KIPDA: *k*-Indistinguishable Privacy-preserving Data Aggregation in Wireless Sensor Networks

Michael M. Groat*, Wenbo He† and Stephanie Forrest*

*Department of Computer Science, University of New Mexico, Albuquerque, New Mexico 87131, USA
†Department of Electrical Engineering, University of Nebraska-Lincoln, Lincoln, Nebraska 68688, USA
Email: {mgroat, forrest}@cs.unm.edu, wenbohe@engr.unl.edu

***Abstract*—When wireless sensor networks accumulate sensitive or confidential data, privacy becomes an important concern. Sensors are often resource-limited and power-constrained, and data aggregation is commonly used to address these issues. However, providing privacy without disrupting in-network data aggregation is challenging. Although privacy-preserving data aggregation for additive and multiplicative aggregation functions has been studied, nonlinear aggregation functions such as maximum and minimum have not been well addressed. We present KIPDA, a privacy-preserving aggregation method, which we specialize for maximum and minimum aggregation functions. KIPDA obfuscates sensitive measurements by hiding them among a set of camouflage values, enabling *k*-indistinguishability for data aggregation. In principle, KIPDA can be used to hide a wide range of aggregation functions, although this paper considers only maximum and minimum. Because the sensitive data are not encrypted, it is easily and efficiently aggregated with minimal in-network processing delay. We quantify the efficiency of KIPDA in terms of power consumption and time delay, studying trade-offs between the protocol's effectiveness and its resilience against collusion.**

I. Introduction

Wireless sensor networks (WSNs), collecting measurements of our daily activities [1], [2], combine, massage or filter data before they reach a final destination, a process known as data aggregation. This paper addresses the problem of maintaining privacy during data aggregation.

While data passes through a network, the limited computational power inside the network can be exploited to aggregate data along its path. Instead of each data value traveling to the base station, information is combined along the way to conserve bandwidth. Data aggregation is more challenging when privacy and security are a concern, as information can potentially be disclosed either to outside observers or neighboring nodes in the network. Due to limited resources such as memory, power, and computation, mainstream solutions such as public key asymmetric encryption are problematic in this domain.

Privacy-preserving data aggregation (PDA) for additive aggregation based on the algebraic properties of polynomials and addition [3], homomorphic encryption [4], [5], and perturbation techniques [6], [7] have been well addressed in WSNs. However, research efforts on PDA for more general nonlinear aggregation functions, such as maximum (MAX) and minimum (MIN), have been limited. This paper presents KIPDA, a light weight k-indistinguishable solution to PDA for general aggregation functions, which we present in the specialized case of MAX and MIN functions.

Current encryption based schemes, either end-to-end or hop-by-hop, are not well-suited for the general data aggregation problem. End-to-end encryption establishes a secure channel between a sensor node and the base station, preventing in-network aggregation altogether. With hop-by-hop encryption, a node receives encrypted data, decrypts it, aggregates it, re-encrypts the aggregate and then sends the data to the next hop, incurring excess overhead and not guaranteeing privacy at the aggregators.

We propose a non-cryptographic method which obfuscates data by adding a set of camouflage values. In our method, the aggregates, referred to as *real values*, are in plain text so the aggregation computation is efficient. Privacy arises from hiding these real values among camouflage values in a *message set*[1]. We define a message set for MIN/MAX aggregation as the union of the single real value with the camouflage values that a node transmits to its parent in the data aggregation tree, usually in one packet.

This is a form of *k-indistinguishability*, where a single value is indistinguishable from $k-1$ other values. We show that this method consumes considerably less power than end-to-end encryption, and is more power and time efficient than hop-by-hop encryption. Our method is similar to a global symmetric key solution [8], except that each node possesses a random part of the global key. Only when enough nodes collude or are captured by adversaries will privacy be broken.

Several applications could benefit from KIPDA for MAX/MIN aggregation. Intelligent or smart meters for electric utilities are an example where individual usage data are sent to the utility company, which then sends real-time data back to the end user, to encourage energy conservation. Information from the meter is usually sent over an existing cell phone infrastructure, radio transmission, or other unsecured network. These public third-party networks need protection [1]. Information such as the MAX or MIN of certain appliances or users in a neighborhood can be compromised. Another potential application class arises in medicine, where a medical worker

[1]While we refer to a message set as a set, it is technically a vector or a one dimensional array of values.

does not have the time or resources to monitor a large group of patients individually. Determining the MAX or MIN value of an indicator could show that the entire group is within the normal range, or that further investigation is warranted. A similar idea could be used to triage patients at a disaster site [2] or to monitor hazardous substances.

The remainder of the paper is organized as follows: Section II gives the background and model assumptions of our research. Section III provides an overview of the protocol, and Section IV presents it in detail. Section V analyzes the power consumption of KIPDA and compares it to similar techniques. Section VI discusses related work, and Section VII outlines ideas for future work and concludes the paper.

II. Model and Assumptions

A. Data Aggregation in WSNs

We model a sensor network as a connected graph $G(\mathcal{V}, \mathcal{E})$, where sensor nodes are represented as vertices $\mathcal{V}$, and wireless links as edges $\mathcal{E}$. The number of sensor nodes is defined as $N = |\mathcal{V}|$. 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. Aggregation from the individual nodes to the base station is assumed to follow a common tree structured route [9]. We focus in this paper on f as the MAX or MIN function, but the method could be applied to more general aggregation functions as well.

B. Honest But Curious Threat Model

We use the *honest but curious* [10] threat model, where every user or sensor node attempts to break privacy but faithfully follows the protocol specification during data aggregation. This threat model is appropriate because sensors deployed by a common authority can collaborate to fulfill a certain task and be trusted to follow the protocol.

There are three levels of privacy preservation in PDA. The first level ensures that data are not revealed to nodes outside of the network. Hop-by-hop encryption with symmetric keys [11], [12] is able to achieve this goal. Another PDA solution called Order Preserving Encryption Scheme (OPES) [13] allows comparison operations directly on encrypted data. Such a scheme preserves the first level of privacy, yet, if in-network sensor nodes use the same set of mapping functions, they can discover each others' measurements. Hence, the second level of privacy ensures that individual private information is not disclosed to in-network nodes. This is more stringent but likely closer to real world situations. Our method aims to preserve privacy under the honest but curious attack model so that even in-network nodes cannot easily determine other nodes' sensitive data. The third level of privacy ensures that data are not revealed to anybody, including the final recipient or base station [14]; KIPDA does not address this level.

