# FDAC: Toward Fine-Grained Distributed Data Access Control in Wireless Sensor Networks

Shucheng Yu, *Member*, *IEEE*, Kui Ren, *Member*, *IEEE*, and Wenjing Lou, *Senior Member*, *IEEE*

**Abstract**—Distributed sensor data storage and retrieval have gained increasing popularity in recent years for supporting various applications. While distributed architecture enjoys a more robust and fault-tolerant wireless sensor network (WSN), such architecture also poses a number of security challenges especially when applied in mission-critical applications such as battlefield and e-healthcare. First, as sensor data are stored and maintained by individual sensors and unattended sensors are easily subject to strong attacks such as physical compromise, it is significantly harder to ensure data security. Second, in many mission-critical applications, fine-grained data access control is a must as illegal access to the sensitive data may cause disastrous results and/or be prohibited by the law. Last but not least, sensor nodes usually are resource-constrained, which limits the direct adoption of expensive cryptographic primitives. To address the above challenges, we propose, in this paper, a distributed data access control scheme that is able to enforce fine-grained access control over sensor data and is resilient against strong attacks such as sensor compromise and user colluding. The proposed scheme exploits a novel cryptographic primitive called attribute-based encryption (ABE), tailors, and adapts it for WSNs with respect to both performance and security requirements. The feasibility of the scheme is demonstrated by experiments on real sensor platforms. To our best knowledge, this paper is the first to realize distributed fine-grained data access control for WSNs.

**Index Terms**—Data access control, wireless sensor network, distributed storage, attribute-based encryption.

## 1 Introduction

Wireless sensor networks (WSNs) have been an area of significant research in recent years [2], [3], [4], [5], [6]. A WSN usually consists of a large number of sensor nodes that can be easily deployed to various terrains of interest to sense the environment. WSNs have found their wide applications in both civilian and military domains. To accomplish the targeted application and fulfill its functionalities, a WSN usually generates a large amount of data continuously over its lifetime. One of the biggest challenges then is how to store and access these sensed data.

Existing data storage and access approaches in WSNs can be roughly divided into two branches, namely, centralized and distributed approaches [7]. In the centralized case, sensed data are collected from individual sensors and transmitted back to a central location, usually the sink, for storage and access. In the distributed approach, after a sensor node has generated some data, it stores the data locally or at some designated nodes within the network instead of immediately forwarding the data to a centralized location out of the network. The stored data later on can be accessed in distributed manner by the users of the WSN. Compared to the centralized case, distributed data storage and access consumes less bandwidth since sensed data are no longer transmitted to a centralized location out of the network. As energy-efficient storage devices are now possible to be equipped with sensor nodes [8], [9], [10], [11] thanks to recent advances in IC manufacturing, reading and writing data on local flash memory become much more efficient than transmitting over radio. Employment of distributed data storage and access thus also implies energy efficiency. In addition, distributed data storage and access can avoid weaknesses such as single point of failure, performance bottleneck, which are inevitable in the centralized case. These advantages together have led to the recent increasing popularity of distributed data storage and access [12], [13], [14], [15], [16].

As a large amount of sensed data are distributedly stored in individual sensor nodes, data security naturally becomes a serious concern. Actually, in many application scenarios, data sensed by WSNs are closely related to security and/or privacy issues and should be accessible only to authorized users. Moreover, in a mission-critical application scenario, various types of data generated by all kinds of sensors may belong to different security levels, and thus are meant to be accessed only by selected types of users. That is, accessibility of a particular type of data to users is solely based on necessity. For example, in the battlefield scenario, the general should be able to access all types of data for the purpose of overall coordinating but a soldier may only need to access the type of data relevant to his mission. In this way, the security of data can be best protected as, for example, a soldier has much larger risk of being compromised as compared to the general, and a tank is much better protected than a simple mobile sink. With such a fine-grained data access control, we can effectively minimize the negative consequence due to user compromise. However, past research on data security in WSNs mainly focused on communication security, such as key management, message

- *S. Yu and W. Lou are with the Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, 100 Institute Road, Worcester, MA 01609. E-mail: {yscheng, wjlou}@ece.wpi.edu.*
- *K. Ren is with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, 3301 Dearborn St, Chicago, Illinois 60616. E-mail: kren@ece.iit.edu.*

*Manuscript received 12 June 2009; revised 14 Mar. 2010; accepted 8 May 2010; published online 16 June 2010.*
*Recommended for acceptance by X. Tang.*
*For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2009-06-0267.*
*Digital Object Identifier no. 10.1109/TPDS.2010.130.*

authentication, intrusion detection, etc., [17], [18], [19], [20], [21], [22]. Distributed data storage and access security [23] has gained limited attention so far, not to mention fine-grained data access control. This becomes a more severe issue given the trend that more and more distributed data storage and retrieval schemes are being proposed.

To provide distributed data access control, a naive solution is to equip each sensor node with an access control list (ACL) as is usually adopted in wired networks. Upon each data access request, the sensor node verifies the user's identity with the ACL, and the access request is approved only if the user is in the list. However, this solution is not applicable to WSNs due to the following facts. First, sensor nodes are often deployed without physical protection and lack of tamper-resistant hardware. Attackers may capture and compromise sensor nodes, and then read historical data out of the memory of sensor nodes. Second, the ACL method is not scalable as it requires the sensor nodes to remember each legitimate user. For the purpose of finding a secure yet efficient solution for fine-grained distributed data access control in WSNs, we naturally shift our attention to data encryption which would introduce two branches, namely symmetric key cryptography (SKC) based approaches and public key cryptography (PKC) based ones [24], [25].

In SKC-based approaches, data encryption and decryption share the same key. If the attacker has compromised the sensor node, he is able to read the data encryption key stored in the sensor's memory and thus decrypt the historical data generated by the same sensor. To avoid this kind of attacks, a natural solution is to divide the lifetime of each sensor into series of periods, and the data encryption keys for these periods are independent of each other. Each sensor just stores the data encryption key for current period, and erases all the previously used keys. The problem that follows is to efficiently update data encryption keys for sensor nodes as well as distribute the keys to legitimate users. State-of-the-art SKC-based approaches adopt techniques such as perturbed polynomial [26] to manage the keys. However, current SKC-based approaches have two major drawbacks: first, fine-grained data access control is hard to realize due to the complexity of key management; second, collusion attacks are possible given an appropriate number of colluding users [27]. Therefore, further research is still desired for fine-grained distributed data access control using SKC-based approaches.

PKC-based approaches can provide better data access security than their SKC-based peers. In such approaches, sensor nodes encrypt the data items with public keys. One apparent advantage of this is that if data storage sensors are compromised, the attacker will not be able to recover the stored data due to lack of the corresponding private keys. Therefore, by applying PKC-based approaches to data access control in WSNs, we can immediately enjoy the perfect resilience against sensor compromise. In traditional public key cryptosystems including identity-based encryption, the encryption is usually targeted to only one recipient, in the sense that any message encrypted using a particular public key can be decrypted only with the corresponding secret key. However, for the purpose of distributed data access control in WSNs, the fundamental encryption paradigm is one-to-many such that one encrypted data item can be decrypted by a number of different authorized users. To achieve this goal, a straightforward approach is to use one-to-one public key cryptosystems, which is obviously inefficient since both the number of encryption operations and the size of ciphertexts are linear to the total number of authorized users. A better solution is broadcast encryption [28], [29], [30], [31], which achieves improved efficiency. But it requires receivers to be represented individually. An encryptor must have the priori knowledge of all the prospective receivers as well as the authorization information associated with each receiver. However, in our targeted applications, it is desirable for sensors to be able to encrypt without exact knowledge of the set of intended receivers. This is because when the WSN is deployed, it maybe impossible to know the exact information of its future users.

From the above discussion, it is clear that achieving fine-grained data access control with efficiency is still an open challenge in WSNs. Toward addressing this challenge, we propose in this paper, a Fine-grained Distributed data Access Control scheme, namely FDAC, specially tailored for WSNs. We base our design on the observation of the inherent nature of the sensor data. As WSNs are, in general, deployed for specific application(s), it is usually easy and convenient to specify individual sensors (and hence their collected data) through a set of predefined attributes such as sensor type, location, time, owner, etc. We further find that this nice property can also be utilized to describe data accessibility in a very expressive manner, that is, it can allow fine-grained tunable data access control. Based on this observation, we propose to associate each attribute of sensor nodes with a predefined keying material. And then, we further examine each user of the WSN with respect to his data access privilege and associate him with an access structure accordingly. Such an access structure in our design is implemented via an access tree which specifies the types of data that this user is authorized to access. Sensor data are then protected by being encrypted under their attributes such that only the users whose access structures satisfy the required data attributes can decrypt. In the access structure, every leaf node maps to a sensor/data attribute, and the interior nodes can be threshold gates. The access structure thus can represent sophisticated logic expressions over the attributes, that is, be able to specify data access privileges of users in the fine-grained manner. By exploring a novel PKC primitive called key-policy attribute-based encryption (KP-ABE), we seamlessly integrate our access structure with data encryption. Our solution has several advantages. First, FDAC is efficient in terms of key storage, computation, and communication overhead at the sensor node side. Second, it is resistant against user collusion, i.e., the cooperation of colluding users will not lead to the disclosure of additional sensor data. Last but not least, FDAC provides efficient user revocation via a single broadcast, and the length of the broadcast message for sensor nodes is only of several hundred bits.

In summary, our paper makes the following contributions: 1) It introduces the fine-grained data access control problem for the first time in WSNs. 2) FDAC applies and tailors KP-ABE to WSNs for achieving fine-grained data

access control. 3) The applicability of FDAC is demonstrated on the current generation of sensor nodes.

