PDA: Privacy-preserving Data Aggregation in Wireless Sensor Networks

Wenbo He [*] Xue Liu [†] Hoang Nguyen [*] Klara Nahrstedt[*] Tarek Abdelzaher[*]

[*] Department of Computer Science
University of Illinois at Urbana-Champaign
Champaign, IL, 61801, United States

[†] School of Computer Science
McGill University
Montreal, Quebec H3A 2A7, Canada

***Abstract*—Providing efficient data aggregation while preserving data privacy is a challenging problem in wireless sensor networks research. In this paper, we present two privacy-preserving data aggregation schemes for additive aggregation functions. The first scheme – *Cluster-based Private Data Aggregation (CPDA)*– leverages clustering protocol and algebraic properties of polynomials. It has the advantage of incurring less communication overhead. The second scheme – *Slice-Mix-AggRegaTe (SMART)*– builds on slicing techniques and the associative property of addition. It has the advantage of incurring less computation overhead. The goal of our work is to bridge the gap between collaborative data collection by wireless sensor networks and data privacy. We assess the two schemes by privacy-preservation efficacy, communication overhead, and data aggregation accuracy. We present simulation results of our schemes and compare their performance to a typical data aggregation scheme – *TAG*, where no data privacy protection is provided. Results show the efficacy and efficiency of our schemes. To the best of our knowledge, this paper is among the first on privacy-preserving data aggregation in wireless sensor networks.**

I. Introduction

A wireless sensor network (WSN) is an ad-hoc network composed of small sensor nodes deployed in large numbers to sense the physical world. Wireless sensor networks have very broad application prospects including both military and civilian usage. They include surveillance [1], tracking at critical facilities [2], or monitoring animal habitats [3]. Sensor networks have the potential to radically change the way people observe and interact with their environment.

Sensors are usually resource-limited and power-constrained. They suffer from restricted computation, communication, and power resources. Sensors can provide fine-grained raw data. Alternatively, they may need to collaborate on in-network processing to reduce the amount of raw data sent, thus conserving resources such as communication bandwidth and energy. We refer to such in-network processing generically as *data aggregation*. In many sensor network applications, the designer is usually concerned with aggregate statistics such as *SUM*, *AVERAGE*, or *MAX/MIN* of data readings over a certain region or period. As a result, data aggregation in WSNs has received substantial attention.

As sensor network applications expand to include increasingly sensitive measurements of everyday life, preserving data privacy becomes an increasingly important concern. For example, a future application might measure household details such as power and water usage, computing average trends and making local recommendations. Without providing proper privacy protection, such applications of WSNs will not be practical, since participating parties may not allow tracking their private data. In this paper, we discuss how to carry privacy-preserving data aggregation in wireless sensor networks. In the following, we first elaborate two specific motivating applications of using wireless sensor network to carry out private data aggregation.

1) As alluded above, wireless sensors may be placed in houses to collect statistics about water and electricity consumption within a large neighborhood. The aggregated population statistics may be useful for individual, business, and government agencies for resource planning purposes and usage advice. However, the readings of sensors could reveal daily activities of a household, such as when all family members are gone or when someone is taking a shower (different water appliances have distinct signatures of consumption that can reveal their identity). Hence we need a way to collect the aggregated sensor readings while at the same time preserve data privacy.
2) Future in-home floor sensors, collecting weight information, are used together with shoe-mounted sensors, collecting exercise-related information, in an obesity study to correlate exercise and weight loss. Aggregate statistics from those data are useful for agencies such as Department of Health and Human Services, as well as insurance companies for medical research and financial planning purposes. However, individual's health data should be kept private and not be known to other people.

From these data aggregation examples, we see why preserving the privacy of individual sensor readings while obtaining accurate aggregate statistics can be an important requirement. The protection of privacy also gives us add-on benefits including enhanced security. Consider the scenario when an adversary compromises a portion of the sensor nodes: when there is no privacy protection, the comprised nodes can overhear the data messages and decrypt them to get sensitive information. However, with privacy protection, even if data are overheard and decrypted, it is still difficult for the adversary to recover sensitive information.

Consequently, providing a reasonable guideline on building systems that perform private data aggregation is desirable. It is well-known that end-to-end data encryption is able to protect private communications between two parties (such as the data source and data sink), as long as the two parties have agreement on encryption keys. However, end-to-end encryption or link level encryption alone is *not* a good candidate for private data aggregation. This is because:

1) If end-to-end communications are encrypted, the intermediate nodes could not easily perform in-network processing to get aggregated results.
2) Even when data are encrypted at the link level, the other end of the communication is still able to decrypt it and get the private data. Hence privacy is violated.

Though research on privacy-preserving computation has been active in other domains including cryptography and data mining, previously-studied schemes are not readily applicable to private data aggregations in WSNs. Most of them are either not suitable for or too computational-expensive to be used in the resource-constrained sensor networks, as we will discuss in detail in Section II.

In this paper, we present two privacy-preserving data aggregation schemes called *Cluster-based Private Data Aggregation (CPDA)* and *Slice-Mix-AggRegaTe (SMART)* respectively, for additive aggregation functions in WSNs. The goal of our work is to bridge the gap between collaborative data aggregation and data privacy in wireless sensor networks. When there is no packet loss, in both *CPDA* and *SMART*, the sensor network can obtain a *precise* aggregation result while guaranteeing that no private sensor reading is released to other sensors. Observe that this is a stronger result than previously proposed protocols that are able to compute *approximate* aggregates only (without violating privacy). Our presented schemes can be built on top of existing secure communication protocols. Therefore, both security and privacy are supported by the proposed data aggregation schemes.

In the *CPDA* scheme, sensor nodes are formed randomly into clusters. Within each cluster, our design leverages algebraic properties of polynomials to calculate the desired aggregate value. At the same time, it guarantees that no individual node knows the data values of other nodes. The intermediate aggregate values in each cluster will be further aggregated (along an aggregation tree) on their way to the data sink. In the *SMART* scheme, each node hides its private data by slicing it into pieces. It sends encrypted data slices to different intermediate aggregation nodes. After the pieces are received, intermediate nodes calculate intermediate aggregate values and further aggregate them to the sink. In both schemes, data privacy is preserved while aggregation is carrying out.

