Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 256852, 9 pages
http://dx.doi.org/10.1155/2013/256852

Research Article

An Efficient Data Aggregation Protocol Concentrated on Data Integrity in Wireless Sensor Networks

Liehuang Zhu, Zhen Yang, Meng Li, and Dan Liu

Beijing Engineering Research Center of Massive Language Information Processing and Cloud Computing Application, School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China

Correspondence should be addressed to Liehuang Zhu; liehuangz@bit.edu.cn

Received 5 March 2013; Accepted 30 April 2013

Academic Editor: Yulong Shen

Wireless sensor networks consist of a great number of sensor nodes with strictly limited computation capability, storage, communication resources, and battery power. Because they are deployed in remote and hostile environments and hence are vulnerable to physical attacks, sensor networks face many practical challenges. Data confidentiality, data integrity, source authentication, and availability are all major security concerns. In this paper, we focus on the very problem of preserving data integrity and propose an Efficient Integrity-Preserving Data Aggregation Protocol (EIPDAP) to guarantee the integrity of aggregation result through aggregation in sensor networks. In EIPDAP, base station can immediately verify the integrity of aggregation result after receiving the aggregation result and corresponding authentication information. However, to check integrity, most existing protocols need an additional phase which will consume a lot of energy and cause network delay. Compared with other related schemes, EIPDAP reduces the communication overhead per node to $O(\Delta)$, where Δ is the degree of the aggregation tree for the network. To the best of our knowledge, EIPDAP has the most optimal upper bound on solving the integrity-preserving data aggregation problem.

1. Introduction

Wireless sensor networks (WSNs) have many security-critical applications such as real-time traffic monitoring, wildfire tracking, or military surveillance. In a sensor network, thousands of low-cost sensor nodes collectively monitor an area within a certain range and report their own data to the base station which distributes a data query. However, this would incur high communication overhead which cannot be afforded by sensor nodes. Data aggregation [1, 2] mechanisms are proposed to reduce the power consumption. Data aggregation poses security threat; many secure data aggregation protocols [3, 4] have been emerging over these years, which prove to be secure and considerably improve the resource utilization.

Although data confidentiality could guarantee that legal parties obtain plain data without being leaked out to adversaries, it does not protect data from being altered [5–7]. In this paper, we focus on the problem of preserving data integrity through aggregation in sensor networks. Message authentication codes (MACs) are used in [8] to protect data integrity, while causing other problems, such as high communication overhead. In this paper, we present a provably secure sensor network integrity-preserving aggregation protocol based on the elliptic curve discrete logarithm problem for general networks with hierarchical aggregator topologies, assuming that adversaries are able to corrupt a (small) fraction of sensors. With the increasing of sensor node's computation capacity, public key cryptography, such as elliptic curve cryptosystems (ECC), is suitable for constrained environments such as WSN. In [9], authors propose secure data aggregation schemes using ECC to obtain data confidentiality and integrity in the data aggregation because of their smaller key size, faster computations, and reductions in processing power, storage space, and bandwidth. TinyECC is proposed by Liu and Ning [10] which provides ECC-based operations that can be flexibly configured and integrated into WSN applications.

An adversary can perform a variety of attacks. For example, a denial-of-service (DoS) attack can totally block the communication between sensor nodes and the base station. However, this attack is not concerned because it is detectable by the querier and solutions can be implemented to remedy this situation. In stealthy attack [4], the attacker's goal is to make the base station accept false aggregation results, which are significantly different from the true results determined by the measure values, while not being detected by the base station. Our goal is to prevent this kind of attack even when high-level aggregator is corrupted.

A number of protocols [11–13] have been proposed which focus on the problem that how can the base station obtain a good approximation of the aggregation result and how to obtain data integrity when a fraction of sensor nodes are compromised. One common sensor feature is the disproportionately high cost of transmitting information, as compared to performing local computation. For example, a Berkeley mote spends approximately the same amount of energy to compute 800 instructions as it does in sending a single bit of data. It thus becomes essential to reduce the number of bits forwarded by intermediate nodes, in order to extend the entire network's lifetime [14]. All the above schemes need to verify the integrity of aggregation result in an additional phase which consumes a lot of energy and causes network delay.

In this paper we propose EIPDAP, which can immediately verify the integrity of aggregation result after receiving the aggregation result and corresponding authentication information, hence significantly reducing energy consumption and communication delay which will be caused if the verification process is done through another query-and-forward phase.

The rest of the paper is organized as follows: in Section 2 we describe a survey of other approaches to integrity-preserving aggregation in sensor networks, in Section 3 more details about the problem we are trying to solve are discussed, in Section 4 we describe a new scheme that is, the centerpiece of our work, and in Section 5 the security properties and performance of our scheme are analyzed.

2. Related Work

There has been a number of works on preserving integrity in aggregation protocols for sensor networks. Many protocols have been proposed for the single-aggregator model [4, 13, 15]. But the aggregator in these schemes suffers from significantly high congestion and only reduces communications on the link between the aggregator and the base station. So this model is not scalable to large multihop sensor deployments.

Another significant work is introduced in [11]. The main idea of this approach is that each node sends its value, complement, and commitment up the aggregation tree and then a commitment would pass down the tree for a node to verify that if its value was added into the SUM aggregation and the complement of its data value was added into the COMPLEMENT aggregation. However, the scheme requires three phases. The delay aggregation strategy used in the second phase increases communication from $O(1)$ to $O(\log n)$, computation from $O(1)$ to $O(q \log n)$, where n is the number of nodes in the network and q is the number of forests in the commitment tree. The result-checking phase costs $O(\Delta \log^2 n)$ congestion. Frikken and Dougherty in [12] improves Chan's approach by reducing the maximum communication to $O(\Delta \log n)$.