C. Requirements and Trade-offs of PDA

The following criteria summarize the desired characteristics and trade-offs of a PDA scheme. Different applications make different trade-offs among these performance metrics.

1) **Privacy:** To preserve data privacy, actual data, d_i of node i, should not be revealed to an outside observer listening to the radio communication, or another node j in the network. Sometimes privacy requires that d_i not be known to the basestation [14]. PDA schemes should also be robust to collusion among several nodes.
2) **Efficiency:** Data aggregation reduces the number of messages transmitted within a sensor network, thus reducing bandwidth and power usage. However, additional overhead is required to protect privacy. If the energy cost of a PDA scheme if greater than the benefit of data aggregation, then it is not useful. Bandwidth, power consumption, and delay are three important metrics of protocol efficiency.
3) **Accuracy:** The accuracy of the aggregated data affects decisions made from the data. In perturbation-based techniques [6], [7], accuracy is sacrificed to achieve privacy and efficiency.

III. Overview of Solution

As mentioned earlier, KIPDA hides sensor data among a set of camouflage values. First, we introduce the notation used in the paper, which is summarized in Table I.

Let U^i be the set of n values in the message set for node i where $(|U^i| = n, \forall i)$. The message set is composed of the real or actual data, d_i, and the *restricted* and *unrestricted camouflage values*. The message set is an array of values, where the real data and camouflage values are assigned to specific positions in the array according to predefined policies that guarantee 100% accuracy of the aggregation together with indistinguishability. Restricted camouflage values are required to be less than or equal to d_i if MAX aggregation is used, or greater than or equal to d_i if MIN aggregation is used. Unrestricted camouflage values can be either greater than, or less than, or equal to d_i. Let $I = \{1, 2, ..., n\}$ be the index set of $U^i, \forall i$. The *global secret set (GSS)*, a subset of I, denotes the secret index values kept at the base station to determine the final aggregated results. GSS contains the global secret information, which is partially shared among the network nodes. The *node's secret set* (NSS^i) is the secret information about GSS shared with node i. The base station specifies NSS^i for each i to include all elements from GSS and a subset of elements from $\overline{GSS}$, i.e. $\overline{GSS} \cap NSS^i \neq \emptyset$. NSS^i_T denotes the index set of the *true* (real) values in U^i for node i. NSS^i_T is always a subset of GSS and for the MAX/MIN, $|NSS^i_T| = 1$. NSS^i_T is also a subset of NSS^i; hence NSS^i is the union of the index set of the restricted camouflage values and the real values.

For MAX/MIN aggregation, a sensor node i hides its single real value, d_i, in message set, $U^i = \{v^i_1, v^i_2, ..., v^i_n\}$, among restricted and unrestricted camouflage values such that:

$$\begin{aligned}
&v^i_l = d_i, \text{ if } l \in NSS^i_T, \text{ (MAX/MIN)}\\
&v^i_l \leq d_i, \text{ if } l \in NSS^i, \text{ (MAX)}\\
&v^i_l \geq d_i, \text{ if } l \in NSS^i, \text{ (MIN)}\\
&v^i_l \leq d_i \text{ or } v^i_l > d_i, \text{ if } l \in \overline{NSS^i}, \text{(MAX/MIN).} \quad (1)
\end{aligned}$$

This scheme ensures several important properties. *Property 1* guarantees that the base station can correctly determine the final aggregated value. *Property 2* ensures that $NSS^i - NSS^i_T$ draws from both sets GSS, and $\overline{GSS}$, where "$-$" denotes set difference. This guarantees that node i cannot determine the entirety of GSS and NSS^j, $\forall i, j, i \neq j$. *Property 3* guarantees that the true maximum value is not filtered out by the aggregation process.

Property 1: The indices of the real values are drawn from GSS:

$$NSS^i_T \subset GSS, \forall i. \tag{2}$$

Property 2: NSS^i contains elements from both GSS and $\overline{GSS}$. This is required to hide the real or actual value in U^i:

$$\overline{GSS} \cap (NSS^i - NSS^i_T) \neq \emptyset, \forall i. \tag{3}$$

Property 3 NSS^i is a proper superset of GSS:

$$NSS^i_T \subset GSS \subset NSS^i, \forall i. \tag{4}$$

We use an example to illustrate the method. Consider a three-node case of MAX aggregation, shown in Figure 1, where nodes 1, 2, and 3 have sensor readings 23, 34, and 12 respectively. Assume GSS equals $\{1, 3, 5\}$, and $NSS^1_T = \{1\}$, $NSS^2_T = \{5\}$, and $NSS^3_T = \{3\}$, are drawn from GSS. Based on Properties 1, 2, and 3, we have $NSS^1 = \{1, 2, 3, 5, 7\}$ and $\overline{NSS^1} = \{4, 6\}$ for node 1; $NSS^2 = \{1, 3, 4, 5, 7\}$ and $\overline{NSS^2} = \{2, 6\}$ for node 2,

TABLE I
KIPDA NOTATIONS

message set	Vector of camouflage and actual values sent to the next aggregator, indexed by I.
restricted camouflage values	Values in the message set that are greater than the real value for MIN aggregation and less for MAX aggregation.
unrestricted camouflage values	Values in a message set that are either more or less than the real value in the message set
U^i	Notation for the message set of node i. $U^i = \{v^i_1, v^i_2, ..., v^i_n\}$.
U^Ω	Last message set sent from the last node Ω to the base station.
d_i	Actual value of node i. It is hidden in plain sight among U^i where $v^i_l = d_i$, if $l \in I^i_T$.
v^i_l	Values of U^i for node i where $l = 1, 2, ..., n$.
I	Index set of $U^i, \forall i, I = \{1, 2, ..., n\}$
n	Number of values in a message set, $n = \lvert U^i \rvert, \forall i$
GSS	The global secret set kept at the base station that contains possible locations for the final network aggregated value.
NSS^i	The node's secret set for node i. Consists of the union of the index set of the restricted camouflage values and the index of the real value.
$\overline{NSS^i}$	Index set of unrestricted camouflage values values of node i.
NSS^i_T	Index set of the real values of node i.