We evaluate the two schemes in terms of efficacy of privacy preservation, communication overhead, and data aggregation accuracy, comparing them with a commonly used data aggregation scheme *TAG* [4], where no data privacy is provided. Simulation results demonstrate the efficacy and efficiency of our schemes.

The rest of the paper is organized as follows. Section II summarizes the related work. Section III describes the model and requirements of privacy-preserving data aggregation in wireless sensor networks. Section IV provides our two algorithms for private data aggregation. Section V evaluates the proposed schemes. We summarize our findings and lay out future research directions in Section VI.

II. Related Work

In typical wireless sensor networks, sensor nodes are usually resource-constrained and battery-limited. In order to save resources and energy, data must be aggregated to avoid overwhelming amounts of traffic in the network. There has been extensive work on data aggregation schemes in sensor networks, including [4], [5], [6], [7], [8], [9]. These efforts share the assumption that all sensors are trusted and all communications are secure. However, in reality, sensor networks are likely to be deployed in an untrusted environment, where links, for example, can be eavesdropped. An adversary may compromise cryptographic keys and manipulate the data.

Work presented in [10], [11], [12] investigates secure data aggregation schemes in the face of adversaries who try to tamper with nodes or steal the information. Work presented in [13], [14] shows how to set up secret keys between sensor nodes to guarantee secure communications. For most existing secure data aggregation schemes though, an intermediate aggregation node has to decrypt the received data, then aggregate the data according to the corresponding aggregation function, and finally encrypt the aggregated result before forwarding it. This sequence is fairly expensive for data aggregation in sensor networks. To reduce computational overhead, Girao et al. [15] and Castelluccia et al. [16] propose using homomorphic encryption ciphers, which allow efficient aggregation of encrypted data without decryption involved in the intermediate nodes. Though these schemes are efficient to preserve data privacy in data aggregation, they do not protect the the trend of private data of a node from being known by its neighboring nodes. This is because when the neighboring nodes can always overhear the sum of the private data and an fixed unknown number (encryption key). In contrast, the private data aggregation schemes we present in this paper ensures that no trend about private data of a sensor node is released to any other nodes.

In privacy-preservation domain, Huang, Wang and Borisov address the problem in a peer-to-peer network application in [17]. Privacy preservation has also been studied in the data mining domain [18], [19], [20], [21]. Two major classes of schemes are used. The first class is based on data perturbation (randomization) techniques. In a data perturbation scheme, a random number drawn from a certain distribution is added to the private data. Given the distribution of the random perturbation, recovering the aggregated result is possible. At the same time, by using the randomized data to mask the private values, privacy is achieved. However, data perturbation techniques have the drawback that they do not yield accurate aggregation results. Furthermore, as shown by Kargupta et al. in [20] and by Huang et al. in [21], certain types of data perturbation might not preserve privacy well.

Another class of privacy-preserving data mining schemes [22], [23], [24] is based on Secure Multi-party Computation (SMC) techniques [25], [26], [27]. SMC deals with the problem of a joint computation of a function with multi-party private inputs. SMC usually leverages public-key cryptography. Hence SMC-based privacy-preserving data mining schemes are usually computationally expensive, which is not applicable to resource-constrained wireless sensor networks.

As we will show in the rest of this paper, unlike previous privacy-preserving approaches, our new private data aggregation schemes have the advantages: (1) They preserve data privacy such that individual sensor data is only known to their owner; (2) The aggregation result is accurate when there is no data loss; (3) They are more efficient and hence more suitable for resource-constrained wireless sensor networks.

III. Model and Background

A. Sensor Networks and the Data Aggregation Model

In this paper, a sensor network is modeled as a connected graph $G(\mathcal{V}, \mathcal{E})$, where sensor nodes are represented as the set of vertices $\mathcal{V}$ and wireless links as the set of edges $\mathcal{E}$. The number of sensor nodes is defined as $|V| = N$.

A data aggregation function is defined as $y(t) \triangleq f(d_1(t), d_2(t), \cdots, d_N(t))$, where $d_i(t)$ is the individual sensor reading at time t for node i. Typical functions of f include *sum*, *average*, *min*, *max* and *count*. If $d_i (i = 1, \cdots, N)$ is given, the computation of y at a query server (data sink) is trivial. However, due to the large data traffic in sensor networks, bandwidth constraints on wireless links, and large power consumption of packet transmition[1], data aggregation techniques are needed to save resources and power.

In this paper, we focus on additive aggregation functions, that is, $f(t) = \sum_{i=1}^{N} d_i(t)$. It is worth noting that using additive aggregation functions is not too restrictive, since many other aggregation functions, including *average*, *count*, *variance*, *standard deviation* and any other *moment* of the measured data, can be reduced to the additive aggregation function *sum* [16].

B. Requirements of Private Data Aggregation

Protecting the data privacy in many wireless sensor network applications is a major concern. The following criteria summarize the desirable characteristics of a private data aggregation scheme:

1) **Privacy:** Each node's data should be only known to itself. Furthermore, the private data aggregation scheme should be able to handle to some extent attacks and collusion among compromised nodes. When a sensor network is under a malicious attack, it is possible that some nodes may collude to uncover the private data of other node(s). Furthermore, wireless links may be eavesdropped by attackers to reveal private data. A good private data aggregation scheme should be robust to such attacks.
2) **Efficiency:** The goal of data aggregation is to reduce the number of messages transmitted within the sensor network, thus reduce resource and power usage. Data aggregation achieves bandwidth efficiency by using in-network processing. In private data aggregation schemes, additional overhead is introduced to protect privacy. However, a good private data aggregation scheme should keep that overhead as small as possible.
3) **Accuracy:** An accurate aggregation of sensor data is desired, with the constraint that no other sensors should know the exact value of any individual sensor. Accuracy should be a criterion to estimate the performance of private data aggregation schemes.

C. Key Setup for Encryption

To set context for our work, in this section, we first briefly review a random key distribution mechanism proposed in [13], on which our proposed schemes operate.

Security Assumptions and Key Setup:

In the new private data aggregation algorithms – *CPDA* and *SMART*– some messages are encrypted to prevent attackers from eavesdropping. Our schemes can be built on top of existing key distribution and encryption schemes in wireless sensor networks. Here, we briefly review a random key distribution mechanism proposed in [13] which we use in the design of our schemes.