A secure hop-by-hop data aggregation protocol SDAP for sensor networks is proposed in [13]. The authors believe that we should be more concerned about high-level nodes, since these nodes represent a large portion of the final result delivered to base station and there would be more catastrophic consequences if they are compromised. Hence, SDAP dynamically partitions the topology tree into multiple logical groups of similar sizes using a probabilistic approach, following the divide-and-conquer principle. In this way, fewer nodes are located under a high-level sensor node in a logical subtree resulting in reduced potential security threat by a compromised high-level node. SDAP introduces probability and attestation to the data result-checking; the communication required per node is $O(\log (n/n_g))$. Because SDAP just let part nodes be attested, attestation algorithm cannot find all compromised nodes. By adding attestation paths can increase the detection probability, but it will increase communication cost.

Aggregate message authentication codes introduced by Katz and Lindell (CT-RSA 2008) [8] provided a new perspective of preserving integrity. In their construction, aggregating MAC simply computes the XOR of all the MACs into one value, the size of which is the same as an ordinary MAC. After receiving all the data and the final aggregate MAC, the base station uses secret keys shared with each node to compute a new aggregate MAC from these data and compares it with the received aggregate MAC. Although it remarkably reduces communication overhead we have seen in former protocols [11–13] and is easy to perform, it suffers from the "mix-and-match" attacks [16] in which the adversary can easily forge several types of aggregate combinations.

In [9], the authors proposed a new algorithm using homomorphic encryption and additive digital signatures to achieve confidentiality, integrity, and availability for in-network aggregation in WSN. However, the protocol cannot resist stealthy attack. We discuss concrete attack on the protocol due to Albath and Madria [9] in the appendix.

Besides, there have been several protocols designed for preserving the confidentiality of the aggregation results [17–19]. This issue is orthogonal to our work and is not considered in this paper.

3. Problem Model

This section contains the definitions of basic problems and includes discussion on the nodes' setup, the security infrastructure, and the attack model.

3.1. Network Assumptions. We assume a query-based sensor network with a large number of sensors and a powerful base station with transmission ranges covering the whole wireless sensor network can broadcast messages to all nodes directly.

Before aggregation process, sensors will form a tree topology where base station locates at the root.

We further assume that the base station would broadcast an authenticated query before collecting data. If there is no aggregation tree, then an aggregation tree should be formed as the query has been sent to all nodes. Our protocol takes the structure of the aggregation tree as given. One method for constructing an aggregation tree is described in TaG [20].

Each node is sensing an integer value r that is in the range $(0, v]$ (we rule out "0" in defense of θ_i which we will explain later) for some application-based value v. The goal is to return the SUM result with two tags proving that the SUM result has not been forged (even in the presence of malicious nodes). Due to resource constrains, all readings need to be aggregated by aggregators while being transmitted over a multihop path.

3.2. Security Infrastructure. We assume that each node i has a unique identifier s_i, private keys $r_i, l_i \in Z_p$ and shares a private key sk_i and a private point $\theta_i \in$ cyclic elliptic group $E(Z_p)$ with base station. ECC domain parameters including the generator point $G \in E(Z_p)$ are preloaded in all nodes. In each node i we set two parameters α_i and β_i which will be used later:

$$\begin{aligned} \alpha_i &= r_i G, \\ \beta_i &= r_i \alpha_i. \end{aligned} \tag{1}$$

3.3. Attack Model and Security Goals. We consider a setting with a polynomially bounded adversary, which can physically access the sensors and read their interval values. The adversary is also restricted to corrupt a (small) fraction of nodes including the aggregators.

Once the adversary compromises a sensor node, it can obtain all the node's secret keys. An adversary can modify, forge, or discard messages or simply transmit false aggregate results, and its goal is to forge valid aggregate result to be accepted by the base station. The higher false aggregate result level is, more catastrophic consequence will be caused.

In this setting, we focus on stealthy attacks [4] where the attacker's goal is to make the base station accept false aggregate results while not being detected by base station. And our security goal is to prevent stealthy attacks. In particular, we want to guarantee that once the aggregate result has been accepted by the base station, it is indeed the real result aggregated by honest nodes.

Definition 1 (integrity-preserving aggregation algorithm). An aggregation algorithm is integrity-preserving if, by tampering with the aggregation process, an adversary is unable to induce the base station to accept any forged aggregate result.

Since if a sensor node is compromised, the adversary can obtain all its confidential information (e.g., cryptographic keys) and send false data without being detected. In this paper we will focus on the situation where an aggregator is compromised and see whether it can forge a valid aggregate result.

In this paper, however, we do not address the denial-of-service attack where the adversary prevents the querier from getting any aggregation result at all; because nodes' not responding queries clearly indicate that something is wrong and solutions can be implemented to remedy this situation.

4. Our Work

In this section, we present a new approach, especially aiming to preserve integrity of the aggregation result. We first give an overview of this approach and then present the details.

4.1. Overview. The design of our algorithm is based on the elliptic curve discrete logarithm problem. The overall algorithm consists of three main phases: query dissemination, aggregation-commit, and result-checking.

In query dissemination phase, the base station broadcasts the query to the network. An aggregation tree, or a directed spanning tree over the network topology with the base station at the root, is formed as the query sent to all the nodes, if one is not already present in the network. Then the path-keys and edge-key for each node encrypted with the secret key shared between base station and node are sent to the corresponding node. Path-key and edge-key are calculated by the base station according to the network topology. We show the detail of the calculation of the path-key in Section 4.2.