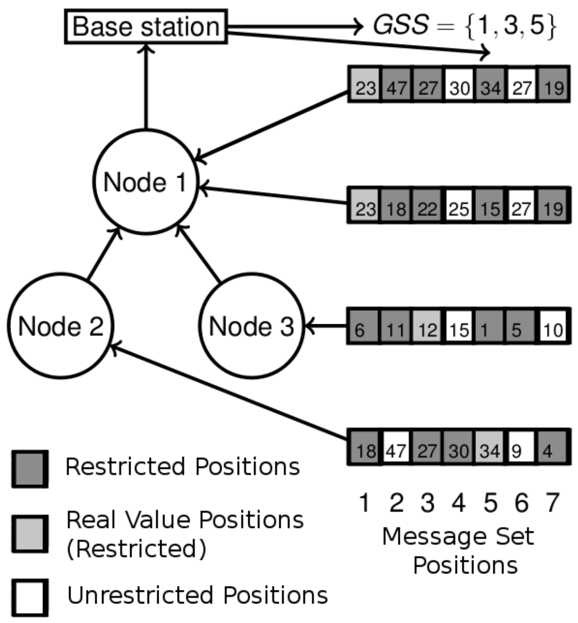


Fig. 1. An example KIPDA aggregation scheme of three nodes.

and $NSS^3 = \{1, 2, 3, 5, 6\}$ and $\overline{NSS^3} = \{4, 7\}$ for node 3. NSS^1, NSS^2, and NSS^3 choose three values from GSS, and two values from $\overline{GSS}$.

After sets NSS^i_T and NSS^i for all i have been determined, the base station sends them to each node in the pre-distribution phase. During the reporting phase, node 2 places its true value, 34, in the 5th slot in U^2. Then it determines the rest of U^2 according to (1). In this way, $U^1 = \{23, 18, 22, 25, 15, 27, 19\}$, $U^2 = \{18, 47, 27, 30, 34, 9, 4\}$, and $U^3 = \{6, 11, 12, 15, 1, 5, 10\}$. During the data aggregation phase, when node 1 receives message sets U^2 and U^3 from its children, it determines the aggregated value where $v^1_l = max\{v^i_l\}$ for each $l = 1, 2, ..., 7$ and $i = 1, 2, 3$. Hence, the aggregated message set, U^1, is $\{23, 47, 27, 30, 34, 27, 19\}$, and replaces the original U^1. U^1 becomes the final message set, U^Ω, which node 1 sends to the base station. The base station determines the final network aggregate among the maximum elements indexed by GSS. In our example, elements at positions 1, 3, 5 of the aggregated set U^Ω are 23, 27 and 34. Hence, 34 is the network MAX aggregate.

As described, this method could be prone to statistical analysis attacks. For example, an adversary could examine the packets for statistical correlations and use this information to guess NSS^i and $\overline{NSS^i}$ for certain i, ultimately guessing GSS and $\overline{GSS}$ of the base station. There are several methods to avoid this problem. In the network, if the theoretical maximum and minimum values are known, then the values in the message set other than d_i could be chosen so the entire message resembles a uniform distribution. If the size of NSS^i prevents this, it can be increased. Alternatively, the sets could be changed or shuffled either after each network wide aggregation, or after a fixed number of aggregations. The base station would choose new sets GSS and $\overline{GSS}$, and end to end encryption could be used to distribute the keys (sets NSS^i and NSS^i_T to each i). These methods would also help if values of neighboring nodes are similar or correlated, or if an adversary manipulated the environment so that some sensor values were known.

A. Accuracy of the Aggregation

Proposition 1: From any one node's viewpoint, KIPDA provides k-indistinguishability where:

$$k = |\overline{NSS^i}| + 1. \tag{5}$$

For MAX aggregation, since $\forall i, j : |\overline{NSS^i}| = |\overline{NSS^j}|$, any node i knows that the real value for any node j is contained in at least the $|\overline{NSS^i}| + 1$ largest values in the positions noted by its own set NSS^i. k is reduced if more rogue nodes collude, which we quantify in Section III-B. To an outside observer without knowledge of GSS, $k = |U^i|$.

Proposition 2: KIPDA accurately computes the MAX and MIN aggregation functions.

The aggregation result can be affected only by unrestricted camouflage values. However, the unrestricted camouflage values occur only in locations indexed by $\overline{NSS^i}$. Since $GSS \subset NSS^i$, and $\overline{NSS^i} \subset \overline{GSS}$, $\forall i$, the unrestricted camouflage values don't affect the aggregated results in positions indexed by GSS at each node. This is because the indexes of the restricted and unrestricted camouflage sets are disjoint.

B. Collusion Attacks

In Figure 1, if node 2 colludes with node 3, they can determine that d_1 is in slots $\{1, 3, 5\}$. They can do this by determining the intersection of NSS^2 and NSS^3, since $GSS \subseteq NSS^i \cap NSS^j$ and $\overline{NSS^i} \cup \overline{NSS^j} \subseteq \overline{GSS}$. Given the size of I, the sizes of GSS and NSS^i must be chosen carefully to achieve good performance against a node collusion attack. There are two ways for colluding nodes to infer either GSS or $\overline{GSS}$. The first, *Proposition 3*, is to infer GSS from $NSS^i_T, \forall i \in$ *colluding nodes*; the other, *Proposition 4*, is to infer $\overline{GSS}$ from $\overline{NSS^i}$ or NSS^i. The optimal size of I is discussed in Section V.

Proposition 3: Assume NSS^i_T is randomly selected from GSS, and nodes collude to infer GSS from NSS^i_T. If there exist x colluding nodes, they can determine:

$$\sum_{i=1}^{x} \frac{|GSS| - i + 1}{|GSS|}, \tag{6}$$

elements in GSS. To get all $|GSS|$ elements in GSS, the expected number of colluding nodes is:

$$x = |GSS| \times \sum_{i=1}^{|GSS|} \frac{1}{i}. \tag{7}$$

This is an instance of the coupon collector's problem [15].