In [13], key distribution consists of three phases: (1)key pre-distribution, (2)shared-key discovery, and (3)path-key establishment. In the pre-distribution phase, a large *key-pool* of K keys and their corresponding identities are generated. For each sensor within the sensor network, k keys are randomly drawn from the *key-pool*. These k keys form a *key ring* for a sensor node. During the key-discovery phase, each sensor node finds out which neighbors share a common key with itself by exchanging discovery messages. If two neighboring nodes share a common key then there is a secure link between two nodes. In the path-key establishment phase, a path-key is assigned to the pairs of neighboring sensor nodes who do not share a common key but can be connected by two or more multi-hop secure links at the end of the shared-key discovery phase.

In the random key distribution mechanism mentioned above, the probability that any pair of nodes possess at least one common key is:

$$p_{connect} = 1 - \frac{((K-k)!)^2}{(K-2k)!K!}. \tag{1}$$

Let the probability that any other node can overhear the encrypted message by a given key be $p_{overhear}$. It is the probability that a third node possesses the same key as this node. Therefore,

$$p_{overhear} = \frac{k}{K}. \tag{2}$$

[1] A Berkeley mote consumes approximately the same amount of energy to compute 800 instructions as it does in sending a single bit of data [4].

The key distribution algorithm discussed above is efficient in terms of using a small number of keys to support secure communication in a large-scale sensor network, hence preventing eavesdroping. This is illustrated in the following numerical example.

Assume a key pool of size $K = 10000$, and key ring size of $k = 200$. The probability that any pair of nodes can find a shared key in common is $p_{connect} = 98.3\%$ by Equation (1). In other words, the probability that a pair of nodes does not share a common key is 1.7%. For these pairs who do not share a common key, they can use the path-key establishment procedure described above to establish a shared key. Once a pair of nodes select a shared key, the probability that any other node owns the same key is $p_{overhear} = \frac{k}{K} = 0.2\%$, which is very small.

IV. Private Data Aggregation Protocols

In this section, we present two private data aggregation protocols focusing on additive data aggregation. The first scheme is called *Cluster-based Private Data Aggregation (CPDA)*. It consists of three phases: cluster formation, calculation of the aggregate results within clusters, and cluster data aggregation. The second scheme is called "*Slice-Mix-AggRegaTe (SMART)*". In *SMART*, each node hides its private data by slicing the data and sending encrypted data slices to different aggregators. Then the aggregators collect and forward data to a query server. When the server receives the aggregated data, it calculates the final aggregation result.

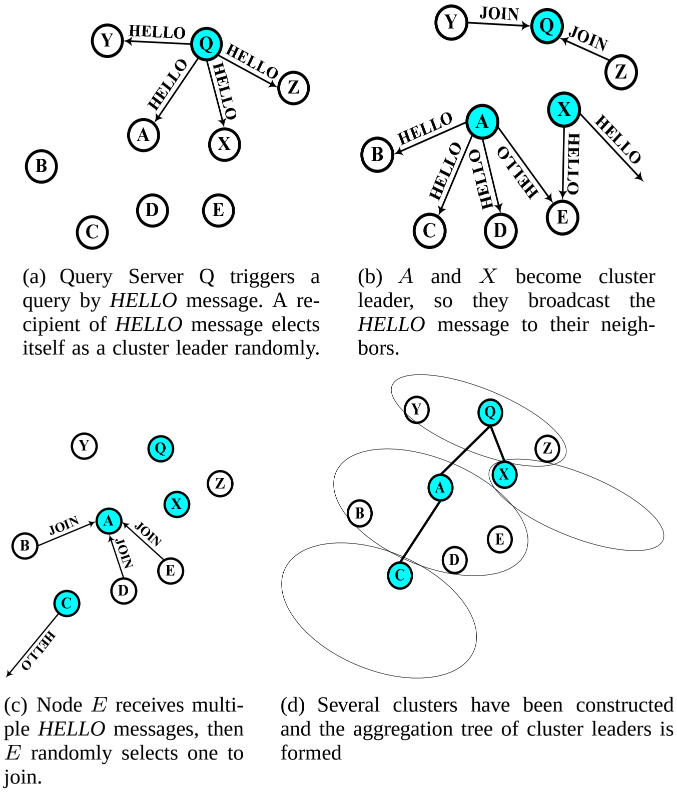
A. Cluster-based Private Data Aggregation (CPDA)

1) **Formation of Clusters:** The first step in *CPDA* is to construct clusters to perform intermediate aggregations. We propose a distributed protocol for this purpose.

The cluster formation procedure is illustrated in Figure 1. A query server Q triggers a query by a *HELLO* message. Upon receiving the *HELLO* message, a sensor node elects itself as a cluster leader with a probability p_c, which is a preselected parameter for all nodes. If a node becomes a cluster leader, it will forward the *HELLO* message to its neighbors; otherwise, the node waits for a certain period of time to get *HELLO* messages from its neighbors, then it decides to join one of the clusters by broadcasting a *JOIN* message. As this procedure goes on, multiple clusters are constructed.

2) **Calculation within Clusters:** The second step of *CPDA* is the intermediate aggregations within clusters. To simplify the discussion, we use a simple scenario, where a cluster contains three members: A, B, and C. a, b and c represent the private data held by nodes A, B and C, respectively. Let A be the cluster leader of this cluster. Let B and C be cluster members. Our privacy-preserving aggregation protocol based on the additive property of polynomials. Figure 2 illustrates the message exchange among the three nodes to obtain the desired sum without releasing individual private data.

First, nodes within a cluster share a common (non-private) knowledge of non-zero numbers, refer to as *seeds*, x, y, and z, which are distinct with each other (as shown in Figure 2(1)). Then node A calculates



(a) Query Server Q triggers a query by *HELLO* message. A recipient of *HELLO* message elects itself as a cluster leader randomly.

(b) A and X become cluster leader, so they broadcast the *HELLO* message to their neighbors.

(c) Node E receives multiple *HELLO* messages, then E randomly selects one to join.