The rest of this paper is organized as follows: Section 2 discusses our system models and assumptions as well as some technical preliminaries on which our scheme is based. In Section 3, we present our scheme in detail. Section 4 analyzes our scheme in terms of security and performance. We conclude this paper in Section 5.

## 2 Models and Assumptions

### 2.1 Network Model

In this work, we consider a wireless sensor network composed of a network controller which is a trusted party, a large number of sensor nodes, and many users. Throughout this paper, we will denote the network controller with the symbol $\mathcal{T}$. Symbol $\mathcal{U}$ and $\mathcal{N}$ are used to represent the universe of the users and the sensor nodes, respectively. Both users and sensor nodes have their unique IDs. Symbol $\mathcal{U}_i$ will be used to denote user $i$, and $\mathcal{N}_j$ to represent sensor node $j$. The trusted party $\mathcal{T}$ can be online or offline. It comes online merely on necessity basis, e.g., in the case of intruders detected. Each sensor could be a high-end sensor node such as iMote2 which has greater processing capability and a larger memory than conventional sensor nodes. Sensor data could be stored locally or at some designated in-network locations [32]. As is conventionally assumed, we consider each user $\mathcal{U}_i$ to have sufficient computational resources to efficiently support expensive cryptographic operations such as bilinear map (cf. Section 2.4.1). In addition, we assume there is a loose time synchronization among the sensor nodes, and the lifetime of the network is further divided into phases based on the time synchronization. Such an assumption can also be found in previous works, such as [26].

### 2.2 Adversary Model

This paper considers attackers whose main goal is to learn about the contents of sensor data that they are not authorized to. The adversaries could be either external intruders or unauthorized network users. Due to lack of physical protection, sensor nodes are usually vulnerable to strong attacks. In particular, we consider the adversary with both passive and active capabilities, which can 1) eavesdrop all the communication traffics in the WSN, and 2) compromise and control a small number of sensor nodes. In addition, 3) unauthorized users may collude to compromise the encrypted data.

### 2.3 Security Requirements

For the purpose of securing distributed data storage in WSNs, the main goal of this work is to protect contents of sensor data from being learned by attackers, including external intruders and unauthorized network users. With respect to data access control in WSNs, we recognize the following unique but not necessarily complete security requirements.

#### 2.3.1 Fine-Grained Data Access Control

In many application scenarios, especially mission-critical cases, disclosure of sensitive data should be well controlled such that different users may have access privileges over different types of data. For this purpose, we need to define and enforce a flexible access policy for each individual user based on the user's role in the system. In particular, the access policy should be able to define a unique set of data that the user is authorized to access, and must be enforced via a cryptographic method since sensor nodes are vulnerable to strong attacks like physical compromise.

#### 2.3.2 Collusion Resilience

As described by our adversary model, unauthorized users may cooperate for the purpose of learning about the contents of sensitive data. This requires our data access control scheme to be resilient to collusion attacks in the sense that the collaboration of unauthorized users will not give them additional advantages over what they can directly obtain from executing attacks individually.

#### 2.3.3 Sensor Compromise Resistance

Due to lack of compromise-resistant hardware, a small number of sensor nodes could be physically compromised by the adversary in hostile environments. Now that the adversary can always obtain the sensor data generated by a sensor node after it is compromised, we should at least secure sensor data such that, 1) compromising the sensor node does not disclose the sensor data generated before the sensor is compromised, and 2) compromising one sensor node does not give the adversary any assistance to obtain sensor data generated by other sensor nodes.

#### 2.3.4 Backward Secrecy

User management is an important functionality required by most application scenarios. In particular, the system should be able to handle user revocation in the case of user leaving request or malicious behavior detected. To support such a functionality, the data access control mechanism should guarantee that the revoked users are not able to access the sensor data generated after they are revoked.

### 2.4 Preliminaries

This section briefly describes the technique preliminaries on which our scheme is designed.

#### 2.4.1 Bilinear Map

Our design is based on some facts about groups with efficiently computable bilinear maps.

Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be multiplicative cyclic groups of prime order $p$. Let $g_1$ and $g_2$ be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. A bilinear map is an injective function $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ with the following properties:

1. *Bilinearity*: for $\forall u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$, $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. *Nondegeneracy*: $e(g_1, g_2) \neq 1$.
3. *Computability*: There is an efficient algorithm to compute $e(u, v)$ for each $u \in \mathbb{G}_1$ and $v \in \mathbb{G}_2$.

#### 2.4.2 Key-Policy Attribute-Based Encryption

In KP-ABE [33], each ciphertext is associated with a set of descriptive attributes. Each private key is associated with an access structure that specifies which type of ciphertexts the key can decrypt. A user is able to decrypt a ciphertext if and only if the attributes associated with a ciphertext satisfy the key's access structure. A KP-ABE scheme is composed of four algorithms:

**Setup.** This algorithm takes as input a security parameter $\kappa$. and returns the public key $PK$ as well as a system master secret key $MK$. $PK$ is used by message senders for encryption. $MK$ is used to generate user secret keys and is known only to the authority party.

**Encryption.** This algorithm takes a message $m$, the public key $PK$, and a set of attributes $\gamma$ as input. It outputs the ciphertext $E$.

**Key generation.** This algorithm takes as input an access structure $\mathcal{P}$, the master secret key $MK$, and the public key $PK$. It outputs a secret key $SK$ that enables the user to decrypt a message encrypted under a set of attributes $\gamma$ if and only if $\gamma$ matches $\mathcal{P}$.

**Decryption.** It takes as input the ciphertext $E$, which was encrypted under the attribute set $\gamma$, the user's secret key $SK$ for access structure $\mathcal{P}$, and the public key $PK$. This algorithm outputs the message $m$ if and only if the attribute set $\gamma$ satisfies the user's access structure $\mathcal{P}$.

Please refer to [33] for more details on KP-ABE algorithms.

## 3 FDAC: Fine-Grained Data Access Control Scheme

This section presents our data access control scheme for distributed data storage in WSNs. We first introduce our access control strategy. Next, we give an overview of FDAC. Then, we present the detailed description of our basic scheme, which is followed by an advanced design.

### 3.1 Access Control Strategy

For the purpose of achieving fine-grained data access control in WSNs, we first explore some inherent natures of WSNs. In general, the deployments of most WSNs are aimed at collecting certain types of data for specific application(s). Therefore, we are able to specify individual sensors (and hence the data collected by them) through a set of predefined attributes. For example, in the battlefield, sensor nodes are usually deployed to collect military information in certain geographic location. Each sensor node maybe responsible for collecting specific types of data such as vibration, smoke, so on and so forth. Sensor nodes may also have their owners, i.e., persons or units who are in charge of them. In particular, some nodes maybe jointly owned by different units. Hence, we may specify sensor nodes using these attributes such as {location = *village*, data type = (*vibration*, *smoke*), owner = (*explosion experts*, *officers*, *scouts*)}. This further enables us to specify data access privileges of users based on these attributes. In the above example, we may designate the access structure of a user as "(location is village) AND (type is vibration)", which allows the users to obtain vibration data within the village area. We may also define more sophisticated access structures such as "(location is village) AND (type is vibration OR smoke) AND (at least owned by two of the following: explosion experts, officers, scouts)". In this case, the user can only access vibration and smoke data collected within the village area. In addition, the last condition implicitly requires the user to belong to at least two of the three designated groups. To enforce these access structures, we predefine a public key component for each of the attributes, and encrypt sensor data with public key components of the corresponding attributes such that only the users with "satisfiable" access structures[1] are able to decrypt.

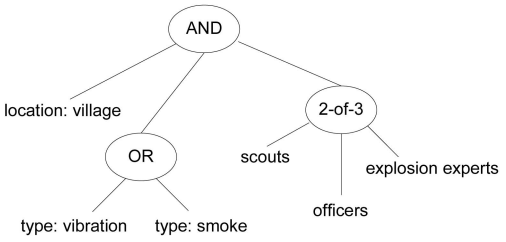



Fig. 1. An example access structure in the battlefield scenario.

Having discussed the intuitive idea of our data access control strategy, we further present it more formally as follows: In FDAC, we associate each sensor node (and hence its collected data) with a set of attributes, for each of which we define a public key component. Each user is assigned an access structure, which is implemented via an access tree and embedded in the user's secret key. Every leaf node of the access tree is labelled with an attribute and the interior nodes are defined as threshold gates.[2] This kind of definition of user access structure is able to represent very expressive logic expressions over attributes, and thus specify data access privileges of users in the fine-grained manner. Actually, we are able to represent any general (monotone or nonmonotone) access structures if we define the NOT of each attribute as a separate attribute, which, in turn, will double the number of attributes in our system. Fig. 1 illustrates the aforementioned access structure in the battlefield scenario.

Notably, the above fine-grained data access control strategy can also be realized in an alternative way as follows: We define a set of attributes for each user to reflect the user's role in the system, and associate each sensor node with an access structure which defines the logic combination over the intended user attributes and is implemented via an access tree. Sensor data later on are encrypted under the access structure such that only the users whose attributes satisfy the access structure are able to decrypt. In this way, we are able to realize the same functionality in terms of fine-grained data access control. However, this approach may cause efficiency issues in terms of computation and communication overhead on sensor nodes when applied to WSNs. This is because the complexity for data encryption and thus the ciphertext size in the alternative approach is linear to the size of the sensor node's access tree. The size of the access tree could potentially be very large since we may assign a very complicated access structure to the sensor node to precisely define the set of intended users. In our proposed strategy, however, the complexity in terms of computation and communication overhead is linear to the number of data attributes assigned to the sensor node. In practical settings, we can specify each sensor node (and hence its collected data) via a small number of attributes due to the limited functionalities of each sensor, though the attribute universe in the whole network could be large. Since sensor nodes are usually not resource abundant, our proposed strategy is more applicable to WSNs than the alternative one.