In aggregation-commit phase, each sensor node collects raw data and computes two corresponding tags before sending them to their own parent node in the aggregation tree. After receiving all the messages from all child nodes, aggregator performs modulo addition operations over the three items and forwards the result to high-level aggregators until the base station.

In the result-checking phase, the base station verifies the integrity of the SUM aggregation with two aggregation tags. Compared with Chan's and Keith's approach, ours does not require any dissemination from top root node down to the leaf nodes which causes congestion $O(\Delta \log^2 n)$ in Chan's approach and $O(\Delta \log n)$ in Keith's approach and energy cost in this phase.

4.2. The SUM Approach

4.2.1. Query Dissemination Phase. Before aggregation, the base station broadcasts an authenticated query to the network. The query request message contains a nonce N to prevent replay attack [1]. If there is no aggregation tree, an aggregation tree with the base station at the root will be formed as the query has been sent to all nodes. Then the tree information will be reported back to the base station. After the base station receives the tree information, it calculates the path-key for each node: for each aggregator or sensor node i, base station generates a key bs, and calculates edge key according to one-way hash function F, where

$$k_{i-j} = F_{\text{bs}}\left(s_i, s_j, N\right), \tag{2}$$

and node i is the parent of node j. s_i and s_j are unique identifiers of node i and node j.

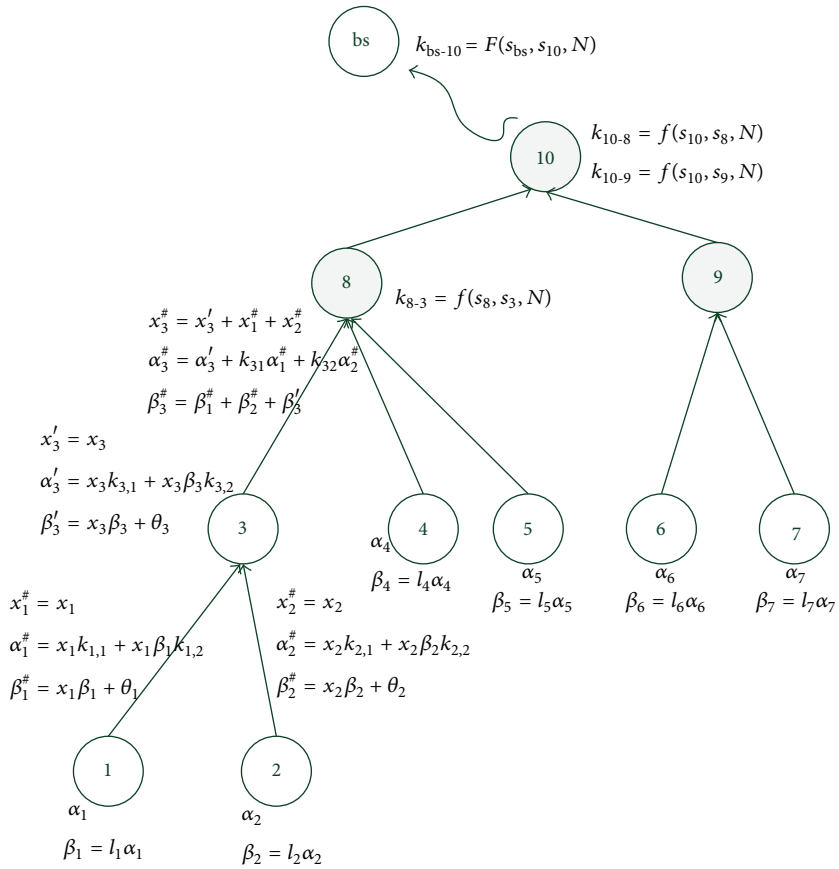

Figure 1: Aggregation phase. The nodes 1, 2, 3, 4, 5, 6, and 7 are sensor nodes, and the nodes 8, 9, and 10 are aggregators while node 3 works as both sensor node and aggregator. Without losing generality, we assume that every intermediate node is able to sense raw data and performs aggregation like node 3 does.

For each sensor node i, base station also calculates two path-keys $k_{i,1}$ and $k_{i,2}$ as follows:

$$k_{i,1} = \frac{\theta}{k_{\text{path}}}, \tag{3}$$

$$k_{i,2} = \frac{l}{k_{\text{path}}}, \tag{4}$$

where θ is a point in $E(Z_p)$, l is an integer and they are both chosen by base station to enable data aggregation and prevent stealthy/replay attacks.

If the path from base station to sensor node i is 1-2-3-i, then $k_{\text{path}} = k_{1\text{-}2}k_{2\text{-}3}k_{3\text{-}i}$.

Finally the base station with transmission ranges covering the whole wireless sensor network directly broadcasts to node i the path-keys and edge-keys encrypted with the secret key sk_i.

4.2.2. Data Aggregation Phase. In the query dissemination phase, each node has already identified their parents and the base station has the overall view of the aggregation tree.

In Figure 1, take paths BS-10-8-3-1 and BS-10-8-3-2, for instance, as sensor node, nodes 3, 1, and 2 each has a message, that is, to be passed to their parents. And the message has the following format:

$$\left\langle x_i^{\#}, \alpha_i^{\#}, \beta_i^{\#} \right\rangle, \tag{5}$$

where $x_i^{\#}$ is the SUM aggregation over all sensor nodes in the subtree; $\alpha_i^{\#}$ and $\beta_i^{\#}$ are the first and second tag, respectively.