(d) Several clusters have been constructed and the aggregation tree of cluster leaders is formed

Fig. 1. Formation of clusters

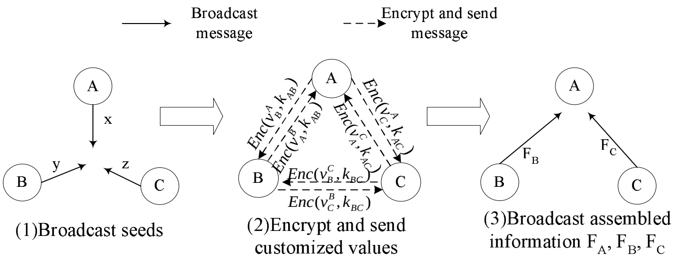


Fig. 2. Message exchange

$$
\begin{aligned}
v_A^A &= a + r_1^A x + r_2^A x^2, \\
v_B^A &= a + r_1^A y + r_2^A y^2, \\
v_C^A &= a + r_1^A z + r_2^A z^2,
\end{aligned}
$$

where r_1^A and r_2^A are two random numbers generated by node A, and known only to node A. Similarly, node B and C calculate v_A^B, v_B^B, v_C^B and v_A^C, v_B^C, v_C^C independently as:

$$
\begin{aligned}
NodeB: v_A^B &= b + r_1^B x + r_2^B x^2, \\
v_B^B &= b + r_1^B y + r_2^B y^2, \\
v_C^B &= b + r_1^B z + r_2^B z^2.
\end{aligned}
$$

$$
\begin{aligned}
NodeC: v_A^C &= c + r_1^C x + r_2^C x^2, \\
v_B^C &= c + r_1^C y + r_2^C y^2, \\
v_C^C &= c + r_1^C z + r_2^C z^2.
\end{aligned}
$$

Then node A encrypts v_B^A and sends to B using the shared key between A and B. It also encrypts v_C^A and sends to C using the sharing key between A and C (Figure 2(2)). Similarly node B encrypts and sends v_A^B to A and v_C^B to C; node C encrypts and sends v_A^C to A and v_B^C to B. When node A receives v_A^B and v_A^C, it has the knowledge of $v_A^A = a + r_1^A x + r_2^A x^2$, $v_A^B = b + r_1^B x + r_2^B x^2$ and $v_A^C = c + r_1^C x + r_2^C x^2$. Next, node A calculates assembled value $F_A = v_A^A + v_A^B + v_A^C = (a + b + c) + r_1 x + r_2 x^2$, where $r_1 = r_1^A + r_1^B + r_1^C$ and $r_2 = r_2^A + r_2^B + r_2^C$. Similarly node B and C calculate their assembled values $F_B = v_B^A + v_B^B + v_B^C = (a+b+c) + r_1 y + r_2 y^2$ and $F_C = v_C^A + v_C^B + v_C^C = (a + b + c) + r_1 z + r_2 z^2$ respectively. Then node B and C broadcast F_B and F_C to the cluster leader A (Figure 2(3)). So far, node A knows all the assembled values:

$$\begin{aligned} F_A = v_A^A + v_A^B + v_A^C = (a+b+c) + r_1 x + r_2 x^2, \\ F_B = v_B^A + v_B^B + v_B^C = (a+b+c) + r_1 y + r_2 y^2, \\ F_C = v_C^A + v_C^B + v_C^C = (a+b+c) + r_1 z + r_2 z^2. \end{aligned} \tag{3}$$

Then the cluster leader A can deduce the aggregate value $(a + b + c)$. This is because x, y, z, F_A, F_B, F_C are known to A. By rewriting Equation (3) as

$$U = G^{-1} F, \tag{4}$$

where $G = \begin{bmatrix} 1 & x & x^2 \\ 1 & y & y^2 \\ 1 & z & z^2 \end{bmatrix}$, $U = \begin{bmatrix} a+b+c \\ r_1 \\ r_2 \end{bmatrix}$, and $F = [F_A, F_B, F_C]^T$, $a+b+c$ is known as the first element of U. Note that G is of full rank, because x, y and z are distinct numbers.

It is necessary to encrypt v_B^A, v_C^A, v_A^B, v_C^B, v_A^C, and v_B^C. For example, if node C overhears the value v_B^A, then C knows v_B^A, v_C^A, and F_A, then C can deduce $v_A^A = F_A - v_B^A - v_C^A$, and further it can obtain a if x, v_A^A, v_B^A, v_C^A are known. However, if node A encrypts v_B^A and sends it to node B, then node C cannot get v_B^A. With only v_C^A, F_A and x from node A, node C cannot deduce the value of a. However, if nodes B and C collude by releasing A's information (v_B^A and v_C^A) to each other, then A's data will be disclosed. To prevent such collusion, the cluster size should be large. In a cluster of size m, if less than $(m - 1)$ nodes collude, the data won't be disclosed.

3) **Cluster Data Aggregation:** A common technique for data aggregation is to build a routing tree. We implement *CPDA* on top of the TAG Tiny AGgregation [4] protocol. Each cluster leader routes the derived sum within the cluster back towards the query server through a TAG routing tree rooted at the server.

4) **Discussions on Parameter Selection in *CPDA*:** In *CPDA*, a larger cluster size introduces a larger computational overhead (Equation (4). However, a larger cluster size is preferred for the sake of improved privacy under node collusion attacks. In *CPDA*, we should guarantee a cluster size $m \geq 3$. Generally, let's define m_c as the minimum cluster size. We should set $m_c \geq 3$. Next, we discuss how to ensure every cluster has a cluster size larger than m_c, and how to tune parameter p_c to reduce communication overhead in *cluster formation* phase.

If a cluster C_i has a size smaller than m_c, ($|C_i| < m_c$), the cluster leader of C_i needs to broadcast a "merge" request to join another cluster. In the following, we show that given a proper p_c, the percentage of clusters that need to merge is small, and the cluster size is in a reasonable range.