1. That is to say, the logic expression represented by the access structure returns TRUE over the data attributes.

2. A $t$-of-$n$ threshold gate outputs TRUE if and only if at least $t$ out of the $n$ inputs are TRUE. Two extreme examples are AND gates ($n$-of-$n$) and OR gates (1-of-$n$).

Formally, in FDAC, we will denote the universe of all the sensor attributes in a WSN by a symbol $\mathcal{I}$. The set of attributes owned by each single sensor node is denoted by a symbol $\mathcal{I}_i$, where $i$ is the sensor node ID. We have $\mathcal{I} = \bigcup_{\forall i \in \mathcal{N}} \mathcal{I}_i$. Let $k = \max_{\forall i \in \mathcal{N}} |\mathcal{I}_i|$. $k$ will be a system parameter used by our scheme. The access structure is generally denoted by $\mathcal{P}$.

### 3.2 Scheme Overview

In our basic scheme, each sensor node is preloaded with a set of attributes as well as the public key $PK$. Each user is assigned an access structure and the corresponding secret key $SK$. As mentioned in Section 2.1, the lifetime of the sensor network is divided into *phases*, each of which has the same time duration. Based on this, we further define each $n$ consecutive phases as a *stage*, where $n < k$ and $k = \max_{\forall i \in \mathcal{N}} |\mathcal{I}_i|$ is a system parameter. Therefore, the lifetime of the sensor network can also be represented by a series of consecutive *stages*, numbering as $1, 2, \ldots, m$, where $m$ is a system parameter. Sensor nodes encrypt sensed data using a symmetric key algorithm, e.g., AES. Over each phase, every sensor node updates its data encryption key once in the way that the data encryption keys during one stage form a one-way key chain. The key update algorithm could be any standard one-way hash function such as SHA-1. We call the first key on this key chain by the $master\ key$, denoted by $K$. The master key of each stage is always generated during its preceding stage, and encrypted under the preloaded attributes. Upon request for sensor data, the sensor node responds with the encrypted master key as well as the ciphertext of the sensor data. If the user is an intended receiver, he is able to decrypt the master key and derive the data encryption keys for phases of his interest, and thus decrypt the sensor data. Based on the basic scheme, our advanced scheme goes one step further by providing the functionality of user revocation, which is demanded by most WSN application scenarios. In the advanced scheme, $\mathcal{T}$ is able to revoke any user via broadcasting a user revocation message to all the users and all the sensor nodes, respectively. In particular, the user revocation message for the sensor nodes contains merely one group element of $\mathbb{G}_T$.

### 3.3 The Basic Scheme

The construction of our basic scheme based on KP-ABE [33] and the one-way key chain is as follows:

#### 3.3.1 System Initialization

On initialization, $\mathcal{T}$ executes the following steps:

1. Select two multiplicative cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of prime order $p$ as well as a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$. Let $g$ be the generator of $\mathbb{G}_1$.
2. Choose a number $t_i$ uniformly at random from $\mathbb{Z}_p$ for each attribute $i \in \mathcal{I}$, and $y$ randomly from $\mathbb{Z}_p$. Output the public key as follows:

$$PK = \langle G_1, g, Y = e(g,g)^y, \quad T_1 = g^{t_1}, \ldots, T_{|\mathcal{I}|} = g^{t_{|\mathcal{I}|}} \rangle.$$

   The master secret key is $MK = (y, t_1, \ldots, t_{|\mathcal{I}|})$.
3. Choose a secure one-way hash function, denoted as $h(\cdot)$. Preload the following information to each sensor node $\mathcal{N}_i$:

$$\mathcal{T} \rightarrow \mathcal{N}_i : \mathcal{I}_i, h(\cdot), PK.$$

4. For each user $\mathcal{U}_j$, $\mathcal{T}$ generates an access structure $\mathcal{P}$ and computes his secret key $SK$ as follows: Starting from the root node $r$ of $\mathcal{P}$ and in the top-down manner, construct a random polynomial $q_x$ of degree $d_x + 1$ using Lagrange interpolation for each node $x$ in $\mathcal{P}$, where $d_x$ is the threshold value of node $x$. For each nonroot node $x$ in $\mathcal{P}$, set $q_x(0) = q_{parent(x)}(index(x))$, where $parent(x)$ is the parent of $x$ and $index(x)$ is the unique index number of $x$ given by its parent. In particular, set $q_r(0) = y$. $SK$ is output as follows:

$$SK = \langle \{ D_i = g^{\frac{q_i(0)}{t_i}} \}_{i \in \mathcal{L}} \rangle,$$

   where $\mathcal{L}$ denotes the set of leaf nodes in $\mathcal{P}$. Then, $\mathcal{U}_j$ is preloaded with the following information:

$$\mathcal{T} \rightarrow \mathcal{U}_j : \mathcal{P}, SK, h(\cdot), PK.$$

#### 3.3.2 Master Key Encryption

During each stage $v \in [1, m]$, $\mathcal{N}_i$ generates a new master key for stage $v + 1$ and encrypts it as follows:

1. Select a number $s$ uniquely at random from $\mathbb{Z}_p$.
2. On each phase of stage $v$, calculate one item $E_i = T_i^s$ for attribute $i \in \mathcal{I}_i$. After $|\mathcal{I}_i|$ phases, $|\mathcal{I}_i| \leq k$, $\mathcal{N}_i$ has the complete set $\{E_i = T_i^s\}_{i \in \mathcal{I}_i}$.
3. Randomly select a number $K \in \mathcal{K}$ as the master key of the key chain, where $\mathcal{K}$ denotes the key space. Then, compute $E' = KY^s$. Finally, store the ciphertext as follows:

$$E^{v+1} = \langle v + 1, \mathcal{I}_i, KY^s, \{ E_i = T_i^s \}_{i \in \mathcal{I}_i} \rangle,$$

   where $E^{v+1}$ represents the encrypted master key for the $(v+1)$th stage.

#### 3.3.3 Data Storage

$\mathcal{N}_i$ encrypts and stores the sensor data generated in the current phase, say phase $t \in [1, n]$ of stage $v \in [1, m]$, as follows:

1. Calculate the data encryption key $K_t = h(K_{t-1})$. In particular, we set $K_0 = K$.
2. Encrypt the sensor data, denoted by $D$, with current data encryption key $K_t$. Then, store the item $\langle v, t, \{D\}_{K_t} \rangle$, where $\{D\}_{K_t}$ represents the encrypted sensor data.
3. Erase $K_{t-1}$ from the memory.

For each sensor node, all the data encryption keys used during one stage form a one-way key chain. The sensor node just keeps the latest data encryption key in its memory, while erasing all the previous ones.

#### 3.3.4 Data Access

Assume user $\mathcal{U}_j$ is requesting for sensor data generated by sensor node $\mathcal{N}_i$ during phase $t$ of stage $v$. $\mathcal{N}_i$ responds the data query request with the following message:

$$\mathcal{N}_i \rightarrow \mathcal{U}_j : \langle E^v, \{D\}_{K_t} \rangle.$$

On receiving the response from $\mathcal{N}_i$, $\mathcal{U}_j$ executes the following steps to obtain the sensor data:

1. Decrypt the master key $K$ of stage $v$ from $E^v$. The decryption process starts from the leaf nodes and in the bottom-up manner. First, $\mathcal{U}_j$ computes the value $F_i$ for each leaf node $i$ in $\mathcal{P}$ as follows:

$$F_i = \begin{cases} e(D_i, E_i) = e(g,g)^{sq_i(0)}, & \text{if } i \in \mathcal{I}_i; \\ \perp, & \text{otherwise.} \end{cases}$$

   Then, it proceeds in the recursive way from the second last layer as follows: for node $x$ which is a $d_x$-of-$n$ gate, if more than $n - d_x$ children returns $\perp$, $F_x = \perp$. Otherwise,

$$F_x = \prod_{i \in S_x} F_i^{\delta_i(0)} = \prod_{i \in S_x} e(g,g)^{sq_i(0)\delta_i(0)} = e(g,g)^{sq_x(0)},$$

   where $S_x$ denotes the set of $x$'s children and $\delta_i(0)$ is the Lagrange coefficient which can be calculated by the user himself. If $\mathcal{P}$ "accepts" $\mathcal{I}_i$, $\mathcal{U}_j$ will finally obtain $e(g,g)^{sq_r(0)} = e(g,g)^{sy}$ and thus decrypt the master key $K$. Otherwise, the decryption algorithm returns $\perp$.
2. If the decryption algorithm returns $\perp$, terminate. Otherwise, $\mathcal{U}_j$ calculates the data encryption $K_t$ from $K$ by $K_t = h^t(K)$, and finally decrypts the sensor data with $K_t$.

In this basic scheme, we assign each sensor node a set of attributes and each user an access structure. Sensor data are encrypted under the attributes such that only the users with "satisfiable" access structures are able to decrypt. As the access structure is very expressive, we are able to precisely control the access privilege of each user, and thus enjoy fine-grained data access control. The access policies in the basic scheme are actually enforced by using KP-ABE [33]. To alleviate the computation overload, we divide the lifetime of sensor nodes into stages and phases. On each stage a master key is generated to serve as the "seed" for the data encryption keys of the underlying phases. For the purpose of access control, we just need to encrypt the master key of each stage under the attribute-based encryption algorithm. Sensor data are encrypted using symmetric key encryption such as AES which is very efficient. As master keys are generated at a relative low frequency, we are able to distribute the computation overload of attribute-based encryption into each phase and thus make the expensive operations affordable to the sensor nodes.