For nodes 1 and 2:

$$\begin{gathered} x_1^{\#} = x_1, \qquad x_2^{\#} = x_2, \\ \alpha_1^{\#} = x_1k_{1,1} + x_1\beta_1k_{1,2}, \\ \alpha_2^{\#} = x_2k_{2,1} + x_2\beta_2k_{2,2}, \\ \beta_1^{\#} = x_1\beta_1 + \theta_1, \qquad \beta_3^{\#} = x_3\beta_3 + \theta_3. \end{gathered} \tag{6}$$

For aggregator/sensor node 3 with data x_3, it first computes α_3' and β_3' as a sensor node:

$$\begin{gathered} x_3' = x_3, \qquad \alpha_3' = x_3k_{3,1} + x_3\beta_3k_{3,2}, \\ \beta_3' = x_3\beta_3 + \theta_3. \end{gathered} \tag{7}$$

Then node 1 and 2 send their data and tags to node 3. After receiving all messages from its subtree, node 3 works as aggregator to perform aggregation:

$$\begin{aligned} x_3^{\#} &= x_1 + x_2 + x_3', \\ \alpha_3^{\#} &= k_{3\text{-}1}\alpha_1^{\#} + k_{3\text{-}2}\alpha_2^{\#} + \alpha_3', \\ \beta_3^{\#} &= \beta_1' + \beta_2' + \beta_3', \end{aligned} \tag{8}$$

and sends $\langle x_3^{\#}, \alpha_3^{\#}, \beta_3^{\#}\rangle$ to node 8.

Aggregators 8 and 10 perform corresponding tasks:

$$\begin{aligned} x_8^{\#} &= x_3 + x_4 + x_5, \qquad x_{10}^{\#} = x_8^{\#} + x_9^{\#}, \\ \alpha_8^{\#} &= k_{8\text{-}3}\alpha_3^{\#} + k_{8\text{-}4}\alpha_4^{\#} + k_{8\text{-}5}\alpha_5^{\#}, \\ \alpha_{10}^{\#} &= k_{10\text{-}8}\alpha_8^{\#} + k_{10\text{-}9}\alpha_9^{\#}, \\ \beta_8^{\#} &= \beta_3^{\#} + \beta_4^{\#} + \beta_5, \qquad \beta_{10}^{\#} = \beta_8^{\#} + \beta_9^{\#}, \end{aligned} \tag{9}$$

where node 8 sends $\langle x_8^{\#}, \alpha_8^{\#}, \beta_8^{\#}\rangle$ to node 10, and node 10 sends $\langle x_{10}^{\#}, \alpha_{10}^{\#}, \beta_{10}^{\#}\rangle$ to base station.

4.2.3. Result-Checking Phase. The purpose of result-checking phase is to enable base station to verify that the integrity of SUM $x_{10}^{\#}$ has not been violated. The verification is performed as follows.

Base station checks if

$$\beta_{10}^{\#} - l^{-1}k_{\text{bs-}10}\alpha_8^{\#} = \sum\theta_i \; - \; l^{-1}\theta x_{10}^{\#}, \tag{10}$$

where $\langle x_{10}^{\#}, \alpha_{10}^{\#}, \beta_{10}^{\#}\rangle$ is sent by node 10 to base station; θ, l, $k_{\text{bs-}10}, \theta_i$ (i: from 1 to 7) are only known to base station and l^{-1} is the inverse of l modulo which is the order q of the elliptic curve group $E(Z_p)$.

Since

$$\begin{aligned} \beta_{10}^{\#} &= \beta_8^{\#} + \beta_9^{\#} \\ &= \beta_3' + \beta_4^{\#} + \beta_5^{\#} + \beta_6^{\#} + \beta_7^{\#} \\ &= \beta_1^{\#} + \beta_2^{\#} + \beta_3^{\#} + \beta_4^{\#} + \beta_5^{\#} + \beta_6^{\#} + \beta_7^{\#} \\ &= x_1\beta_1 + \theta_1 + x_2\beta_2 + \theta_2 + x_3\beta_3 + \theta_3 + x_4\beta_4 \\ &\quad + \theta_4 + x_5\beta_5 + \theta_5 + x_6\beta_6 + \theta_6 + x_7\beta_7 + \theta_7 \\ &= (x_1\beta_1 + x_2\beta_2 + x_3\beta_3 + x_4\beta_4 + x_5\beta_5 + x_6\beta_6 + x_7\beta_7) \\ &\quad + (\theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5 + \theta_6 + \theta_7), \\ l^{-1}k_{\text{bs-}10}\alpha_{10}^{\#} &= l^{-1}k_{\text{bs-}10}\left(k_{10\text{-}8}\alpha_8^{\#} + k_{10\text{-}9}\alpha_9^{\#}\right) \\ &= l^{-1}k_{\text{bs-}10} \\ &\quad \times \left(k_{10\text{-}8}\left(k_{8\text{-}3}\alpha_3^{\#} + k_{8\text{-}4}\alpha_4^{\#} + k_{8\text{-}5}\alpha_5^{\#}\right)\right. \\ &\quad \left. + k_{10\text{-}9}\left(k_{9\text{-}6}\alpha_6^{\#} + k_{9\text{-}7}\alpha_7^{\#}\right)\right) \\ &= l^{-1}k_{\text{bs-}10}k_{10\text{-}8}k_{8\text{-}3} \\ &\quad \times \left(k_{3\text{-}1}\left(\frac{l}{k_{\text{bs-}10}k_{10\text{-}8}k_{8\text{-}3}k_{3\text{-}1}}\right)x_1\beta_1\right. \\ &\quad + k_{3\text{-}2}\left(\frac{l}{k_{\text{bs-}10}k_{10\text{-}8}k_{8\text{-}3}k_{3\text{-}2}}\right)x_2\beta_2 \\ &\quad \left. + \left(\frac{l}{k_{\text{bs-}10}k_{10\text{-}8}k_{8\text{-}3}}\right)x_3\beta_3\right) \\ &\quad + l^{-1}k_{\text{bs-}10}k_{10\text{-}8} \\ &\quad \times \left(k_{8\text{-}4}\left(\frac{l}{k_{\text{bs-}10}k_{10\text{-}8}k_{8\text{-}4}}\right)x_4\beta_4\right. \\ &\quad \left. + k_{8\text{-}5}\left(\frac{l}{k_{\text{bs-}10}k_{10\text{-}8}k_{8\text{-}5}}\right)x_5\beta_5\right) \\ &\quad + l^{-1}k_{\text{bs-}10}k_{10\text{-}9} \\ &\quad \times \left(k_{9\text{-}6}\left(\frac{l}{k_{\text{bs-}10}k_{10\text{-}9}k_{9\text{-}6}}\right)x_6\beta_6\right. \\ &\quad \left. + k_{9\text{-}7}\left(\frac{l}{k_{\text{bs-}10}k_{10\text{-}9}k_{9\text{-}7}}\right)x_7\beta_7\right) \\ &= (x_1\beta_1 + x_2\beta_2 + x_3\beta_3 + x_4\beta_4 + x_5\beta_5 + x_6\beta_6 + x_7\beta_7) \\ &\quad + l^{-1}\theta(x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7). \end{aligned} \tag{11}$$