Proposition 4: Let x be the number of nodes colluding in the network. Let $g(x)$ be the number of elements known in $\overline{GSS}$ given x. We assume no bias when $\overline{NSS^i}$ is determined from $\overline{GSS}$. The object of the colluding nodes is to infer $\overline{GSS}$ from the information contained in $\overline{NSS^i}$. When x nodes collude, the following equation computes the number of elements known in set $\overline{GSS}$, where $g(x) \leq |\overline{GSS}|$:

$$g(x) = |\overline{GSS}| - (|\overline{GSS}| - |\overline{NSS^i}|) \left(\frac{|\overline{GSS}| - |\overline{NSS^i}|}{|\overline{GSS}|} \right)^{x-1}. \tag{8}$$

This is determined first by giving a recurrence relation about $g(x)$. When the x-th node colludes with the other $(x-1)$ nodes, $g(x)$ elements in $\overline{GSS}$ will be disclosed as follows:

$$g(x) = g(x-1) + |\overline{NSS^i}| \left(\frac{|\overline{NSS^i}| - g(x-1)}{|\overline{GSS}|} \right). \tag{9}$$

Let $a = |\overline{GSS}|$ and $b = |\overline{NSS^i}|$, (9) implies $g(x) = \frac{a-b}{a} g(x-1) + b$, where $g(1) = b$. Let $y_i = g(i) - a$. Replace the $g(i)$ with $y_i + a$, then $y_i = \frac{a-b}{a} y_{i-1}$ and $y_1 = b - a$, giving $y_i = \left(\frac{a-b}{a}\right)^{i-1} y_1$. Therefore, $g(i) = a - (a-b)\left(\frac{a-b}{a}\right)^{i-1}$.

From Proposition 4, the expected number of collusive nodes required to recover all the elements in $\overline{GSS}$ is obtained by converting (8) into the following equation where $g(x) = |\overline{GSS}|$:

$$|\overline{GSS}| = \left\lceil |\overline{GSS}| - (|\overline{GSS}| - |\overline{NSS^i}|) \left(\frac{|\overline{GSS}| - |\overline{NSS^i}|}{|\overline{GSS}|} \right)^{x-1} \right\rceil. \tag{10}$$

Isolating x in (10) yields the following equation:

$$x = 2 + \left\lfloor \frac{log(\frac{1}{|\overline{GSS}| - |\overline{NSS^i}|})}{log(\frac{|\overline{GSS}| - |\overline{NSS^i}|}{|\overline{GSS}|})} \right\rfloor. \tag{11}$$

According to Proposition 4, the expected number of colluding nodes required to discover GSS decreases as $|GSS|$ increases. According to Proposition 3, the expected number of colluding nodes needed to obtain GSS increases with $|GSS|$. From Propositions 3 and 4, we conclude that the number of collusive nodes needed to infer GSS or $\overline{GSS}$ is the minimum of (7) and (11). Setting (7) equal to (11) solves for the optimal size of GSS.

Figure 2 shows the expected number of colluding nodes that are required to discover a given number of elements in GSS. $|I|$ is set to 15 and $|NSS^i|$ varies between 2 and 4 with different values of k. The figure shows that if we choose $|\overline{NSS^i}| = 3$, i.e. ($k = 4$), the optimal value of $|GSS|$ is 4, and it will take 8 colluding nodes before GSS is entirely known.

IV. Protocol Design

In this section we describe our protocols for the MAX/MIN aggregation functions. We assume the aggregation trees are constructed according to standard data aggregation protocols [9]. There are four phases to KIPDA: pre-distribution, reporting, aggregating, and base station processing.

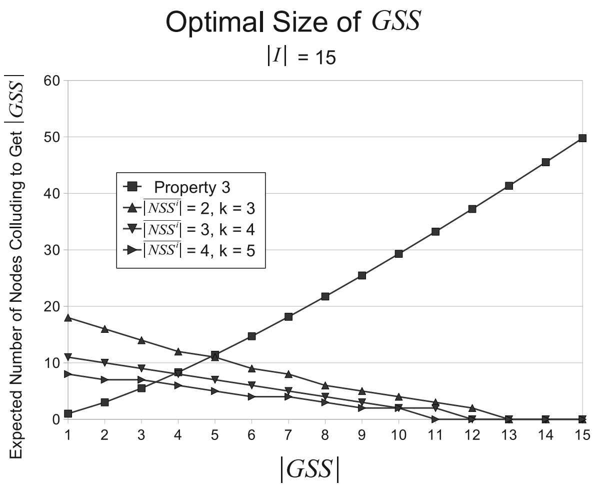


Fig. 2. The optimal size of GSS with $|I| = 15$ is given by the intersection of (7) with (11). (11) varies for different values of k.

A. Pre-distribution

In the pre-distribution phase, the base station chooses a set $GSS \subset I$. This is the global secret information, and each node i keeps a subset of GSS in set NSS^i along with some noise values drawn from $\overline{GSS}$. Hence, no nodes can infer the exact set GSS from their local information, NSS^i. Since $GSS \subset I$, and $|I| = |U^i|$ is proportional to the bandwidth and power consumption, $|I|$ cannot be too large. We discuss the proper size of I in Section V.

After $|I|$ is determined, the optimal values of $|GSS|$ and $|NSS^i|$ are determined by the requirement of k-indistinguishability and Propositions 3 and 4. As an example, Figure 2 shows how to compute $|GSS|$ as the intersection of two curves: (7) and (11). When $|\overline{NSS^i}| = 2$, i.e. $k = 3$, the optimal size of GSS is 5.

Next, the base station determines NSS^i_T and NSS^i for each node i. For MAX/MIN, we have $|NSS^i_T| = 1$, so the base station can determine NSS^i_T by choosing an element from GSS. NSS^i contains all the elements in set GSS and $|NSS^i| - |GSS|$ elements from $\overline{GSS}$, in addition to NSS^i_T. One way to select $|NSS^i| - |GSS|$ elements from set $\overline{GSS}$ is to throw a die with $|I|$ faces without replacement, stopping when there are $|NSS^i| - |GSS|$ distinct numbers in $\overline{GSS}$. There are many methods for distributing keys securely in WSNs, e.g. [16], which could easily be modified to distribute NSS^i and NSS^i_T to the nodes. Once a node has received NSS^i, it can trivially determine $\overline{NSS^i}$.

B. Reporting

In the reporting phase, each node i determines the values for the set U^i where $U^i = \{v^i_1, v^i_2, ..., v^i_n\}$. The message set U^i contains the real values, the restricted camouflage values, and the unrestricted camouflages values. If sensed values have the range $[d_{min}, d_{max}]$, the restricted camouflage values are drawn from $[d_{min}, d_i]$ for MAX aggregation, and from $[d_i, d_{max}]$ for MIN aggregation, for node i. The unrestricted camouflages values are drawn from $[d_{min}, d_{max}]$. Different nodes can use different distributions to generate random values for the restricted or unrestricted camouflages values, so it is harder for others to infer the real value from U^i. Node i places these values in U^i according to (1) and sends message set U^i to its aggregator.