#### 3.3.5 User Revocation

Another fundamental functionality of WSNs is user management. In particular, we stress that the network operator should be able to revoke the user's access privilege when necessary. In our basic scheme, we can use the following approaches to revoke users from the system: one approach is to define some time attributes [34], and embed an expiration date to each user's access structure based on the time attributes. Sensor nodes can then associate a time stamp to each ciphertext using the time attributes. If sensor nodes always associate the current time stamp to ciphertexts, users will be automatically revoked after their designated expiration dates. Another approach for user revocation is to define some "identity attributes" [35], e.g., defining a binary attribute for each bit of user identity, and associate the corresponding identity attributes to each user's access structure. Sensor nodes can then associate any intended user list with each ciphertext using the "identity attributes." To revoke a user, sensor nodes can encrypt data using a selected set of "identity attributes" which exclude the revoked user's identity. The advantage of the two approaches is that they do not involve extra communication with users. However, the limitation of them is also obvious. For the first approach, users can only be revoked at a predefined time. It does not support user revocation on the fly. The second approach is "stateful," i.e., every ciphertext (and hence the sensor nodes) needs to remember all the revoked users in the history. The ciphertext size would keep increasing as more and more random users are revoked, which ends up with a heavy computation and communication overhead on each sensor node after several rounds of user revocation. To resolve this issue, in this work, we propose to update secret keys of all the users but the one(s) to be revoked. More specifically, we will update a common master key component which is embedded into every user's secret key as we will discuss in detail. The benefits of this key update method can be summarized as follows: First, this approach is "stateless" and sensor nodes do not need to "remember" any revoked user in the history. Second, the user revocation process does not introduce too much communication or computation overhead on each sensor node. Actually, each sensor node just needs to update one of its public key components which can be efficiently achieved by broadcasting the common public key component to all the sensor nodes. Consequently, the affect of user revocation on each sensor node is minimal. The main issue with the proposed solution, however, is that it needs every user to communicate with the authority via unicast to update his secret key. To resolve this issue, we revise the original KP-ABE construction so that we can update secret keys for all nonrevoked users by broadcasting a common element to them, which can be efficiently realized by existing broadcast encryption techniques. We also prove that our revision to KP-ABE has the same security strength as the original construction in terms of semantic security of data.

### 3.4 The Advanced Scheme

The basic idea of our advanced user revocation solution is to separate the master secret key $y$ from the user access structure in the user secret key $SK$. Update of user secret keys can thus be realized by updating the embedded secret $y$ which is common to every user's secret key. As a result, we can update user secret keys via broadcasting the incremental of $y$ while excluding the leaving user from the recipient list. Based on this general idea, we present our advanced scheme as follows: For brevity, we just present the parts that need to be changed as compared to our basic scheme.

#### 3.4.1 System Initialization

$\mathcal{T}$ executes the following steps:

1. The same as step a) of Section 3.3.1 in the basic scheme.
2. In addition to the elements generated by step b) of Section 3.3.1 in the basic scheme, $\mathcal{T}$ selects a number

$\beta$ uniquely at random from $\mathbb{Z}_p$. The public key $PK$ and the master secret key $MK$ are then output as follows:

$$PK = \langle G_1, g, Y, \{T_i = g^{t_i}\}_{i\in\mathcal{I}}, g^{\beta}\rangle,$$
$$MK = \langle y, t_1, \ldots, t_{|\mathcal{I}|}, \beta\rangle.$$

3. The same as step c) of Section 3.3.1 in the basic scheme.
4. Sensor node $\mathcal{N}_i$ is preloaded with the following

$$\mathcal{T} \rightarrow \mathcal{N}_i : \mathcal{I}_i, h(\cdot), PK.$$

5. The process of key generation is similar to step d) of Section 3.3.1 in the basic scheme. $\mathcal{T}$ outputs the user secret key $SK$ as follows:

$$SK = \langle g^{\frac{y-\theta}{\beta}}, \{D_i = g^{\frac{q_i(0)}{t_i}}\}_{i\in\mathcal{L}}\rangle.$$

Compared to the basic scheme, this algorithm introduces a new element $g^{\frac{y-\theta}{\beta}}$ into $SK$, where $\theta = q_r(0)$ is randomly selected from $\mathbb{Z}_p$, and $q_r$ denotes the polynomial for the root node $r$ in $\mathcal{P}$. $\mathcal{U}_j$ is then preloaded with $\langle \mathcal{P}, SK, h(\cdot), PK\rangle$.

### 3.4.2 Master Key Encryption

Similar to Section 3.3.2 in the basic scheme. The advanced scheme introduces a new element $g^{\beta s}$ into the ciphertext as follows:

$$E^{v+1} = \langle v+1, \mathcal{I}_i, KY^s, \{E_i = T_i^s\}_{i\in\mathcal{I}_i}, g^{\beta s}\rangle.$$

### 3.4.3 Data Storage

The same as Section 3.3.3 in the basic scheme.

### 3.4.4 Data Access

This part is the same as Section 3.3.4 in the basic scheme except for step a).

1. The decryption process is similar to that in the basic scheme. When the data attributes satisfy the user's access structure $\mathcal{P}$, the user obtains $e(g,g)^{\theta s}$. Then, he decrypts the message as follows:

$$M = \frac{Me(g,g)^{ys}}{e\left(g^{\frac{y-\theta}{\beta}}, g^{\beta s}\right)e(g,g)^{\theta s}}.$$

In this advanced scheme, $\mathcal{T}$ is able to update the master secret key $y$ embedded in the user secret key $SK$ by broadcasting $g^{\frac{\Delta y}{\beta}}$ to the users, where $\Delta y$ is the incremental of $y$. With the above enhancement, we can present our user revocation scheme as follows.

### 3.4.5 User Revocation

To revoke a user $\mathcal{U}_j$, $\mathcal{T}$ needs to update the master secret key $y$ for the sensor nodes as well as the remaining users. The process can be illustrated as follows:

$$\mathcal{T} : y' \leftarrow \mathbb{Z}_p, \Delta y \leftarrow y' - y, Y' \leftarrow e(g,g)^{y'}, g^{\frac{\Delta y}{\beta}}$$
$$\mathcal{T} \rightarrow \mathcal{N} : Y'$$
$$\mathcal{T} \rightarrow \mathcal{U}\backslash\mathcal{U}_j : g^{\frac{\Delta y}{\beta}}.$$

First, $\mathcal{T}$ chooses a random number $y' \in \mathbb{Z}_p$ as the new value of the master secret key $y$. The incremental is set as $\Delta y = y' - y$. Then, it calculates the new public key $Y' = e(g,g)^{y'}$ and the group element $g^{\frac{\Delta y}{\beta}}$. Finally, $\mathcal{T}$ broadcasts $Y'$ to all the sensor nodes and $g^{\frac{\Delta y}{\beta}}$ to all the users excluding the one to be revoked. Upon receiving the master secret key update message, each sensor node simply replaces the public key $Y$ with $Y'$.[3] The master key for the next stage will be encrypted under the new public key. Each user updates his secret key as follows: $g^{\frac{y-\theta}{\beta}} g^{\frac{\Delta}{\beta}} = g^{\frac{y'-\theta}{\beta}}$. The master secret key $y$ is thus updated as $y'$. In this user revocation scheme, one challenging issue is to selectively broadcast $g^{\frac{\Delta y}{\beta}}$ such that all but the leaving users are able to receive it. Fortunately, there are plenty of off-the-shelf selectively broadcast schemes available for different application scenarios. In [31], it is able to broadcast any $n-r$ out of $n$ users with ciphertext size of $O(r)$ and private key size of $O(log^2 n)$ by, which is further reduced to $O(logn)$ by [36]. This scheme is suitable for application scenarios where the number of revoked users each time is small. In [28], Boneh et al. proposed a scheme which is able to broadcast to arbitrary subset of users with constant ciphertext size (only two group elements). This scheme is extremely suitable for bandwidth-critical applications. One drawback of this basic scheme of [28] is that the public key size is of $O(n)$. To balance the size between the public key and the ciphertext, a revised scheme is presented in which both the ciphertext and the public key are of size $O(\sqrt{n})$. In [29], Cheung et al. proposed a collusion-resistant broadcast encryption scheme based on flat table scheme [37] and attribute-based encrytpion [34]. Both the ciphertext and the user secret key are of size $O(logn)$. In [35], Yu et al. further improved [29] by supporting receiver anonymity. Also, [29] and [35] are suitable for scenarios in which the system wants to revoke users of some common attributes, or the number of revoked users each time is small. In our proposed scheme, we do not designate any particular selective broadcast scheme for user secret key update. The system designer can pick an appropriate broadcast scheme from the above candidates according to the requirement of the actual system.