We can say that base station accepts the SUM aggregation $x_{10}^{\#}$, or the two tags will only verify if all tags are generated by honest nodes and aggregated correctly along the path.

Again, note that l^{-1} is the inverse of l modulo q which is the order of the elliptic curve group $E(Z_p)$, and θ, l, $k_{\text{bs-}10}$ are only known to base station.

5. Analysis

This section discusses the security and congestion complexity of EIPDAP.

5.1. Overview. Once a node has been compromised, it is under the full control of the adversary which can record and inject messages as will. We also assume that the adversary can only corrupt a (small) fraction of nodes including the aggregators. Also, we do not concern denial-of-service attack. The following is the proof for security of EIPDAP.

Definition 2 (sensor node inconsistency). Let $\langle x_t^{\#}, \alpha_t^{\#}, \beta_t^{\#} \rangle$ be a message sent by sensor node t. There is an inconsistency at node t if either

(1) $\alpha_t^{\#} \neq x_t k_{t,1} + x_t \beta_t k_{t,2}$ or

(2) $\beta_t^{\#} \neq x_t \beta_t + \theta_t$.

Definition 3 (sensor node forgery). An adversary eavesdropping on sensor node i successfully forges a new message $\langle x_i^*, \alpha_i^*, \beta_i^* \rangle$ if

(1) $x_i^* \neq x_i^{\#}$,

(2) $\alpha_i^* = x_i^* k_{i,1} + x_i^* \beta_i k_{i,2}$,

(3) $\beta_i^{\#} = x_i^* \beta_i + \theta_i$.

Since once a sensor node is compromised, the adversary can obtain all its confidential information (e.g., cryptographic keys) and send false data without being detected; however, we do not address this kind of forgery here.

Lemma 4. *Let the final SUM aggregation received by the base station be $x_{final}^{\#}$, then $S_L + \mu \leq x_{final}^{\#} \leq S_L + \mu v$ where S_L is the sum of the data values of all the legitimate nodes, and μ is the total number of corrupted nodes.*

Proof. As the conclusion is obvious here, so we do not prove in detail. □

Lemma 5. *If elliptic curve discrete logarithm problem is hard, then it is not possible to forge a valid message as an honest sensor node for all eavesdropping probabilistic, polynomial adversaries.*

Proof. Let $\langle x_i^{\#}, \alpha_i^{\#}, \beta_i^{\#} \rangle$ be an internal message sent by sensor node i to its parent.

Say adversary is eavesdropping on node i. In order to forge a valid message $\langle x_i^*, \alpha_i^*, \beta_i^* \rangle$ for x_i^*, A can easily compute a valid $\alpha_i^* = x_i^* x_i^{\#-1} \alpha_i^{\#}$.

As $\beta_i^{\#} = x_i \beta_i + \theta_i = x_i y G$ for some integer y, a valid β_i^* should be computed as

$$\begin{aligned} \beta_i^* &= x_i^* \beta_i + \theta_i = x_i^* y^* G \\ &= x_i^* y^* x_i^{-1} y^{-1} \beta_i^{\#} G. \end{aligned} \tag{12}$$

Due to θ_i and β_i, adversary cannot forge β_i^* directly from $x_i^* \beta_i + \theta_i$ but to compute $x_i^* y^* x_i^{-1} y^{-1} \beta_i^{\#} G$. A has x_i^*, x_i, $\beta_i^{\#}$, and G; the factors it lack are y and y^*.

Calculating y from $\beta_i^{\#} = x_i y G$ and y^* from $\beta_i^* = x_i^* y^* G$ is ECDLP, which means calculating β_i^* from $x_i^* y^* x_i^{-1} y^{-1} \beta_i^{\#} G$ is hard.