We model a sensor network as a random network, assuming d_i is the degree of a node i. If the node i is the cluster leader of a cluster of C_i, then the probability that a neighbor of i joins the C_i is

$$p_i = P(\text{a neighbor of } i \text{ joins } C_i) = (1 - p_c)\frac{1}{d_i p_c}, \tag{5}$$

where $1 - p_c$ is the probability that the neighbor is not a leader of another cluster. Only in this case is the neighbor able to join C_i. A neighbor is surrounded by $d_i p_c$ cluster leaders including i, therefore $\frac{1}{d_i p_c}$ is the probability that a non-leader neighbor of i joins C_i. The probability that cluster C_i has k members is:

$$P(|C_i| = k) = \begin{pmatrix} d_i \\ k-1 \end{pmatrix} {p_i}^{(k-1)} (1 - p_i)^{d_i - k + 1}. \tag{6}$$

Therefore, the percentage of clusters that need to merge is given by:

$$\begin{aligned} P(|C_i| < m_c) &= \sum_{k=1}^{m_c - 1} P(|C_i| = k) \\ &= \sum_{k=0}^{m_c - 2} \begin{pmatrix} d_i \\ k \end{pmatrix} {p_i}^k (1 - p_i)^{d_i - k}. \end{aligned} \tag{7}$$

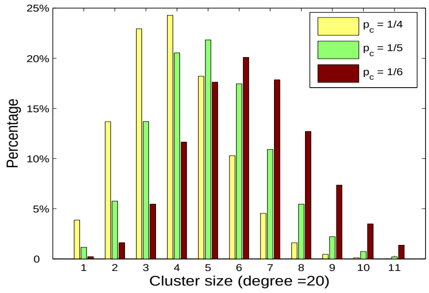


Fig. 3. Distribution of cluster size with different p_c

For a regular network with degree 20 ($d_i = 20$), $P(|C_i| < 3) = 6.9\%$ if $p_c = 1/5$; $P(|C_i| < 3) = 1.8\%$ if $p_c = 1/6$. Figure 3 shows that the distribution of cluster size can be controlled by parameter p_c without merging. By local observation of any sensor node, the number of clusters is $(d_i + 1)p_c$. On the other hand, if we desire k nodes in each cluster, then the desired cluster size should be $\frac{d_i+1}{k}$. Therefore, if we target the cluster size around k, and choose $p_c = \frac{1}{k}$.

B. Slice-Mix-AggRegaTe (SMART)

One drawback of the cluster based protocol is the computational overhead of data aggregation within clusters (Equation (4)). In this section, we present a new scheme *SMART*, which reduces computational overhead at the cost of slightly increased communication bandwidth consumption. As the name suggests, "Slice-Mix-AggRegaTe (*SMART*)" is a three-step scheme for private-preserving data aggregation.

Step 1 ("Slicing"): Each node i ($i = 1, \cdots, N$), randomly selects a set of nodes S_i ($J = |S_i|$) within h hops. For a dense WSN, we can take $h = 1$. Node i then slices its private data d_i randomly into J pieces (i.e., represents it as a sum of J numbers).

One of the J pieces is kept at node i itself. The remaining $J - 1$ pieces are encrypted and sent to nodes in the randomly selected set S_i. We denote d_{ij} as a piece of data sent from node i to node j. For nodes to which node i does not send any slice, $d_{ij} = 0$. The desired aggregate result can be expressed as

$$f = \sum_{i=1}^{N} d_i = \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij}, \tag{8}$$

where $d_{ij} = 0, \forall j \notin S_i$.

Step 2 ("Mixing"): When a node j receives an encrypted slice, it decrypts the data using its shared key with the sender. Upon receiving the first slice, the node waits for a certain time, which guarantees that all slices of this round of aggregation are received. Then, it sums up all the received slices $r_j = \sum_i^N d_{ij}$, where $d_{ij} = 0,\ j \notin S_i$.

Step 3 ("Aggregation"): All nodes aggregate the data and send the result to the query server. Similar to the aggregation step of *CPDA*, the aggregation is designed using tree-based routing protocols. When a node gets all data slices, it forwards a message of the sum addressed to its parent, which in turn forwards the message along the tree. Eventually the aggregation reaches the root (query server). Since

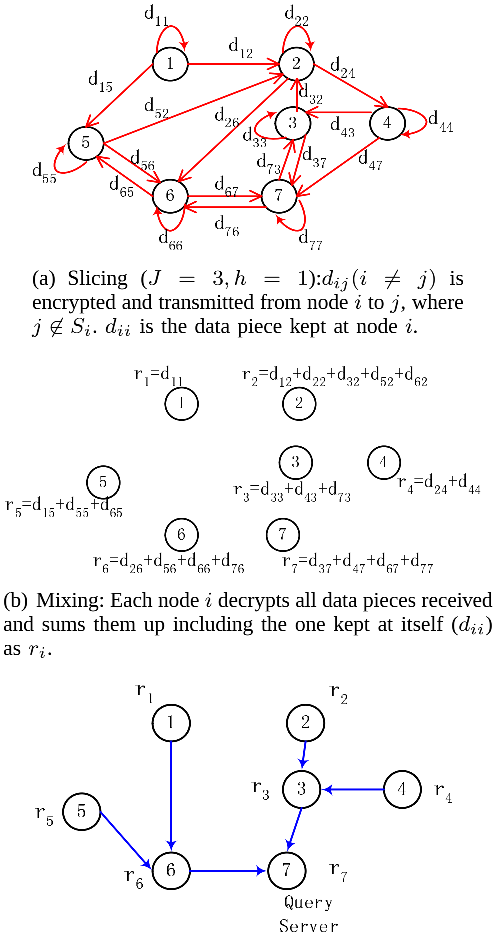
$$\sum_{j=1}^{N} r_j = \sum_{j=1}^{N} \sum_{i=1}^{N} d_{ij} = \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij}. \tag{9}$$

The final data at the root is the aggregation of all sensor data f by Equation (8) and (9).

Figure 4 illustrates the 3-step scheme of the *SMART* protocol for a sensor network with network size $N = 7$, slicing size $J = 3$, and hop length $h = 1$. For *SMART*, in step 1, sliced data should be encrypted as in *CPDA*.

V. Evaluation

In this section we evaluate the private-preserving data aggregation schemes presented in this paper. We evaluate how our schemes perform in terms of privacy-preservation, efficiency, and aggregation accuracy. We use *TAG* [4], a typical data aggregation scheme as the baseline. Since the design of *TAG* does not take privacy into consideration, no data privacy protection is provided. We only use it to evaluate the efficiency and aggregation accuracy compared with our proposed schemes.



(a) Slicing ($J = 3, h = 1$):$d_{ij}(i \neq j)$ is encrypted and transmitted from node i to j, where $j \notin S_i$. d_{ii} is the data piece kept at node i.

(b) Mixing: Each node i decrypts all data pieces received and sums them up including the one kept at itself (d_{ii}) as r_i.