### 3.4.6 Further Enhancement

In the above user revocation scheme, $g^{\frac{\Delta y}{\beta}}$ is an update message common to all nonrevoked users, which opens the door for a nonrevoked user to collude with revoked users and help them decrypt the data. In [38], Boldyreva et al. proposed a user revocation scheme for IBE and KP-ABE in which user collusion attacks are well addressed. The proposed scheme is built on top of the construction of Fuzzy IBE [39] and the binary tree data structure. More specifically, it introduces a time attribute and uses it in the encryption of

3. Note that the new public key $Y'$ should be signed by the authority so that the sensor nodes can verify its authenticity. The signature scheme can be off-the-shelf algorithms such as ECDSA. For brevity, we do not explicitly include the signature in our proposed scheme.

each message. The root node of each user's access tree is an AND gate with one child being the time attribute and the other being the root node of an ordinary access structure. When a user is to be revoked, the system administrator generates key updates on the time attribute using the binary tree, each leaf node of which is associated to one user. Since new messages will be encrypted with the updated time attribute, users didn't receive the key updates will not be able to decrypt. In this scheme, the complexity of encryption and decryption is comparable to that of current KP-ABE [33]. The complexity of user revocation in terms of message size and computation overhead is $O(rlog(\frac{n}{r}))$ when $1 < r \leq n/2$, or $O(n-r)$ when $n/2 < r < n$, where $r$ is the total number of revoked users and $n$ is the total number of users. It should be noted that, we can also use this revocable KP-ABE in our scheme for achieving fine-grained data access control. One significant advantage of using the revocable KP-ABE is its enhanced security against user collusion. However, the complexity of user revocation is linear to the number of revoked users which could raise concerns in large-scale systems when that number is approaching $n/2$. In our proposed user revocation solution, the system designer is free to choose a broadcast scheme. For example, he/she can use [28] which has the constant ciphertext size if communication overhead is of the most importance. However, security level is reduced in this solution. We treat the above issue as a trade-off between efficiency and security.

### 3.5 Discussions

#### 3.5.1 Change of Sensor Attributes

Conventionally, the set of attributes of each sensor node does not change throughout the node's lifetime, or we can make this assumption as it is enough for many application scenarios. Nevertheless, there are still some cases in which the attributes of sensor nodes would change. For example, in some dynamic environments such as battlefields, the location of a portion of sensor nodes might be adjusted frequently. In this case, it is desirable to change the location attributes for the involved sensor nodes while not affecting the others. To achieve this goal, we just need to load the involved sensor nodes with the new attribute lists as well as the corresponding public key components. This can be easily realized in our proposed scheme as long as the involved sensor nodes are convinced of the authenticity of the update, which can be realized easily by attaching the network controller's signature to the update.

#### 3.5.2 Support for Concealed Data Aggregation

In-network aggregation of data has been put forward as an important paradigm for wireless sensor networks which enables in-network consolidation of redundant data and thus saves energy [40]. In practical settings, it is often desired to provide in-network data aggregation while guaranteeing data confidentiality for privacy concerns. The concept of Concealed Data Aggregation (CDA)[41] was proposed to address this issue. With CDA, the intermediate nodes are able to aggregate data by performing the aggregation operations on incoming ciphertexts without knowing the data encryption keys nor the plaintext. To realize CDA, sensing nodes should encrypt data using certain encryption transformation, a.k.a. *privacy homomorphism (PH)*. A survey on existing PH schemes, including symmetric and asymmetric ones, can be found in [42]. We stress that our proposed scheme can seamlessly integrate existing symmetric PH schemes and thus realize CDA. To justify this, we take [43] as an example and show how that PH scheme is integrated with our proposed scheme. At a high level the PH scheme in [43] has the property as follows: Given two messages $m_1$ and $m_2$, and their respective encryption keys $k_1$ and $k_2$, if $c_1 = Enc(m_1, k_1)$ and $c_2 = Enc(m_2, k_2)$, then $c_1 + c_2 = Enc(m_1 + m_2, k_1 + k_2)$ and $Dec(c_1 + c_2, k_1 + k_2) = m_1 + m_2$. This property still holds for the case of more than two messages and can be used to compute statistical values, e.g., mean, variance, and standard deviation, of sensed data. Intuitively, we can integrate this PH scheme into our proposed scheme in the following way: First, let each sensing node encrypt the sensed data with its data encryption key $k_i$ and encrypt $k_i$ (actually its seed) under its attributes. This process is basically the same as that of our proposed scheme. Then, upon data query every sensing node sends both the ciphertext of the sensed data and that of $k_i$ to its upstream aggregating node. The intermediate aggregating nodes, after having collected all the downstream data, do the aggregation operations on the ciphertexts of data while keeping ciphertexts of the data encryption keys intact. Subsequently, they transmit the aggregated ciphertext of data along with the ciphertexts of data encryption keys to their respective upstream aggregating nodes. The above process is recursively executed until it reaches the user. The user, on receiving the ciphertexts, first recovers the data encryption keys if his access structure satisfies with the sets of attributes of all the sensing nodes. Then, he does the same aggregation operations over the recovered data encryption keys as all the aggregating nodes did to compose a "aggregated" data encryption key of the final data ciphertext and decrypt the aggregated value of data. The drawback of this intuitive solution is that the ciphertexts of the data encryption keys could be a heavy communication overhead. This is because the size of such a ciphertext grows linearly to the number of attributes of the sensing node and each intermediate aggregating node should forward these ciphertexts of all its downstream sensing nodes. To alleviate this overhead, we can enforce our access control strategy only on few designated upstream aggregating nodes. In this way, the downstream sensing nodes just need to encrypt sensed data with their data encryption keys. These designated upstream aggregating nodes fulfill our access control strategy by encrypting the "aggregated" data encryption keys under certain set of attributes. One issue underlying this method is that the aggregating nodes should distribute data encryption keys to all its downstream sensing nodes, which can be solved using existing key distribution methods [44].

## 4 Scheme Evaluation

This section evaluates FDAC in terms of its security and performance.

### 4.1 Security Analysis

We evaluate the security of our work by analyzing the its fulfillment of the security requirements described in Section 2.

**Fine-grained data access control.** To provide fine-grained data access control, the proposed scheme should provide a strategy that is able to define and enforce complex access policies for sensor data of various types or security levels. In FDAC, the access structure embedded in each user's secret key is able to represent complicated predicates such as disjunctive normal form (DNF), conjunctive normal form (CNF), and threshold gates. The combination of these predicates are able to represent sophisticated access structures. In fact, FDAC is able to support nonmonotonic (general) access structures if we define the *NOT* of each attribute as a separate attribute, which, in turn, will double the number of attributes in our system. To enforce our access control strategy, FDAC encrypts the master key of the key chain in each stage under a set of attributes. Without the master key, the adversary is not able to derive the data encryption keys due to the one-wayness of the key chain, which can be guaranteed by choosing a secure one-way hash function such as SHA-1. In our basic scheme, the master key is actually encrypted under the standard KP-ABE scheme [33] which is provably secure. Our advanced scheme, to achieve efficient user revocation, makes some enhancement to the standard KP-ABE when encrypting the master key. The enhanced KP-ABE is provably secure under the Decisional Bilinear Diffie-Hellman (DBDH) assumption. A formal security proof is available in Appendix. This turns out that the adversary is not able to decrypt the master key unless he owns the intended access structure. Therefore, FDAC is able to control the accessibility of sensor data so that only authorized users are able to access.

**Collusion resilience.** To compromise sensor data, the main task of the colluding users is to decrypt the master key of the target data if the one-wayness of the underlying one-way has function, e.g., SHA-1, is guaranteed. Since the master key is encrypted under our enhanced scheme, we have to prove that it is collusion-resistant. Intuitively, we can sketch the collusion-resistance of our enhanced scheme as follows: Recall that the master key is encrypted in the form of $Ke(g,g)^{ys}$. The user has to cancel $e(g,g)^{ys}$ to recover $K$. To compose $e(g,g)^{ys}$, the only way is to execute the following: $e(g^{\frac{y-r}{\beta}}, g^{\beta s}) = e(g,g)^{ys}/e(g,g)^{rs}$. To extract $e(g,g)^{ys}$, the user should compute $e(g,g)^{rs}$. Actually, for each user, $r$ is randomly and independently selected from $\mathbb{Z}_p$. The secret key from one unauthorized user does not give the other user any help in terms of computing $e(g,g)^{rs}$. Actually, in our security proof the security definition (cf. Appendix) implies collusion resistance. As our scheme is provably secure under this security definition, collusion resistance is also guaranteed.

**Sensor compromise resistance.** To meet this security requirement, we should achieve two security goals: 1) compromising the sensor node does not disclose the sensor data generated before the sensor is compromised, and 2) compromising one sensor node does not give the adversary any advantage to obtain data generated by other sensor nodes. We can show the fulfillment of our scheme with respect to these two security goals as follows: 1) In our scheme, each sensor node just keeps the current data encryption key in the memory, while erasing all the previously used keys. Because of the one-wayness of the key chain, the compromiser is not able to derive the previously keys from the current key. 2) It is easy to prove since each sensor node encrypts data independently.

**Backward secrecy.** As is described in the previous section, our advanced scheme is able to update the master key $y$ for legitimate users while excluding those to be revoked. Since the new sensor data will be encrypted under the latest master key, the revoked users are not able to decrypt. One problem in our scheme is, the user revocation instruction will not take effort until a new stage starts. Such a delay occurs because it would take one stage for each sensor node to encrypt the master key under the attributes. This delay may differ for different systems. For example, if a system defines 30 phases as a stage and each phase lasts 1 second, the delay will be at the most half a minute. Generally, if a system has a stage with less phases and each phase takes less time, e.g., each sensor node is assigned a smaller number of attributes or has a more powerful computational capability, the delay can be shorter. In practical applications, we leave this delay as a system parameter, and the system designer can adjust this parameter by changing the number of attributes assigned to each sensor node or using a different type of sensor nodes.

In addition to the security goals listed above, there are also some other security requirements such as data integrity and authenticity, which are desired by conventional WSN applications. In fact, security requirements such as message integrity can be easily supported in our scheme with minor modifications using existing off-the-shelf techniques. A challenging orthogonal issue would be data authenticity which requires sensing nodes to provide proofs of data authenticity to users. Some current work such as [45], [46] has provided salient solutions to this problem. As the main focus of this paper is fine-grained data access control, we do not explicitly address all those security problems.