C. Aggregation

In the aggregation phase, each node i computes the MAX (or MIN) for each $l = \{1, 2, ..., n\}$ in v^j_l, among all child nodes j, plus its own v^i_l if it is also a sensing node. The aggregated message set of node i, $U^i = \{v^i_1, v^i_2, \cdots, v^i_n\}$, is determined for the MAX function as:

$$v^i_l = max(v^h_l, v^i_l), \forall h, \tag{12}$$

for each $l = \{1, 2, ..., n\}$, where h ranges over all the child nodes of i. The aggregator i replaces values v^i_l in U^i with the aggregated values, and then passes the aggregated message set, U^i, to its next hop. For the MAX (or MIN) aggregation functions, the values in the message sets grow larger (or smaller) as they approach the base station. One possible solution is to replace one or more values indexed by $\overline{NSS^i}$ in the message sets so that the values appear more uniformly distributed about $[d_{min}, d_{max}]$.

D. Base Station Processing

The final aggregated message set, U^Ω, arrives at the base station and the aggregated result is computed by selecting the MAX (or MIN) from the values indexed by GSS in U^Ω. For example, the result for MAX is:

$$max_{k \in GSS}(v^\Omega_k). \tag{13}$$

E. Generalizing to the Additive Aggregation Function

Other types of aggregation functions can be used in place of MAX/MIN. However, care must be taken when choosing NSS^i. For example, if an additive aggregation is used, we could choose $|NSS^i_T| > 1$, and in the data reporting phase, we would set:

$$\begin{aligned} d_i &= \sum_{l \in NSS^i_T} v^i_l \\ 0 &= \sum_{l \in NSS^i} v^i_l \\ z &= \sum_{l \in \overline{NSS^i}} v^i_l, \end{aligned} \tag{14}$$

where z can be any value. Here, we allow negative values in U^i. The aggregators can sum up each slot their message sets and the base station can retrieve the final aggregation result, which is $\sum_{l \in GSS} v^\Omega_l$, from the final message set, U^Ω.

V. Evaluation

In this section, we compare KIPDA to hop-by-hop and end-to-end encryption methods paying particular attention to power consumption and time delay.

TABLE II
BANDWIDTH AND ENERGY USAGE OF END-TO-END ENCRYPTION PER LEVEL IN A TREE NETWORK WITH A BRANCHING FACTOR OF 3, ASSUMING NO AGGREGATION

Level	Number of Nodes	Bits Sent Per Node	MICAz Energy per Node (μJ)
1	3	17488	10492.8
2	9	5824	3494.4
3	27	1936	1161.6
4	81	640	384.0
5	243	208	124.9
6	729	64	38.4
7	2187	16	9.6

A. *Power Analysis*

1) End-to-End Encryption: End-to-end encryption without aggregation is power-intensive because each sensed value is transmitted to the base station. Let us assume that the data are 16 bits wide. (In some cases with block encryption the data sent over the network would be even larger because of padding.) Table II shows the bandwidth and energy consumption per node for each level in a network with a branching factor of 3. The level is the number of hops a node is away from the base station. Energy usage is determined using calculations from Meulenaer et al. [17] for the energy consumed per bit transmitted of a MICAz architecture.

Nodes closer to the base station consume more bandwidth because more data pass through them. To balance traffic load among nodes, either the sink must be moved around, or the nodes themselves have to migrate [5], both of which are impractical in many cases. Average bandwidth consumption in this scenario is $O(logN)$ per node, assuming no aggregation and N nodes in the network. For KIPDA, bandwidth consumption is $O(|U^i|)$ per node, i.e., the number of values in a message set. Thus, power usage will grow more quickly with network size for end-to-end encryption than for KIPDA. Additionally, because nodes near the sink have to send more information, there will be larger delays due to the time to transmit the extra bits over the radio.

Despite the energy cost, end-to-end encryption provides the best privacy protection. Outsiders, aggregator nodes, and neighboring nodes in the network are all prevented from determining the sensed values or aggregated results.

2) Hop-by-Hop Encryption: We compare KIPDA to hop-by-hop encryption using IDEA, RC5, and RC4, showing its power savings. Although hop-by-hop methods consumes less power near the sink than end-to-end methods, a large amount of power is consumed throughout the network, and more delay is introduced in the decryption and re-encryption phases.

To determine power consumption of various encryption methods in WSNs, we use the results from [18] for IDEA, RC4 and RC5, which were generalized to any generic mote architecture. We apply this generalization to two common architectures to estimate encryption times: The MICAz that has a bus width of 8 bits and runs at 7.37 MHz, and the TelosB that has a bus width of 16 bits and runs at a speed of 4 MHz.

TABLE III
ENERGY CONSUMPTION IN μJ AND nJ OF COMMON OPERATIONS ON THE MICAZ MOTE,7.37 MHZ, AND THE TELOSB MOTE, 4 MHZ [17] .

Operation	MICAz	TelosB
Compute for 1 Clock Tick	3.5 nJ	1.2 nJ
Transmit 1 bit	0.60 μJ	0.72 μJ
Receive 1 bit	0.67 μJ	0.81 μJ

TABLE IV
PARAMETERS a_{BASE} AND b_{BASE} [18]

Algorithm	a_{BASE}	b_{BASE}	blocksize (bits)
RC5 init/encrypt	352114	40061	64
RC5 init/decrypt	352114	39981	64
IDEA encrypt	67751	80617	64
IDEA decrypt	385562	84066	64
RC4	68540	13591	8

To determine energy costs of encryption we used the cost of common operations reported by Meulenaer et al. [17] which are given in Table III, together with the general equation from Ganesan et al. [18] given as:

$$Time_{ENC/DEC} = \frac{a + b * \lceil textlength/blocksize \rceil}{processorfreq * buswidth}, \quad (15)$$

where variables a and b are given as follows:

$$a = a_{BASE} + a_{MUL} + a_{RISC}$$
$$b = b_{BASE} + b_{MUL} + b_{RISC}. \quad (16)$$

Parameters a_{BASE} and b_{BASE} are given in Table IV, and a_{MUL} and b_{MUL} are given in [18], depending on whether a multiplication instruction is native to the architecture. a_{RISC} and b_{RISC} are also given in [18] and depend on whether a RISC or CISC architecture is used. a_{BASE}, b_{BASE}, a_{MUL}, b_{MUL}, a_{RISC}, and b_{RISC}, were determined by minimizing the least square relative error in their experiments. With this information, we can estimate the time, and hence the number of CPU cycles, spent on various encryption methods. The final result, given in Table V, shows time spent per node to encrypt and decrypt 10 bits of data on the MICAz and TelosB architectures with IDEA, RC4, and RC5 encryption.

