l} \mathsf{E}[\rho_l] \, . \end{aligned} \tag{31}$$

■