## 4.2 Performance Evaluation

This section evaluates the performance of FDAC in terms of computation and communication overheads. In our scheme, sensor data are generated and encrypted by sensor nodes, and retrieved and decrypted by users. As sensor nodes are usually resource-constrained, they may not be able to execute expensive cryptographic primitives efficiently and thus become the bottleneck of the scheme. For this reason, our evaluation focuses on the performance of sensor nodes. In the following sections, we first discuss the numeric results in terms of computation and communication overheads for sensor nodes. Then, we present our implementation results on real sensors.

### 4.2.1 Numeric Result

In FDAC, each sensor node is responsible for the following operations in each stage: 1) generate the master key and encrypt it using our enhanced KP-ABE, 2) derive the data encryption keys based on the master key, and 3) encrypt sensor data using the data encryption keys. These operations are further distributed to each phase. Specifically, if we choose elliptic curves as the underlying bilinear group, in each phase the sensor node needs to execute at the most one scalar multiplication on elliptic curves, one one-way hash, and one symmetric key data encryption. Table 1 lists all these operations.

TABLE 1
Computational Overhead on Each Sensor

| | Scalar Mul | Hash | Data Encryption |
|---|---|---|---|
| Each Stage | $\|\mathcal{I}_i\|$ +1 | n | n |
| Each Phase | 1 or 0 | 1 | 1 |

TABLE 2
Communication Overload

| Data Retrieval | User Revocation |
|---|---|
| ($\|\mathcal{I}_i\|$ +1) $\mathbb{G}_1$ + $1\mathbb{G}_T$ + data payload | $1\mathbb{G}_T$ |

On each data retrieval request, the sensor node responds with $\langle E^v, \{D\}_{K_t} \rangle$ for sensor data of phase $t$ in stage $v$, where $E^v$ contains $|\mathcal{I}_i| + 1$ group elements on $\mathbb{G}_1$ and one on $\mathbb{G}_T$, and $\{D\}_{K_t}$ is the data payload. On user revocation, $\mathcal{T}$ only needs to broadcast one group element of $\mathbb{G}_T$ to all the sensor nodes. The communication overload for each sensor node is shown in Table 2 .

#### 4.2.2 Implementation

In our implementation, we choose Tmote Sky and iMote2 as the target platforms. We use SHA-1 as the one-way hash function and AES (supported by CC2420 Radio module of the motes) as the data encryption algorithm. Our implementation shows that it takes about $0.06\ ms$ for SHA-1 to execute one hash operation and $0.4\ ms$ for AES to encrypt 64 bytes data. Our implementation also shows that one scalar multiplication takes several seconds in the worst case. The scalar multiplication operation is thus the bottleneck of the sensor performance. To optimize this operation, one key issue is to find appropriate parameters for the elliptic curve.

In past years, many works have efficiently implemented Elliptic Curve Cryptography (ECC) on various sensor platforms. In these works, elliptic curves are ususally chosen according to standards such as NIST [47] and SECG [48], which enable most of the optimization methods. Although these elliptic curves serve perfectly for security schemes such as ECDH, ECDSA, etc., they are not pairing-friendly, i.e., they cannot be used as bilinear groups. In FDAC, however, the elliptic curve is required to be pairing-friendly. Most pairing-friendly elliptic curves studied by current work fall into two categories, namely Supersingular (SS) curves and MNT curves. In the case of SS curves, the two elliptic groups $\mathbb{G}_1$ and $\mathbb{G}_2$ (cf. Section 2.4.1) could be the same. For MNT curves, $\mathbb{G}_1$ and $\mathbb{G}_2$ are different. To choose an appropriate elliptic curve, several factors should be taken into account as follows: Let $l$ be the group size of the elliptic curve and $k$ be its embedding degree. To achieve a comparable security strength of 1,024-bit RSA, we should have $lk$ to be larger than 1,024, or at least close to 1,024. Given the security level, a higher $k$ results in a shorter group size. Therefore, choosing a high embedding degree for the elliptic curve in our scheme may result in not only a short ciphertext, but also an efficient scalar multiplications on each sensor. However, the embedding degree $k$ of the elliptic curve cannot be arbitrarily large. Choosing an appropriate embedding degree for the elliptic curve is actually another research area. According to the benchmark of Pairing-Based Crypto (PBC) library [49], elliptic curves with $l = 512$ and $k = 2$ results in the fastest bilinear pairing as compared to those with $k > 2$ for SS curves. The case is on the opposite for MNT curves. According to our testing of the PBC library on Linux platform with an Intel Pentium D 3.0 GHz CPU, SS curves with $l = 512$ and $k = 2$ (type $a$ curves in PBC) take about $6\ ms$ to execute a pairing, while MNT curves with $l = 159$ and $k = 6$ (type $d$ curves in PBC) take about $14\ ms$ (Actually, on the user side of FDAC decryption time is linear to the number of pairings). Although both results are acceptable to users, MNT curves imply a much shorter ciphertext as well as key size to sensor nodes. More importantly, scalar multiplication over 512-bit curves may not be supported by low-end sensor nodes such as Tmote Sky because it consumes too much RAM. For these reasons, we believe MNT curves with high embedding degrees are suitable for FDAC.

TABLE 3
Computation Overhead of One Phase on iMote2
(For MNT Curves)

| SHA-1 | AES | One Scalar Multiplication | | |
|---|---|---|---|---|
| | | 104MHz | 208MHz | 416MHz |
| 0.06ms | 0.4ms | 139ms | 69ms | 35ms |

TABLE 4
Computation Overhead of One Phase on iMote2
(For SS Curves)

| SHA-1 | AES | One Scalar Multiplication | | |
|---|---|---|---|---|
| | | 104MHz | 208MHz | 416MHz |
| 0.06ms | 0.4ms | 682ms | 341ms | 170ms |

TABLE 5
Computation Overhead of One Phase on Tmote Sky

| SHA-1 | AES | One Scalar Multiplication | |
|---|---|---|---|
| | | MNT Curves | SS Curves |
| 0.07ms | 0.4ms | 4.1s | N/A |

In our implementation, the elliptic curve is a MNT curve over $F_q$ with embedding degree of 6, where $q$ is a 159-bit prime number. The curve has the form $y^2 = x^3 + ax + b$. Our implementation is based on the TinyECC library [50] with curve-specific optimization disabled since the group size $q$ is not a Mersenne prime. Our result shows that iMote2 consumes about $35\ ms$ to execute a scalar multiplication when working at 416 MHz, $69\ ms$ at 208 MHz, and $139\ ms$ at 104 MHz. Tmote Sky consumes $4.1\ s$. For the 512-bit SS curve, iMote2 consumes $170\ ms$ at 416 MHz, $341\ ms$ at 208 MHz, and $682\ ms$ at 104 MHz. Tmote Sky does not have enough RAM to support 512-bit SS curve. Tables 3, 4, and 5 summarize the above implementation results.

We can estimate energy consumption of one phase using the equation $W = U \times I \times t$, where $U$ is the voltage, $I$ is the current draw, and $t$ is the execution time for one phase. According to the date sheet of each platform [51], [52], the current draw for TMote Sky is $1.8\ mA$ and the voltage can be chosen as $3\ v$. The iMote2 data sheet [51] just gives the current draw for running at 104 MHz with radio on, which is $66\ mA$. To be conservative, we use this value in our computation. The voltage of iMote2 can be chosen as $0.95\ v$. Based on the execution time we measured, the energy

consumption on the iMote2 platform (running at 104 MHz) for one phase is $8.74\,mJ$ in case of MNT curves and $42.79\,mJ$ in case of SS curves. The energy consumption on the TMote Sky platform for one phase is $24.68\,mJ$ (for MNT curves only).

## 5 Conclusion

In this paper, we analyzed a novel yet important issue of fine-grained data access control for distributed storage in WSNs. To address the problem, we proposed a scheme called FDAC in which each sensor node is assigned a set of attributes, and each user is assigned an access structure which designates the access capability of the user. The sensor data are encrypted under the attributes such that only the users with the intended access structure are able to decrypt. As the access structure is extremely expressive, we are able to control data access precisely, and thus achieve fine-grained access control. Moreover, FDAC is able to provide security assurance such as resilience to user colluding and sensor compromising attacks as well as user revocability. We also showed that FDAC is able to support attribute change of sensor nodes and seamlessly integrate existing PH schemes to realize concealed data aggregation. Our experiment shows that the system overload in FDAC is reasonable in practical scenarios, especially for high-end sensor nodes. An interesting future work of FDAC maybe on its efficient implementation on WSNs with low-end sensor nodes.

## Appendix

### Security Definition

Following the security definition of the standard KP-ABE [33], we define the security of our enhanced KP-ABE using the following game which reflects the notion of IND-CPA security. The security game can be described by the following steps:

**Init.** The adversary chooses the set of attributes, $\gamma$, that he wants to be challenged upon.

**Setup.** The challenger runs the setup algorithm, steps a) and b) in Section 3.4.1, and gives the public parameters PK to the adversary.

**Phase 1.** The adversary is allowed to issue queries for secret keys (SK) for access structures $\mathbb{A}_j$, where $\gamma \notin \mathbb{A}_j$ for all $j$.

**Challenge.** The adversary submits two equal length messages $M_0$ and $M_1$. The challenger flips a random fair coin $b$, and encrypts (cf. Section 3.4.2) $M_b$ with $\gamma$. The ciphertext is passed to the adversary.

**Phase 2.** Phase 1 is repeated.

**Guess.** The adversary outputs a guess $b'$ of $b$.

The advantage of an adversary $A$ in this game is defined as $Pr[b' = b] - \frac{1}{2}$. This model can be extended to handle chosen-ciphertext attacks by allowing for decryption queries in Phase 1 and Phase 2.