(c) Aggregation (No encryption is needed)

Fig. 4. Illustration of three steps in *SMART*

A. Privacy-preservation Efficacy

In order to evaluate the performance of privacy-preservation, we first define the privacy metric. In wireless sensor networks, private data of a sensor node s may be disclosed to others when attackers can eavesdrop on communication and/or collude. That is, there are two cases that may lead to privacy violation: (1) An unauthorized sensor node holds a communication key and is able to decrypt messages sent from s. Under our key distribution mechanism, the probability that an eavesdropper has the communication key used by s and one of its neighbors is $p_{overhear}$ (Equation (2)). (2) Multiple neighbors of s collude to steal private data collected by s. We can assume the probability that any two nodes collude is $p_{collude}$.

For the simplicity of derivation, let us define $p_{overhear} = p_{collude} \triangleq q$. q is interpreted as the probability that the link level privacy is broken. A privacy metric $\mathbb{P}(q)$ is defined as the probability that the private data of node s is disclosed for a given q under either conditions above. $\mathbb{P}(q)$ measures the performance of the privacy-preservation of a private data aggregation scheme.

1) Privacy-preservation Analysis of CPDA*:* In the *CPDA* scheme, private data may be disclosed to neighbors only when the sensor nodes exchange messages within the same cluster. Given a cluster of size m, a node needs to send $m-1$ encrypted messages to other $m-1$ members within the cluster. Only if a node knows all $m-1$ keys of a given member, can it crack the private data of the member. Otherwise, the private data cannot be disclosed. Consequently, $\mathbb{P}(q)$ is estimated as

$$\mathbb{P}(q) = \sum_{k=m_c}^{d_{max}} P(m=k)(1-(1-q^{k-1})^k), \tag{10}$$

where d_{max} is the maximum cluster size. m_c is the required minimum cluster size. $P(m=k)$ represents the probability that a cluster size is k.

2) Privacy-preservation Analysis of SMART: In the *SMART* scheme, a sensor node s slices its private data into J pieces and then encrypts and sends $J-1$ pieces to its neighbors. It keeps one piece to itself. As a result, the out-degree of s is $J-1$ and the in-degree of s is the number of neighbors who encrypt and send data pieces to s. Only if an eavesdropper breaks $J-1$ outgoing links and all incoming links of a node s, will it be able to crack the private data held by s. Therefore, $\mathbb{P}(q)$ can be approximated by

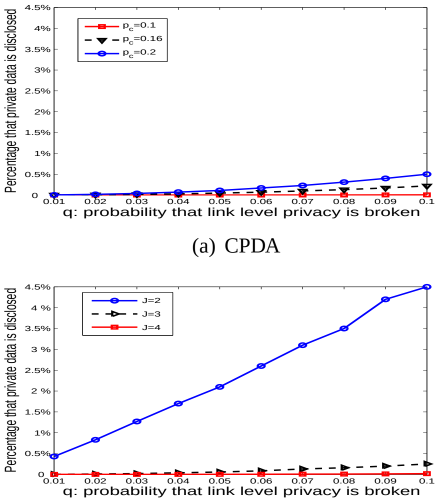
$$\mathbb{P}(q) = q^{x-1} \sum_{k=0}^{d_{max}} P(in-degree=k)\ q^k, \tag{11}$$

where d_{max} is the maximum in-degree in a network. $P(in-degree=k)$ is the probability that the in-degree of a node is k.

Figure 5 compares privacy-preservation performance of *CPDA* and *SMART* via simulation, where we consider a 1000-node random network. The average degree of a node is 16. As we can see from Figure 5, for *CPDA*, the smaller the value of p_c (the probability of a node independently becoming a cluster leader), the larger the average cluster size, hence the better the privacy-preservation performance is. However, if a cluster size is larger, the computational overhead to compute the intermediate aggregation value by Equation (4) will also be larger. In *SMART*, the larger the value of J (the number of slices each node chooses to decompose its private data), the better privacy can be achieved. However, a larger J will also yield larger communication overhead. For both *CPDA* and *SMART*, there is a design tradeoff between the privacy protection and computation/communication efficiency.

B. Communication Overhead

CPDA and *SMART* use data-hiding techniques and encrypted communication to protect data privacy. This introduces some communication overhead. In order to investigate bandwidth efficiency of these schemes, we implemented *CPDA* and *SMART* in *ns2* on top of the data aggregation component of *TAG*. We did extensive simulations and collected results to compare these two schemes together with *TAG* (no privacy protection). In our experiments, we consider networks with 600 sensor nodes. These nodes are randomly deployed over a $400 meters \times 400 meters$ area. The transmission range of a sensor node is 50 meters and data rate is 1 *Mbps*.



(a) CPDA

(b) SMART

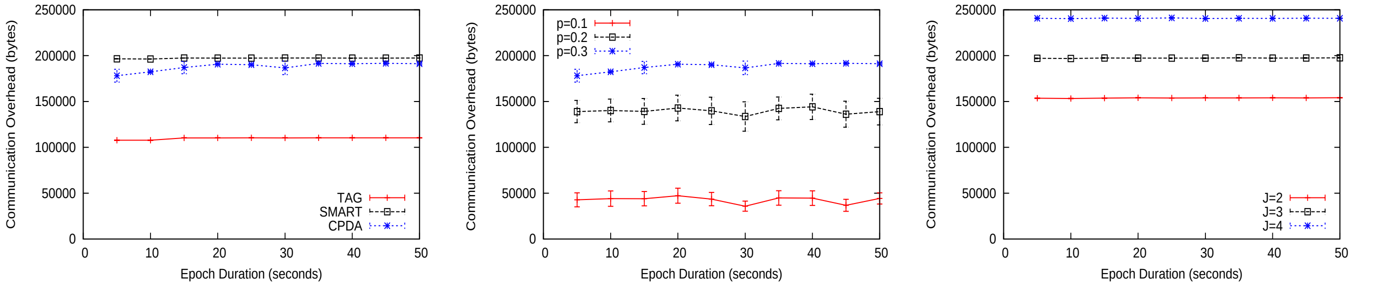
Fig. 5. $\mathbb{P}(q)$ under *CPDA* and *SMART*.

At the beginning of each simulation, a query is delivered from the query server to the sensor nodes. Similar to *TAG* [4], the query specifies an *epoch duration* E, which is the amount of time for the data aggregation procedure to finish. Upon receiving such a query, a parent node on the aggregation tree subdivides the epoch such that its children are required to deliver their data (protected data in *CPDA* and *SMART*, or unprotected data in *TAG*) in this parent-defined time interval.