IDEA and RC5 are both block methods that operate on block sizes of 64 bits, RC4 is a stream method that works in segments of 8 bits. Because the plain text size in all cases was only 10 bits, padding is required. The time to encrypt and decrypt in microseconds was determined from (15). The number of clock ticks were determined by multiplying the time by clock frequency. And finally energy use was determined from the number of clock ticks according to energy per tick given in [17] shown in Table III.

To determine the energy consumption of each node in a hop-by-hop method, we combine the information in Table V with the costs of 1 CPU clock cycle, and the transmission and reception of 1 bit. These costs can be found in [17] and

TABLE V
COST TO ENCRYPT 10 BITS OF DATA ON THE MICAZ AND TELOSB ARCHITECTURES.

Method, Architecture	Time ms	Clock Ticks	Energy μJ
IDEA Enc, MICAz	2902.12	21388.63	74.86
IDEA Enc, TelosB	2673.58	10694.31	12.83
IDEA Dec, MICAz	8350.80	61546.13	215.41
IDEA Dec, TelosB	7693,27	30773.06	36.93
RC5 Enc, MICAz	7037.25	51864.50	181.53
RC5 Enc, TelosB	6483.06	25932.25	31.12
RC5 Dec, MICAz	7035.89	51854.50	181.49
RC5 Dec, TelosB	6481.81	25927.25	31.11
RC4, MICAz	2018.00	14872.63	52.05
RC4, TelosB	1859.08	7436.31	8.92

TABLE VI
ENERGY CONSUMPTION OF HOP-BY-HOP ENCRYPTION PER NODE FOR 10 BITS OF DATA FOR THE MICAZ AND TELOSB ARCHITECTURES

Method, Architecture	Energy μJ
IDEA, MICAz	1404.74
IDEA, TelosB	502.76
RC5, MICAz	1341.80
RC5, TelosB	491.97
RC4, MICAz	375.55
RC4, TelosB	129.87

are given in Table III. The amount of energy consumed is determined by the following equation:

$$E_{HBH} = c * (R(blk) + Dec + Agg) + Enc + T(blk), \quad (17)$$

where c is the branching factor, $R(blk)$ and $T(blk)$ are the energy costs of receiving and transmiting blk bits, blk is the number of bits in the encryption block sizes, and Dec and Enc are the energy consumptions of decrypting and encrypting given in Table V. Agg is the energy required to compute the aggregate. We estimated the time to aggregate 2 values with one clock tick. Table VI gives the energy consumption of hop-by-hop encryption when the average branching factor is 5 and a value consists of 10 bits. 10-bits were chosen to capture 1,024 distinct values, which is enough to express a sensor reading in many WSN applications.

The energy consumption of KIPDA is determined with the following equation:

$$E_{KIPDA} = c * m(R(bpv) + Agg) + m * T(bpv), \quad (18)$$

where m is the number of values in a message set, and bpv is the number of bits per value.

B. Size of Set I

Figure 3 shows that KIPDA can accommodate 34 to 35 values before it consumes the same energy as IDEA and RC5 for the MICAz architecture. Note that these two methods operate on a block size of 64 bits, which is why RC4, a streaming encryption method, appears so low. It is operating on segments of 8 bits. KIPDA can use up to about 7 camouflage values before it uses the same amount of power as RC4. Figure 3 also gives results for the TelosB architecture where RC4 works so efficiently that the crossover point between the two methods is about 2 values, yet for RC5 and IDEA the crossover point is about 9.

To achieve a net power savings from IDEA or RC5 encryption, the size of I for the MicaZ motes needs to be less than 33, or less than 9 for RC4. For the TelosB architecture, the size of I needs to be less than 10 in comparison to IDEA and RC5 encryption. It would not be an advantage to use KIPDA on the TelosB architecture if RC4 encryption is used. However, as described in the next section, KIPDA significantly reduces delay in the network, and would be appealing in networks that require a minimal network delay.

The sizes of the rest of the sets can be determined from $|I|$. If we choose $|I|$ based on power analysis to be 25, and we want our k-indistinguishability factor to be 8, then based on (5), $|NSS^i|$ should be 7, and the optimal size of GSS, based on the method discussed in Section III-B, should be 4.

C. Delay Analysis

Timing for the hop-by-hop method is determined with the following equation:

$$Time_{HBH} = c * Dec_t + Enc_t + (c+1) * \lceil textsize/blk \rceil * Blk/BW, \quad (19)$$

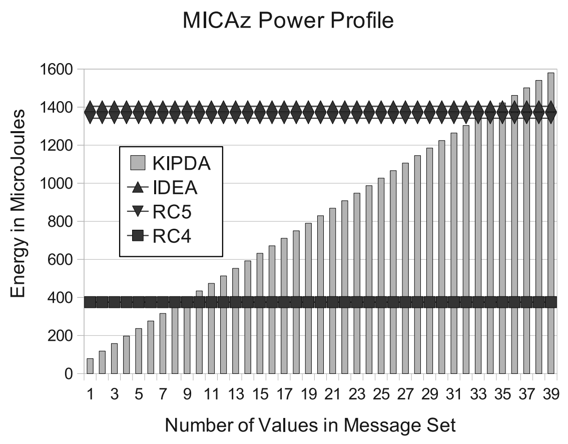
where blk is the message block size, and BW is the bandwidth for both architectures (.25 bits per microsecond). Since the bandwidth on both architectures is the same, the calculated times are the same. The equation to determine the times for KIPDA is given as follows:

$$Time_{KIPDA} = (c+1) * m * bpv/BW. \quad (20)$$

We used 10 bits per value and an average branching factor of 5 in our analysis. We compared the time it takes for IDEA, RC4, and RC5 encryption on both architectures to KIPDA, omitting the figure because of space constrains. However, the analysis shows that KIPDA excels at timing and can send about 47 decoy values before it reaches the same time used by RC4 to encrypt and send one value on either architecture. For IDEA and RC5, KIPDA can send about 160 decoy messages before it uses the same amount of time that these methods can encrypt and send one value. This could be important on delay intolerant networks. We conclude that it would be acceptable for our scheme to consume more power in transmitting camouflage values if delay were important.