### The Decisional Bilinear Diffie-Hellman (DBDH) Assumption

Our security proof is based on the DBDH assumption which can be summarized as follows:

Let $a, b, c, z \in \mathbb{Z}_p$ be chosen at random and $g$ be a generator of $\mathbb{G}_1$. The DBDH assumption is that no probabilistic polynomial-time algorithm $\mathcal{B}$ can distinguish the tuple $(A = g^a, B = g^b, C = g^c, e(g,g)^{abc})$ from the tuple $(A = g^a, B = g^b, C = g^c, e(g,g)^z)$ with more than a negligible advantage. The advantage of $\mathcal{B}$ is

$$|Pr[B(A, B, C, e(g,g)^{abc}) = 0] - Pr[B(A, B, C, e(g,g)^z) = 0]|,$$

where the probability is taken over the random choice of the generator $g$, the random choice of $a, b, c, z$ in $\mathbb{Z}_p$, and random bits consumed by $\mathcal{B}$.

### Proof of Security

**Theorem 5.1.** *If a polynomial-time adversary $\mathcal{A}$ can break our scheme in the above security game with probability $\varepsilon$, then we can build a simulator $\mathcal{B}$ that is able to solve the DBDH problem with probability $\frac{\varepsilon}{2}$.*

**Proof.** In the DBDH game, the challenger chooses random numbers $a, b, c$ from $\mathbb{Z}_p$ and flips a fair coin $\mu$. If $\mu = 0$, set $z = abc$; If $\mu = 1$, set $z$ as a random value in $\mathbb{Z}_p$. $\mathcal{B}$ is given $(A, B, C, Z)$=$(g^a, g^b, g^c, e(g,g)^z)$ and asked to output $\mu$. To answer this challenge, $\mathcal{B}$ then simulates Game 1 as follows:

**Init.** $\mathcal{B}$ runs $\mathcal{A}$. $\mathcal{A}$ chooses the set of attributes $\gamma$ it wants to be challenged upon.

**Setup.** $\mathcal{B}$ creates public parameters as follows: First, set $Y = e(A, B) = e(g,g)^{ab}$. Then, for each attribute $i \in \gamma$, generate $T_i$ by the following steps:

- choose a random number $t_i \in \mathbb{Z}_p$.
- if $i \in \gamma$, sets $T_i = g^{t_i}$; otherwise, set $T_i = g^{bt_i} = B^{t_i}$.

Finally, output $PK$ as in the real scheme.

**Phase I.** $\mathcal{A}$ submits a query for secret key of access structure $T$, where $\gamma \not\models T$. To generate secret key components for attributes attached to $T$, $\mathcal{B}$ defines a recursive function $PolyDef(x)$ and runs it over the root node $r$ of $T$. For each node $x$ in $T$, use $k_x$ and $p_x$ to represent the node's threshold value and the number of its satisfied children, respectively (the satisfied child is a child node of $x$ that returns true over $\gamma$). We use $x_{pa}$ and $idx(x)$ to denote node $x$'s parent node and the unique index of node $x$ assigned by $x_{pa}$, respectively.

`PolyDef(x)`: It is defined by the following steps:

- Define the polynomial $q_x$ for node $x$ as follows:
  - If $x$ is not $r$, set $q_x(0) = q_{x_{pa}}(idx(x))$; otherwise, set $q_x(0) = ab + br'$, $r'$ is randomly chosen from $\mathbb{Z}_p$.
  - Select $d$ $(= k_x - 1)$ children of $x$. For each selected child $i$, choose a random number $r'_i$ from $\mathbb{Z}_p$ and let $q_x(idx(i)) = br'_i$. This completes the construction of polynomial $q_x$. Note that, if $p_x \leq d$, the set of selected children should include all the $p_x$ satisfied ones; otherwise, all the $d$ selected children should be satisfied ones. We denote the set of these selected children of $x$ plus $x$ itself by $X_s$.
- For each remaining child $j$ (not selected by the above step), calculate

$$q_x(j) = \sum_{i \in X_s} q_x(idx(i)) \Delta_{i,S_x}(j).$$

- For each child $i$ of $x$, run $PolyDef(i)$.

When $PolyDef(r)$ terminates, $\mathcal{B}$ completes the construction of the polynomials for all the nodes in $T$. In particular, $p_r(0) = ab + br'$. Note that, in our construction of polynomials, for each node $x$, the polynomial values have the following properties:

1. If $q_x(0)$ has the form of $R_x b$, then for each of its children $i$, $q_i(0)$ $(= q_x(idx(i)))$ has the form of $R_i b$.
2. If $q_x(0)$ has the form of $C_x ab + R_x b$, then for each of its children $i$, 1) if $i \in X_s$ (selected), $q_i(0)$ has the form of $R_i b$; otherwise, 2) $q_i(0)$ has the form of $C_i ab + R_i b$.
3. In properties 1 and 2, $C_x, R_x, C_i$, and $R_i$ are functions of Lagrange coefficients and random numbers (i.e., $r'_j$'s), and independent of $a$ and $b$.

From these properties, we may categorize a leaf nodes $x$ into one of the following three types:

1. Type A: $x \in \gamma$, i.e., $x$ is a satisfied node. $q_x(0)$ has the form of $R_x b$.
2. Type B: $x \notin \gamma$ but one of $x$'s ancestors (including $x$ itself) is selected by its parent. $q_x(0)$ has the form of $R_x b$.
3. Type C: all the other leaf nodes, $q_x(0)$ has the form of $C_x ab + R_x b$.

Therefore, the secret key component corresponding to each leaf node $x$ of $T$ is given as follows:

$$D_x = \begin{cases} g^{\frac{R_x b}{t_x}} = B^{\frac{R_x}{t_x}}, & x \text{ in Type A.} \\ g^{\frac{R_x b}{t_x b}} = g^{\frac{R_x}{t_x}}, & x \text{ in Type B.} \\ g^{\frac{C_x ab + R_x b}{t_x b}} = A^{\frac{C_x}{t_x}} g^{\frac{R_x}{t_x}}, & x \text{ in Type C.} \end{cases}$$

The secret key component $g^{\frac{y-\theta}{\beta}}$ of $SK$ is output as follows:

$$g^{\frac{y-\theta}{\beta}} = g^{\frac{ab - q_r(0)}{\beta}} = g^{\frac{-br'}{\beta}} = B^{\frac{-r'}{\theta}},$$

where $\theta$ and $r'$ are chosen by $\mathcal{B}$ himself. All the other components are generated as in the real scheme. Therefore, $\mathcal{B}$ is able to construct a secret key of $T$ which has the same distribution as that in the original scheme. The adversary $\mathcal{A}$ can repeat this step for polynomial times.

**Challenge.** The adversary $\mathcal{A}$ submits two equal length challenge messages $m_0$ and $m_1$ to $\mathcal{B}$. $\mathcal{B}$ flips a fair binary coin $v$ and picks out $m_v$. The ciphertext of $m_v$ is output as: $E = (\gamma, m_v Z, \{E_i = C^{t_i}\}_{i \in \gamma}, C^{\beta})$. Note that if $\mu = 0$ it is easy to show that the ciphertext is a valid random encryption of message $m_v$. Otherwise, if $\mu = 1$, then $Z = e(g,g)^z$ and $m_v Z = m_v e(g,g)^z$. Since $z$ is random, $m_v Z$ is just a random element of $\mathbb{G}_T$ from the adversary's view and contains no information about $m_v$.

**Phase II.** The simulator acts exactly as it did in Phase I.

**Guess.** The adversary $\mathcal{A}$ submits a guess $v'$ of $v$. If $v' = v$, $\mathcal{B}$ outputs $\mu' = 0$, indicating that the given DBDH-tuple is a valid one. Otherwise it outputs $\mu' = 1$, indicating that the given DBDH-tuple is just a random quadruple. In the case of $\mu = 1$, the ciphertext $E$ contains no information about $m_v$. Therefore, $v'$ is just a random guess of $v$, and thus $\mu'$ is just a random guess of $\mu$. Thus, we have $Pr[\mu' = \mu | \mu = 1] = \frac{1}{2}$. If $\mu = 0$, the ciphertext $E$ is a valid encryption of $m_v$. Since by definition $\mathcal{A}$ has the advantage of $\varepsilon$ to output a correct guess, i.e., $v' = v$, $\mathcal{B}$ outputs $\mu' = 0$ with the probability of $\frac{1}{2} + \varepsilon$, i.e., $Pr[\mu' = \mu | \mu = 0] = \frac{1}{2} + \varepsilon$. Therefore, the overall advantage of $\mathcal{B}$ in the DBDH game is $\frac{1}{2} Pr[\mu' = \mu | \mu = 0] + \frac{1}{2} Pr[\mu' = \mu | \mu = 1] - \frac{1}{2} = \frac{1}{2}(\frac{1}{2} + \varepsilon) + \frac{1}{2}\frac{1}{2} - \frac{1}{2} = \frac{1}{2}\varepsilon$. □

## ACKNOWLEDGMENTS

This work was supported in part by the US National Science Foundation under grants CNS-0716306, CNS-0746977, CNS-0831628, and CNS-0831963. The preliminary version of this paper appears in [1].

## REFERENCES

[1] S. Yu, K. Ren, and W. Lou, "FDAC: Toward Fine-Grained Distributed Data Access Control in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* Apr. 2009.

[2] I.F. Akyildiz and I.H. Kasimoglu, "Wireless Sensor and Actor Networks: Research Challenges," *Ad Hoc Networks,* vol. 2, no. 4, pp. 351-367, Oct. 2004.