Another concern rises when adversary keeps eavesdropping on sensor node i and records the messages sent by i. Assume that adversary has

$$\left\{ \left\langle x_{i\#j}^{\#}, \alpha_{i\#j}^{\#}, \beta_{i\#j}^{\#} \right\rangle \mid j \in [1, n] \right\}, \tag{13}$$

where $\langle x_{i\#j}^{\#}, \alpha_{i\#j}^{\#}, \beta_{i\#j}^{\#} \rangle$ represents the message $\langle x_i^{\#}, \alpha_i^{\#}, \beta_i^{\#} \rangle$, node i sends to parent in the jth query, and n is the number of queries. Note that in each query, every sensor node i chooses a new secret key l_i.

For all $j \in [1, n]$, adversary has

$$\beta_{i\#j}^{\#} = \beta_{i\#j} x_{i\#j}^{\#} + \theta_i. \tag{14}$$

Because the number of variable is the number of equations plus one, so adversary cannot solve equations in (14) to obtain $\beta_{i\#j}$ or θ_i.

In conclusion, the probability of an adversary successfully forging a new message $\langle x_i^*, \alpha_i^*, \beta_i^* \rangle$ when eavesdropping on sensor node i is negligible, completing the proof. □

Definition 6 (aggregator inconsistency). Let $\langle x_t^{\#}, \alpha_t^{\#}, \beta_t^{\#} \rangle$ be an internal message aggregated by node t with two children u and v. Let $\langle x_u^{\#}, \alpha_u^{\#}, \beta_u^{\#} \rangle$ and $\langle x_v^{\#}, \alpha_v^{\#}, \beta_v^{\#} \rangle$ be two messages from u and v. There is an inconsistency at node t if

(1) $x_t^{\#} \neq x_t' + x_u^{\#} + x_v^{\#}$ or

(2) $\alpha_t^{\#} \neq \alpha_t' + \alpha_u^{\#} + \alpha_t^{\#}$ or

(3) $\beta_t^{\#} \neq \beta_u' + \beta_u^{\#} + \beta_t^{\#}$.

Definition 7 (compromised aggregator forgery). An adversary which compromised a aggregator j successfully forges a new aggregate result $\langle x_j^*, \alpha_j^*, \beta_j^* \rangle$ if

(1) $x_j^* \neq x^{\#}{}_j$,

(2) $\alpha_j^* = \alpha_j' + k_{j-1} \alpha_{j1}^{\#} + k_{j-2} \alpha_{j2}^{\#} + \cdots + k_{j-l}\ \alpha_{jl}^{\#}$ ($\alpha_j' = 0$ if j does not sense data),

(3) $\beta_j^{\#} = \beta_j' + \beta_{j1}^{\#} + \beta_{j2}^{\#} + \cdots + \beta_{jl}^{\#}$ ($\beta_j' = 0$ if j does not sense data), assuming aggregator j has l children $j1, j2, \ldots, jl$.

Lemma 8. *If elliptic curve discrete logarithm problem is hard, then it is not possible to forge a valid aggregate result for all probabilistic, polynomial adversaries even when a high-level aggregator is compromised.*

Proof. We assume that aggregator 10 has been compromised where an adversary is in complete control of node 10, obtaining all secret keys of node 10. Now an adversary attempts to forge SUM aggregation and two corresponding tags after eavesdropping several aggregations and records

$$\left\{ \left\langle x_{10\#i}^{\#}, \alpha_{10\#i}^{\#}, \beta_{10\#i}^{\#} \right\rangle \mid i \in [1, n] \right\}, \tag{15}$$

where $\langle x_{10\#i}^{\#}, \alpha_{10\#i}^{\#}, \beta_{10\#i}^{\#} \rangle$ represents the message $\langle x_{10}^{\#}, \alpha_{10}^{\#}, \beta_{10}^{\#} \rangle$, node 10 sends to base station in the ith query, and n is the number of queries. Note that in each query, every sensor node j chooses a new secret key l_j.

For all $i \in [1, n]$, adversary has

$$\begin{aligned} \beta_{10\#i}^{\#} &= l^{-1} k_{\text{bs-10}} \alpha_{10\#i}^{\#} \\ &= (\theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5 + \theta_6 + \theta_7) \\ &\quad - l^{-1} \theta x_{10\#i}^{\#}. \end{aligned} \tag{16}$$

Now adversary tries to forge a new message $\langle x_{10}^*, \alpha_{10}^*, \beta_{10}^* \rangle$ which satisfies (14). Since node 10 is compromised, adversary has $x_8^\#, x_9^\#, x_{10}^\#, k_{10\text{-}8}, k_{10\text{-}9}, \alpha_8^\#, \alpha_9^\#, \beta_8^\#, \beta_9^\#$ in each query and the knowledge of the elliptic curve group $E(Z_p)$.

Case 1. Intuitively, adversary tries to obtain l, $k_{\text{bs-}10}$, θ, and $\sum \theta_i$ (i: from 1 to 7). However, this requires a powerful adversary which we do not concern here.

Case 2. Adversary tries to compute $l^{-1}\theta$ by multiplying $k_{i,1}$ (equals θ/k_{path}) and the inverse of $k_{i,1}$ (equals l/k_{path}). However, all path-keys and temporal key are encrypted before forwarding to nodes, so adversary cannot compute $l^{-1}\theta$ when node 10 only works as an aggregator.