VI. RELATED WORK

Data aggregation without privacy achieves bandwidth and energy efficiency in resource-limited WSNs [9]. Previous work [19]–[22] addresses data aggregation in various application scenarios with the assumption that all sensors are working in trusted and friendly environments. LeMay et al. summarize the functional characteristic of wireless metering sensors and categorize attacks in [23], where both privacy and security are concerns in the given scenarios. Refs. [24]–[26] investigate



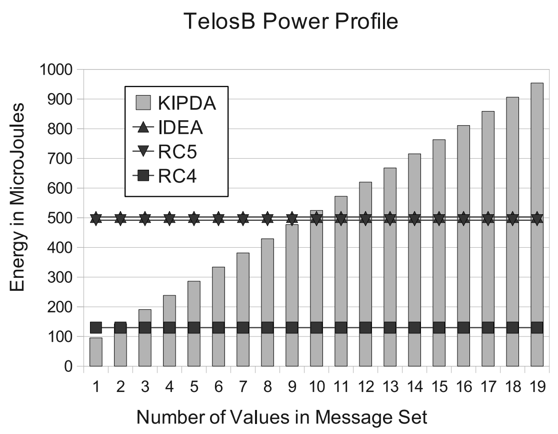


Fig. 3. MicaZ and TelosB power profiles per node for 10 bits of sensed data in a network with a branching factor of 5. KIPDA can send about 30 decoy values on the MicaZ architecture and 8 on the TelosB architecture before it uses more energy than IDEA or RC5.

secure data aggregation against adversaries who try to tamper with the intermediate aggregation result. PDA is also closely related to and has been studied in the data mining domain [6], [7], [27] and peer-to-peer network applications [28].

The majority of research in secure data aggregation takes either a hop-by-hop or end-to-end approach. In a hop-by-hop approach, data are decrypted before the aggregation step, aggregated, then encrypted and forwarded to its next destination. Because data are decrypted, it cannot provide data confidentiality at the aggregator nodes. Additionally there is a latency delay due to the decryption/encryption process. To combat these problems, a set of algorithms have been developed to operate on data without decryption. Homomorphic encryption schemes [4], [5] support efficient aggregation of encrypted data without decryption for additive aggregation functions. Because the data are encrypted from end to end, data confidentiality in the network is not a problem. However, these methods do not work well with nonlinear functions such as MAX and MIN. Although end-to-end encryption uses less computation, distributing encryption keys to the nodes is an issue. Hop-by-hop encryption incurs more computation overhead, and the plain text is available at each node, which increases the risk of data leakage through node capture attacks.

Previous efforts on PDA focused on the additive aggregation functions. Horey et al. propose a data collection scheme based on negative surveys [14], where sensor nodes transmit a sample of the data complement to a base station instead of transmitting their actual data. The base station then uses the negative samples to reconstruct a histogram of the original sensor readings. In [29], Feng et al. propose a family of secret perturbation-based schemes that can protect sensor data confidentiality without disrupting the additive data aggregation result. He et al. proposed two PDA protocols in [3] based on algebraic properties of polynomials and addition operation. These efforts in privacy preservation domain do not assume data manipulation/pollution attacks. In [30], Ganti et al. present architectural components for privacy guarantees on stream data from private owned sensors to collect mutually interested aggregated phenomena.

Although the concept of camouflage has not, to the best of our knowledge, been applied to data aggregation, it has been applied to routing methods [31], [32]. In [33], the authors use a decoy sink to perturb traffic and hence protect the location of the real sink.

k-Indistinguishability is closely related to k-anonymity. k-Anonymity is designed to prohibit linking attacks, where an adversary matches auxiliary information with public or broadcasted information to determine the identity of one or more individuals. In contrast, KIPDA ensures the indistinguishability of the data itself instead of the identity of individuals or the source of the data.

VII. Future Work and Conclusion

Future work will focus on using variable sizes for sets I, GSS, NSS^i. As the cost for radio communication declines, more emphasis will be placed on methods that take advantage of such a decline. New keyless methods can be investigated for the pre-distribution phase, possibly by querying the base station for values that are in GSS after deployment. For example, the base-station could randomly deny queries that ask if a certain value is present in set GSS. Additional future work will focus on addressing different adversarial models and implementing KIPDA in actual sensors.

While encryption provides a stronger level of privacy, we have shown in Section V that it is costly. Future network implementors will have to determine the appropriate tradeoffs between privacy and energy and time constraints. In this paper we showed that WSNs can protect confidentiality by hiding values in plain text along with decoys. By allowing the maximum or minimum to be in plain-text, aggregation can take place efficiently, which would otherwise be difficult. Confidentiality is achieved through k-indistinguishability with the aggregates hidden among $k-1$ other values. Dividing a message set into different subsets, NSS^i, NSS^i_T, and $\overline{NSS^i}$, allow us to camouflage the message sets with restricted decoys and unrestricted decoys. We use a semi shared global key, GSS, which allows some resistance to node collusion and capture. We have shown that it is more power efficient to slightly increase the message bandwidth with decoys than to use conventional methods of hop-by-hop encryption.

Acknowledgments

The authors thank Gelareh Taban for her suggestions, ideas, and conversations. M.M.G. acknowledges support from Motorola and NSF grant HRD-0622930. S.F. acknowledges the partial support of NSF (grants CCF-0621900, CCR-0331580, SHF-0905236) and AFOSR MURI grant FA9550-07-1-0532. W.H. acknowledges support from DOE NNSA grant DE-FG52-06NA27494.

References

[1] Google, "Google powermeter," 2009, [Online]. Available:http://www.google.org/powermeter/.

[2] H. S. N. Lab, "Codeblue: Wireless sensors for medical care," 2008, [Online]. Available:http://fiji.eecs.harvard.edu/CodeBlue.