Figure 6(a) shows the communication overhead of *TAG*, *CPDA* with $p_c = 0.3$, and *SMART* with J=3 under different epoch durations. We use the total number of bytes of all packets communicated during the aggregation as the metric. Each point in the figure is the average result of 50 runs of the simulation. In each run, one randomly generated sensor network topology is used. The vertical line of each data point represents the 95% confidence interval of the data collected.

Simulation results can be explained by analyzing the number of exchanged messages in each scheme. In *TAG*, each node needs to send 2 messages for data aggregation: one *Hello* message to form an aggregation tree, and one message for data aggregation. In our implementation of *CPDA*, a cluster leader sends roughly 4 messages and cluster members sends 3 messages for private data aggregation. Accordingly, $4p_c + 3(1-p_c) = 3 + p_c$ is the average number of messages sent by a node in *CPDA*. Thus, the message overhead in *CPDA* is less than twice as that in *TAG*. *SMART*, with $J = 3$, needs to exchange 2 messages during the slicing step and 2 messages for data aggregation (the same as *TAG*). Hence, each node needs 4 messages for the private data aggregation. Therefore, the overhead of *SMART* is double that of *TAG*.

Now let us further study the effect of p_c on the communication overhead in *CPDA*. Figure 6(b) shows the result with $p_c = 0.1, 0.2, 0.3$ respectively. As we can see, the larger the p_c value, the larger the communication overhead. It is very interesting to notice that when $p_c = 0.1$, communication

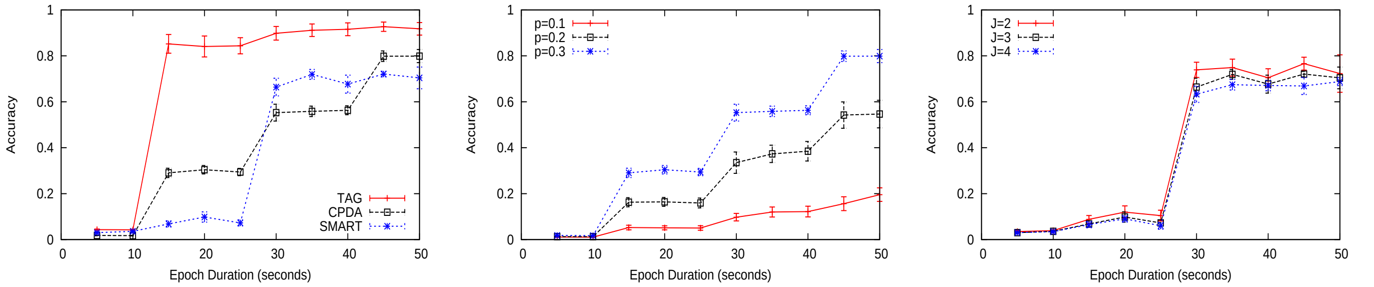


(a) Comparison of TAG, CPDA ($p_c = 0.3$) and SMART (J=3).

(b) Communication overhead of CPDA with respect to p_c.

(c) Communication overhead of SMART with respect to J.

Fig. 6. Communication overhead



(a) Accuracy comparison of TAG, CPDA ($p_c = 0.3$) and SMART (J=3).

(b) Accuracy of CPDA with respect to p_c.

(c) Accuracy of SMART with respect to J.

Fig. 7. Accuracy under collision and packet loss

overhead is much lower than *TAG*. This is because when p_c is too small, many nodes cannot be covered due to insufficient number of cluster leaders. This also explains why accuracy is very low when $p_c = 0.1$ (in Section V-C).

Finally, let us study the effect of J on the communication overhead in *SMART*. Figure 6(c) shows the result with $J = 2, 3, 4$ respectively. As we can see, the larger the J value, the larger the communication overhead. This is because J represents the number of slices each node chooses to decompose its private data into. Since, in slicing phase of SMART, each node sends $J - 1$ pieces of sliced data to its selected neighbors. Including one message for tree formation and one for aggregation, the total number of messages exchanged is roughly proportional to $J + 1$. Hence the larger the value of J, the larger the communication overhead.

C. Accuracy

In ideal situations when there is no data loss in the network[2], both *CPDA* and *SMART* should get 100% accurate aggregation results. However, in wireless sensor networks, due to collisions over wireless channels and processing delays, messages may get lost or delayed. Therefore, the aggregation accuracy is affected. We define the accuracy metric as the ratio between the collected sum by the data aggregation scheme used and the real sum of all individual sensor nodes. A higher accuracy value means the collected sum using the specific aggregation scheme is more accurate. An accuracy value of 1.0 represents the ideal situation.

Figure 7(a) shows the accuracy of *TAG*, *CPDA* (with $p_c = 0.3$) and *SMART* (with J=3) from our simulation. Here we have two observations. First, the accuracy increases as the epoch duration increases. Two reasons contribute to this: 1) With a larger epoch duration, the data packets to be sent within this duration will have less chance to collide due to the increased average packet sending intervals; 2) With a larger epoch duration, the data packets will have a better chance of being delivered within the deadline. The second observation is that *TAG* has better accuracy than *CPDA* and *SMART*. That is because without the communication overhead introduced by privacy-preservation, there will be less data collisions.

Figure 7(b) shows the aggregation accuracy of *CPDA* with respect to the selection of p_c. First, we see when using the same p_c, a larger epoch duration gives better accuracy. This is due to the fact that a larger epoch duration lets the data packets have a better chance of being delivered before the timeout. Second, we see that *CPDA* is sensitive to p_c values. The larger the p_c value, the higher the aggregation accuracy. This is because: (1)The larger p_c value is, the smaller portion of nodes are disconnected to query server through aggregation tree. Those nodes uncovered by aggregation tree cannot contribute their value in aggregation. (2)A larger p_c usually yields a smaller cluster size, which causes less collisions within the cluster under the same epoch duration. Therefore, we recommend $0.2 \leq p_c \leq 0.3$ in *CPDA* protocol.