Case 3. Aggregator 10 also senses data. So adversary can compute $l^{-1}\theta$ as in Case 2 and $l^{-1}k_{\text{bs-}10}$ which is the inverse of $l/k_{\text{bs-}10}$ modulo q. Now adversary has

$$\begin{aligned} &\beta_{10}^\# - \alpha_{10}^{\#\#} \\ &\quad = \theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5 + \theta_6 + \theta_7 - x_{10}^{\#\#} \\ &\quad \Longrightarrow \beta_{10}^{\#\#} \\ &\quad = y_1 (\theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5 + \theta_6 + \theta_7) = y_2 G \end{aligned} \tag{17}$$

for some integer y_2. Similar to Lemma 4, since $\sum \theta_i$ (i: from 1 to 7) is kept secret from adversary and computing y_2 from $\beta_3^{\#\#} = y_2 G$ is ECDLP, then $\beta_{10}^{\#\#}$ cannot be forged either.

In all cases, adversary can only forge a new message $\langle x_{10}^*, \alpha_{10}^*, \beta_{10}^* \rangle$ with negligible probability, completing the proof. □

Theorem 9. *EIPDAP is integrity-preserving.*

Proof. From Lemmas 5 and 8, we know that EIPDAP is secure against sensor node forgery in the presence of an eavesdropper and aggregator forgery when an aggregator is compromised. Thus, EIPDAP is integrity-preserving, completing the proof. □

5.2. Congestion Complexity. The computational and memory costs are likely to be insignificant compared to communication [3, 14]. Higher computation surely causes more energy, but a Berkeley mote spends approximately the same amount of energy to compute 800 instructions as it does in sending a single bit of data [13, 20] in WSN.

Unlike general hard problems, there is no sub-exponential algorithm is known to solve the elliptic curve discrete logarithm problem (ECDLP), meaning that smaller parameters can be used in ECC than in other systems like RSA and DSA but with equivalent level of security. Because of their smaller key size, faster computations and reductions in processing power, storage space, and bandwidth, ECC is ideal for WSN. Although the use of elliptic curve cryptography incurs higher computational overhead than symmetric-key cryptography, our protocol is mainly designed to save energy.

In query dissemination phase, the base station collects aggregation tree information and broadcasts edge keys and path keys directly to the corresponding nodes. Collecting aggregation tree information costs each edge $O(1)$ congestion, and there is no congestion for sensor nodes and aggregators in broadcasting keys. In aggregation phase, each node forwards a message. The edge congestion in the aggregation tree is $O(1)$. In result-checking phase, all operations are done in the base station, so there is no congestion in the aggregation tree. Congestion complexity comparisons with Chan's scheme, Keith's scheme, and SDAP are shown in Tables 1, 2, and 3.

Table 1: Edge congestion in the aggregation tree comparison, n is the number of the nodes, and n_g is the group size.

	Query dissemination	Data aggregation	Result-checking
Chan's scheme	$O(1)$	$O(\log n)$	$O(\log^2 n)$
Keith's scheme	$O(1)$	$O(\log n)$	$O(\log n)$
SDAP	$O(1)$	$O(\Delta \log (n/n_g))$	$O(\Delta \log (n/n_g))$
EIPDAP	$O(1)$	$O(1)$	0

Table 2: Node congestion in the aggregation tree comparison, Δ is the degree of the aggregation tree.

	Query dissemination	Data aggregation	Result-checking
Chan's scheme	$O(\Delta)$	$O(\Delta \log n)$	$O(\Delta \log^2 n)$
Keith's scheme	$O(\Delta)$	$O(\Delta \log n)$	$O(\Delta \log n)$
SDAP	$O(\Delta)$	$O(\Delta \log (n/n_g))$	$O(\Delta \log (n/n_g))$
EIPDAP	$O(\Delta)$	$O(\Delta)$	0

Table 3: Aggregation tree congestion comparison.

	Query dissemination	Data aggregation	Result-checking
Chan's scheme	$O(n)$	$O(n \log n)$	$O(n \log^2 n)$
Keith's scheme	$O(n)$	$O(n \log n)$	$O(n \log n)$
SDAP	$O(n)$	$O(n)$	$O(n \log n)$
EIPDAP	$O(n)$	$O(n)$	0

By the comparison, we can conclude that EIPDAP has the minimum congestion and is much more energy efficient. Therefore it is much more suitable for power limited sensor networks.

6. Conclusion and Future Work

Protecting hierarchical data aggregation from losing integrity is a challenging problem in sensor networks. In this paper, we focus on the very problem of preserving data integrity and propose a novel approach to guarantee the integrity of aggregation result through aggregation in sensor networks. The main algorithm is based on performing modulo addition operation using ECC.

EIPDAP can immediately verify the integrity of aggregation result after receiving the aggregation result and corresponding authentication information, hence significantly

reducing energy consumption and communication delay which will be caused if the verification phrase is done through another query-and-forward phase.

Compared with the other related schemes, our scheme reduces the communication required per node to $O(\Delta)$, where Δ is the degree of the aggregation tree for the network. To the best of our knowledge, our scheme has the most optimal upper bound on solving the integrity-preserving data aggregation problem. Based on the elliptic curve discrete logarithm problem, we prove that EIPDAP is integrity-preserving.

In the future, we will first further enrich EIPDAP in detail. Second, we will focus on the possibility of reducing the number of secret keys shared between sensor nodes and base station or the keys broadcast to all nodes. Third, based on the proposed algorithm, we may consider meeting other security requirements, like data confidentiality, source authentication, and availability.

We anticipate that our work provides new perspective on preserving integrity of hierarchical aggregation and encourages other researchers to consider this approach.