[3] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "PDA: Privacy-preserving data aggregation in wireless sensor networks," in *26th IEEE Intl. Conf. on Comp. Comm. (INFOCOM)*, 2007.

[4] J. Girao, D. Westhoff, and M. Schneider, "CDA: Concealed data aggregation for reverse multicast traffic in wireless sensor networks," in *40th IEEE Intl. Conf. on Communications, (ICC)*, May 2005.

[5] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks," *2nd Intl. ICST Conf. on Mobile and Ubiquitous Sys.: Comp., Net. and Services (Mobiquitous)*, 2005.

[6] H. Kargupta, Q. W. S. Datta, and K. Sivakumar, "On the privacy preserving properties of random data perturbation techniques," in *IEEE Intl. Conf. on Data Mining*, November 2003.

[7] Z. Huang, W. Du, and B. Chen, "Deriving private information from randomized data," in *Proceedings of the ACM SIGMOD Conf. on Management of Data*, June 2005.

[8] B. Lai, S. Kim, and I. Verbauwhede, "Scalable session key construction protocol for wireless sensor networks," in *IEEE Workshop on Large Scale RealTime and Embedded Systems (LARTES)*, December 2002, p. 7.

[9] S. Madden, M. J. Franklin, J. Hellerstein, and W. Hong, "TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks," in *5th Symp. on Operating Systems Design and Implementation (OSDI)*, 2002.

[10] O. Goldreich, *Foundations of Cryptography: Volume 2, Basic Applications*. New York, NY, USA: Cambridge University Press, 2004.

[11] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *IEEE Symp. on Research in Sec. and Priv.*, 2003, pp. 197–213.

[12] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key predistribution scheme for wireless sensor networks," in *Proc. of the 10th ACM Conf. on Comp. and Comm. Security (CCS)*, October 2003, pp. 42–51.

[13] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proc. of the 2004 ACM SIGMOD Intl. Conf. on Management of Data*, 2004, pp. 563–574.

[14] J. Horey, M. M. Groat, S. Forrest, and F. Esponda, "Anonymous data collection in sensor networks," in *4th Annual Intl. Conf. on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous)*, Aug. 2007.

[15] V. G. Papanicolaou, G. E. Kokolakis, and S. Boneh, "Asymptotics for the random coupon collector problem," *J. of Computational and Applied Mathematics*, vol. 93, no. 2, pp. 95–105, 1998.

[16] A. Perrig, M. Luk, and C. Kuo, "Message-in-a-bottle: User-friendly and secure key deployment for sensor nodes," in *Proc. of the ACM Conf. on Embedded Networked Sensor System*, October 2007.

[17] G. de Meulenaer, F. Gosset, F.-X. Standaert, and O. Pereira, "On the energy cost of communication and cryptography in wireless sensor networks," in *Proc. of the 2008 IEEE Intl. Conf. on Wireless & Mobile Computing, Networking & Comm.* IEEE Computer Society, 2008, pp. 580–585.

[18] P. Ganesan, R. Venugopalan, P. Peddabachagari, A. Dean, F. Mueller, and M. Sichitiu, "Analyzing and modeling encryption overhead for sensor network nodes," in *Proc. of the 2nd ACM Intl. Conf. on Wireless Sensor Networks and App.*, 2003, pp. 151–159.

[19] T. Abdelzaher, T. He, and J. Stankovic, "Feedback control of data aggregation in sensor networks," in *43rd IEEE Conf. on Decision and Control (CDC)*, vol. 2, December 2004, pp. 1490–1495.

[20] J.-Y. Chen, G. Pandurangan, and D. Xu, "Robust computation of aggregates in wireless sensor networks: distributed randomized algorithms and analysis," in *Proceedings of the 4th international symposium on Information processing in sensor networks (IPSN)*. Piscataway, NJ, USA: IEEE Press, 2005.

[21] X. Tang and J. Xu, "Extending network lifetime for precision-constrained data aggregation in wireless sensor networks," in *25th IEEE Intl. Conf. on Comp. Comm. (INFOCOM)*, 2006.

[22] M. Li and Y. Liu, "Underground structure monitoring with wireless sensor networks," in *6th Intl. Sym. on Information Processing in Sensor Networks (IPSN)*, Cambridge, Massachusetts, USA, April 2007.

[23] M. LeMay, G. Gross, C. A. Gunter, and S. Garg, "Unified architecture for large-scale attested metering," in *Proc. of the 40th Annual Hawaii Intl. Conf. on Sys. Sciences (HICSS)*. IEEE Computer Society, January 2007, p. 115b.

[24] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure information aggregation in sensor networks," in *Proc. of the 1st Intl. Conf. on Embedded Networked Sensor Systems (SenSys)*. New York, NY, USA: ACM, 2003, pp. 255–265.

[25] Y. Yang, X. Wang, S. Zhu, and G. Cao, "SDAP: A secure hop-by-hop data aggregation protocol for sensor networks," *ACM Trans. Inf. Syst. Secur.*, vol. 11, pp. 18:1–18:43, July 2008.

[26] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," in *Proc. of 13th ACM Conference on Computer and Communications Security (CCS)*, October 2006.

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

[28] 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.

[29] T. Feng, C. Wang, W. Zhang, and L. Ruan, "Confidentiality protection schemes for data aggregation in sensor networks," in *27th IEEE Intl. Conf. on Comp. Comm. (INFOCOM)*, Phoenix, AZ, April 2008.

[30] R. Ganti, N. Pham, Y.-E. Tsai, and T. Abdelzaher, "Poolview: Stream privacy for grassroots participatory sensing," in *6th ACM Conf. on Embedded Networked Sensor Systems (Sensys)*, November 2008.

[31] D. Goldschlag, M. Reed, and P. Syverson, "Onion routing for anonymous and private internet connections," *Communications of the ACM*, vol. 42, no. 2, February 1999.

[32] R. Dingledine, N. Mathewson, and P. Syverson, "Deploying low-latency anonymity: Design challenges and social factors," in *Proc. of the IEEE Symp. on Security & Privacy*, September 2007.

[33] W. Conner, T. F. Abdelzaher, and K. Nahrstedt, "Using data aggregation to prevent traffic analysis in wireless sensor networks," in *Intl. Conf. on Dist. Computing in Sensor Systems (DCOSS)*, 2006, pp. 202–217.