[2] Data loss may be caused by collision in wireless channels, deadline missing or disconnection to the query server through an aggregation tree

Figure 7(c) illustrates the aggregation accuracy of *SMART* with respect to the selection of J. Accuracy of *SMART* is not sensitive to J. However, there is a slightly difference between different J values: the larger the value of J, the lower the aggregation accuracy. This is because when a private data held by a node is sliced into more pieces, more messages are needed to send all $J-1$ pieces to other neighboring nodes. Hence, more collisions occur, which causes a reduction in the aggregation accuracy. We recommend $J=3$ in *SMART* protocol.

VI. Concluding Remarks

Providing efficient data aggregation while preserving data privacy is a challenging problem in wireless sensor networks. Many civilian applications require privacy, without which individual parties are reluctant to participate in data collection. In this paper, we propose two private-preserving data aggregation schemes – *CPDA*, and *SMART* – focusing on additive data aggregation functions. Table I summarizes these two schemes in terms of privacy-preservation efficacy, communication overhead, aggregation accuracy, and computational overhead.

TABLE I

Performance comparison of CPDA and SMART

	CPDA	SMART
Privacy preservation efficacy	Excellent	Excellent ($J \geq 3$)
Communication overhead	Fair	Large
Aggregation accuracy	Good (but sensitive to p_c)	Good (not sensitive to J)
Computational overhead	Fair	Small

We compare the performance of our presented schemes to a typical data aggregation scheme – *TAG*. Simulation results and theoretical analysis show the efficacy of our two schemes.

Our future work includes designing private-preserving data aggregation schemes for general aggregation functions. We are also investigating robust private-preserving data aggregation schemes under malicious attacks.

VII. Acknowledgement

This research was supported by Vodafone Fellowship and NSF grant under TCIP (Trustworthy Cyber Infrastructure for the Power Grid) 492473-727001-191100. Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies. Authors would like to thank Professor Nikita Borisov for the invaluable discussions and comments for this paper.

References

[1] D. Culler, D. Estrin, and M. Srivastava, “Overview of Sensor Networks,” *IEEE Computer*, August 2004.

[2] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, “A Wireless Sensor Network for Structural Monitoring,” *Proceedings of the ACM Conference on Embedded Networked Sensor Systems, Baltimore, MD*, November 2004.

[3] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, “Wireless Sensor Networks for Habitat Monitoring,” *WSNA’02, Atlanta, Georgia*, September 2002.

[4] S. Madden, M. J. Franklin, and J. M. Hellerstein, “TAG: A Tiny AGgregation Service for Ad-Hoc Sensor Networks,” *OSDI*, 2002.

[5] C. Itanagonwiwat, R. Govindan, and D. Estrin, “Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks,” *MobiCom*, 2002.

[6] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, “Impact of Network Density on Data Aggregation in Wireless Sensor Networks,” *In Proceedings of the 22nd International Conference on Distributed Computing Systems*, 2002.

[7] A. Deshpande, S. Nath, P. B. Gibbons, and S. Seshan, “Cache-and-query for wide area sensor databases,” *SIGMOD*, 2003.

[8] I. Solis and K. Obraczka, “The impact of timing in data aggregation for sensor networks,” *ICC*, 2004.

[9] X. Tang and J. Xu, “Extending network lifetime for precision-constrained data aggregation in wireless sensor networks,” *INFOCOM*, 2006.

[10] B. Przydatek, D. Song, and A. Perrig, “SIA: Secure Information Aggregation in Sensor Networks,” *In Proc. of ACM SenSys*, 2003.

[11] Y. Yang, X. Wang, S. Zhu, and G. Cao, “SDAP: A Secure Hop-by-Hop Data Aggregation Protocol for Sensor Networks,” *ACM MobiHoc*, 2006.

[12] D. Wagner, “Resilient Aggregation in Sensor Networks,” *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, 2005.

[13] L. Eschenauer and V. D. Gligor, “A key-management scheme for distributed sensor networks,” in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, November 2002, pp. 41–47.

[14] D. Liu and P. Ning, “Establishing pairwise keys in distributed sensor networks,” in *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS03)*, October 2003, pp. 52–61.

[15] J. Girao, D. Westhoff, and M. Schneider, “CDA: Concealed Data Aggregation for Reverse Multicast Traffic in Wireless Sensor Networks,” in *40th International Conference on Communications, IEEE ICC*, May 2005.

[16] C. Castelluccia, E. Mykletun, and G. Tsudik, “Efficient Aggregation of Encrypted Data in Wireless Sensor Networks,” *Mobiquitous*, 2005.

[17] Q. Huang, H. J. Wang, and N. Borisov, “Privacy-preserving friends troubleshooting network,” in *Symposium on Network and Distributed Systems Security (NDSS)*, San Diego, CA, Feburary 2005.

[18] R. Agrawal and R. Srikant, “Privacy preserving data mining,” in *ACM SIGMOD Conf. Management of Data*, 2000, pp. 439–450.

[19] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke, “Privacy Preserving Mining of Association Rules,” in *Proceedings of The 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 2002.

[20] H. Kargupta, Q. W. S. Datta, and K. Sivakumar, “On The Privacy Preserving Properties of Random Data Perturbation Techniques,” in *the IEEE International Conference on Data Mining*, November 2003.

[21] Z. Huang, W. Du, and B. Chen, “Deriving Private Information from Randomized Data,” in *Proceedings of the ACM SIGMOD Conference*, June 2005.

[22] B. Pinkas, “Cryptographic techniques for privacy preserving data mining,” *SIGKDD Explorations*, vol. 4, no. 2, pp. 12–19, 2002.

[23] W. Du and M. J. Atallah, “Secure multi-party computation problems and their applications: A review and open problems,” in *Proceedings of the 2001 Workshop on New Security Paradigms*. Cloudcroft, NM: ACM Press, September 2001, pp. 13–22.

[24] M. Kantarcioglu and C. Clifton, “Privacy-preserving distributed mining of association rules on horizontally partitioned data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1026–1037, 2004.

[25] A. C. Yao, “Protocols for secure computations,” in *23rd IEEE Symposium on the Foundations of Computer Science (FOCS)*, 1982, pp. 160–164.

[26] I. D. Ronald Cramer and S. Dziembowski, “On the Complexity of Verifiable Secret Sharing and Multiparty Computation,” in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, 2000, pp. 325–334.

[27] J. Halpern and V. Teague, “Rational Secret Sharing and Multiparty Computation,” in *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, 2004, pp. 623–632.