Appendix

Attack on the Julia-Sanjay Scheme

If the adversary has compromised a sensor node, then it can obtain the network wide integer k. With the k, it can modify any aggregated data received from its child nodes. For example, say the adversary has compromised a node with message $\langle s_i, \text{enc}(m_i)\rangle$ received from its child i. In order to modify m_i to $m_i' = m_i + m_{\text{forge}}$, the adversary can easily forge an encrypted message:

$$\text{enc}\left(m_i'\right) = \text{enc}\left(m_i + m_{\text{forge}}\right) = \left\langle u_i, v_i + m_{\text{forge}}\right\rangle, \quad \text{(A.1)}$$

given as $\text{enc}(m_i) = \langle u_i, v_i\rangle$. And it is also easy to forge a valid signature:

$$s_i' = k^{-1}\left(m_i + m_{\text{forge}} + z_i * r\left(x\right)\right) \bmod p. \quad \text{(A.2)}$$

If the compromised node is in high level, this will cause more serious effects on the aggregation result since the aggregate result it handles represents large portions of the overall data in the WSN.

Acknowledgments

The authors would like to thank the anonymous reviewers and their coworkers for their valuable comments and useful suggestions.

References

[1] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[2] K. Akkaya, M. Demirbas, and R. S. Aygun, "The impact of data aggregation on the performance of wireless sensor networks," *Wireless Communications and Mobile Computing*, vol. 8, no. 2, pp. 171–193, 2008.

[3] L. Hu and D. Evans, "Secure aggregation for wireless networks," in *Proceedings of the Workshop on Security and Assurance in Ad Hoc Networks*, Orlando, Fla, USA, January 2003.

[4] B. Przydatek, D. Song, and A. Perrig, "SIA: Secure information aggregation in sensor networks," in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03)*, pp. 255–265, November 2003.

[5] W. He, X. Liu, H. Nguyen, K. Nahrstedt, and T. Abdelzaher, "Pda: privacy-preserving data aggregation in wireless sensor networks," in *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM '07)*, pp. 2045–2053, 2007.

[6] T. Feng, C. Wang, W. Zhang, and L. Ruan, "Confidentiality protection for distributed sensor data aggregation," in *Proceedings of the 27th IEEE Communications Society Conference on Computer Communications (INFOCOM '07)*, pp. 475–483, April 2007.

[7] J. Shi, R. Zhang, Y. Liu, and Y. Zhang, "PriSense: privacy-preserving data aggregation in people-centric urban sensing systems," in *Proceedings of the 29th IEEE International Conference on Computer Communications (INFOCOM '10)*, pp. 758–766, IEEE, March 2010.

[8] J. Katz and A. Y. Lindell, "Aggregate message authentication codes. Topics in cryptology," in *Proceedings of the Cryptographers' Track at the RSA Conference (CT-RSA '08)*, Lecture Notes in Computer Science, pp. 155–169, Springer, 2008.

[9] J. Albath and S. Madria, "Secure hierarchical data aggregation in wireless sensor networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '09)*, pp. 1–6, April 2009.

[10] A. Liu and P. Ning, "TinyECC: a configurable library for elliptic curve cryptography in wireless sensor networks," in *Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN '08)*, pp. 245–256, April 2008.

[11] H. Chan, A. Perrig, and D. Song, "Secure hierarchical in-network aggregation in sensor networks," in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS '06)*, pp. 278–287, ACM Press, Alexandria, Va, USA, November 2006.

[12] K. B. Frikken and J. A. Dougherty, "An efficient integrity-preserving scheme for hierarchical sensor aggregation," in *Proceedings of the 1st ACM Conference on Wireless Network Security (WiSec '08)*, pp. 68–76, ACM Press, Alexandria, Va, USA, April 2008.

[13] Y. Yang, X. Wang, S. Zhu, and G. Cao, "SDAP: a Secure hop-by-hop Data Aggregation Protocol for sensor networks," in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC '06)*, pp. 356–367, ACM Press, Florence, Italy, May 2006.

[14] C. Castelluccia, A. C. F. Chan, E. Mykletun, and G. Tsudik, "Efficient and provably secure aggregation of encrypted data in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 5, no. 3, pp. 1–36, 2009.

[15] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A witness-based approach for data fusion assurance in wireless sensor networks," in *Proceedings of the Global Telecommunications Conference (GLOBECOM '03)*, vol. 3, pp. 1435–1439, IEEE, December 2003.

[16] O. Eikemeier, M. Fischlin, J.-F. Götzmann et al., "History-free aggregate message authentication codes," in *Proceedings of the 7th International Conference on Security and Cryptography for Networks*, vol. 6280 of *Lecture Notes in Computer Science*, pp. 309–328, Amalfi, Italy, 2010.

[17] J. Girao, D. Westhoff, and M. Schneider, "CDA: concealed data aggregation in wireless sensor networks," in *Proceedings of the ACM Workshop on Wireless Security (WiSe '04)*, ACM Press, Philadelphia, Pa, USA, 2004.

[18] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient aggregation of encrypted data in wireless sensor networks," in *Proceedings of the 2nd Annual International Conference on Mobile and Ubiquitous Systems -Networking and Services (MobiQuitous '05)*, pp. 109–117, IEEE Computer Society, San Diego, Calif, USA, July 2005.

[19] H. Çam, S. Özdemir, P. Nair, D. Muthuavinashiappan, and H. Ozgur Sanli, "Energy-efficient secure pattern based data aggregation for wireless sensor networks," *Computer Communications*, vol. 29, no. 4, pp. 446–455, 2006.

[20] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "TAG: a tiny aggregation service for ad-hoc sensor networks," *SIGOPS Operating Systems Review*, vol. 36, pp. 131–146, 2002.