[3] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Comm. Magazine,* vol. 40, no. 8, pp. 102-116, Aug. 2002.

[4] C.-Y. Chong and S.P. Kumar, "Sensor Networks: Evolution, Opportunities, and Challenges," *Proc. IEEE,* vol. 91, no. 8, pp. 1247-1256, Aug. 2003.

[5] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat Monitoring: Application Driver for Wireless Communications Technology," *Proc. ACM SIGCOMM Workshop Data Comm. in Latin Am. and the Caribbean,* Apr. 2001.

[6] D. Estrin, D. Culler, and K. Pister, "Connecting the Physical World with Pervasive Networks," *IEEE Pervasive Computing,* vol. 1, no. 1, pp. 59-69, Jan.-Mar. 2002.

[7] B. Thuraisingham, "Secure Sensor Information Management and Mining," *IEEE Signal Processing Magazine,* vol. 21, no. 3, pp. 14-19, May 2004.

[8] A. Banerjee, A. Mitra, W. Najjar, D. Zeinalipour-Yazti, V. Kalogeraki, and D. Gunopulos, "RISE Co-S: High Performance Sensor Storage and Co-Processing Architecture," *Proc. Second Ann. IEEE Comm. Soc. Conf. Sensor and Ad Hoc Comm. and Networks (SECON '05),* 2005.

[9] A. Mitra, A. Banerjee, W. Najjar, D. Zeinalipour-Yazti, V. Kalogeraki, and D. Gunopulos, "High-Performance Low Power Sensor Platforms Featuring Gigabyte Scale Storage," *Proc. Ann IEEE/ACM Int'l Conf. Mobile and Ubiquitous Systems: Networking and Service (MobiQuitous '05),* 2005.

[10] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy, "Capsule: An Energy-Optimized Object Storage System for Memory-Constrained Sensor Devices," *Proc. ACM Conf. Embedded Networked Sensor Systems (Sensys),* Nov. 2006.

[11] D. Zeinalipour-Yazti, V. Kalogeraki, D. Gunopulos, A. Mitra, A. Banerjee, and W. Najjar, "Towards In-Situ Data Storage in Sensor Databases," *Proc. 10th Panhellenic Conf. Informatics (PCI '05),* pp. 36-46, 2005.

[12] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "GHT: A Geographic Hash Table for Data-Centric Storage," *Proc. Int'l Workshop Wireless Sensor Networks and Applications (WSNA '02),* Sept. 2002.

[13] J. Newsome and D. Song, "GEM:Graph Embedding for Routing and Data-Centric Storage in Sensor Networks without Geographic Information," *Proc. ACM Conf. Embedded Networked Sensor Systems (SenSys '03),* Nov. 2003.

[14] J. Girao, D. Westhoff, E. Mykletun, and T. Araki, "Tinypeds: Tiny Persistent Encrypted Data Storage in Asynchronous Wireless Sensor Networks," *Ad Hoc Networks,* vol. 5, no. 7, pp. 1073-1089, 2007.

[15] R.D. Pietro, L.V. Mancini, C. Soriente, A. Spognardi, and G. Tsudik, "Catch Me (If You Can): Data Survival in Unattended Sensor Networks," *Proc. Sixth Ann. IEEE Int'l Conf. Pervasive Computing and Comm. (PerCom '08),* Mar. 2008.

[16] Q. Wang, K. Ren, W. Lou, and Y. Zhang, "Dependable and Secure Sensor Data Storage with Dynamic Integrity Assurance," *Proc. IEEE INFOCOM,* Apr. 2009.

[17] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "LHAP: A Lightweight Hop-by-Hop Authentication Protocol for Ad-Hoc Networks," *Proc. IEEE Int'l Conf. Distributed Computing Systems Workshops (ICDCSW '03),* May 2003.

[18] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical En-Route Filtering of Injected False Data in Sensor Networks," *Proc. IEEE INFOCOM,* Mar. 2004.

[19] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," *Proc. IEEE Symp. Security and Privacy (S & P '03),* May 2003.

[20] M. Shao, S. Zhu, W. Zhang, and G. Cao, "pDCS: Security and Privacy Support for Data-Centric Sensor Networks," *IEEE Trans. Mobile Computing,* vol. 8, no. 8, pp. 1023-1038, Aug. 2009.

[21] R.D. Pietro, L. Mancini, C. Soriente, A. Spognardi, and G. Tsudik, "Playing Hide-and-Seek with a Focused Mobile Adversary in Unattended Wireless Sensor Networks," *Ad Hoc Networks,* vol. 7, no. 8, pp. 1463-1475, 2009.

[22] W. Zhang, H. Song, S. Zhu, and G. Cao, "Least Privilege and Privilege Deprivation: Towards Tolerating Mobile Sink Compromises in Wireless Sensor Networks," *ACM Trans. Sensor Networks,* vol. 4, no. 4, Nov. 2008.

[23] H. Wang and Q. Li, "Distributed User Access Control in Sensor Networks," *Proc. IEEE Int'l Conf. Distributed Computing Systems (DCOSS),* June 2006.

[24] H. Wang, B. Sheng, C.C. Tan, and Q. Li, "Comparing Symmetric-Key and Public-Key Based Schemes in Sensor Networks: A Case Study for User Access Control," *Proc. IEEE Int'l Symp. Distributed Computing Systems (ICDCS),* June 2008.

[25] C.C. Tan, H. Wang, S. Zhong, and Q. Li, "Body Sensor Network Security: An Identity-Based Cryptography Approach," *Proc. ACM Conf. Wireless Network Security (WiSec),* Mar.-Apr. 2008.

[26] N. Subramanian, C. Yang, and W. Zhang, "Securing Distributed Data Storage and Retrieval in Sensor Networks," *Pervasive and Mobile Computing,* (Special Issue for PerCom 2007), vol. 3, no. 6, pp. 659-676, Nov. 2007.

[27] M. Albrecht, C. Gentry, S. Halevi, and J. Katz, "Attacking Cryptographic Schemes Based on 'Perturbation Polynomials'," Cryptology ePrint Archive Report 2009/098, 2009.

[28] D. Boneh, C. Gentry, and B. Waters, "Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys," *Proc. Advances in Crytology (CRYPTO '05),* 2005.

[29] L. Cheung, J. Cooley, R. Khazan, and C. Newport, "Collusion-Resistant Group Key Management Using Attribute-Based Encryption," Cryptology ePrint Archive Report 2007/161, 2007.

[30] A. Fiat and M. Noar, "Broadcast Encryption," *Proc. Advances in Crytology (CRYPTO '93),* 1993.

[31] D. Naor, M. Naor, and J. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers," *Proc. Advances in Crytology (CRYPTO '01),* 2001.

[32] B. Karp and H. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proc. ACM Mobicom,* Aug. 2000.

[33] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data," *Proc. ACM Conf. Computer and Comm. Security (CCS),* 2006.

[34] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," *Proc. IEEE Symp. Security and Privacy (S & P),* 2007.

[35] S. Yu, K. Ren, and W. Lou, "Attribute-Based On-Demand Multicast Group Setup with Membership Anonymity," *Proc. Int'l Conf. Security and Privacy in Comm. Networks (SecureComm '08),* Sept. 2008.

[36] M. Goodrich, J. Sun, and R. Tamassia, "Efficient Tree-Based Revocation in Groups of Low-State Devices," *Proc. Advances in Cryptology (CRYPTO),* 2004.

[37] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B. Plattner, "The VersaKey Framework: Versatile Group Key Management," *IEEE J. Selected Areas in Comm.,* vol. 17, no. 9, pp. 1614-1631, Sept. 1999.

[38] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-Based Encryption with Efficient Revocation," *Proc. ACM Conf. Computer and Comm. Security (CCS),* Oct. 2008.

[39] A. Sahai and B. Waters, "Fuzzy Identity-Based Encryption," *Proc. Ann. Int'l Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT),* May 2005.

[40] B. Krishnamachari, D. Estrin, and S. Wicker, "The Impact of Data Aggregation in Wireless Sensor Networks," *Proc. Int'l Workshop Distributed Event-Based Systems,* July 2002.

[41] J. Girao, D. Westhoff, and M. Schneider, "CDA: Concealed Data Aggregation for Reverse Multicast Traffic in Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Comm. (ICC '05),* May 2005.

[42] S. Peter, D. Westhoff, and C. Castelluccia, "A Survey on the Encryption of Convergecast-Traffic with In-Network Processing," *IEEE Trans. Dependable and Secure Computing,* vol. 7, no. 1, pp. 20-34, Jan.-Mar. 2010.

[43] C. Castelluccia, E. Mykletun, and G. Tsudik, "Efficient Aggregation of Encrypted Data in Wireless Sensor Networks," *Proc. IEEE/ACM Int'l Conf. Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous '05),* July 2005.

[44] S.A. Camtepe and B. Yener, "Key Distribution Mechanisms for Wireless Sensor Networks: A Survey," Technical Report TR-05-07, Rensselaer Polytechnic Inst., Computer Science Dept., 2005.

[45] K. Ren, W. Lou, and Y. Zhang, "LEDS: Providing Location-Aware End-to-End Data Security in Wireless Sensor Networks," *Proc. IEEE INFOCOM,* pp. 1-12, Apr. 2006.

[46] K. Ren, W. Lou, and Y. Zhang, "Multi-User Broadcast Authentication in Wireless Sensor Networks," *Proc. IEEE Ann. IEEE Comm. Soc. Conf. Sensor and Ad Hoc Comm. and Networks (SECON '07),* Jun. 2007.

[47] Nat'l Inst. of Standards and Technology, "Recommended Elliptic Curves for Federal Government Use," Aug. 1999.

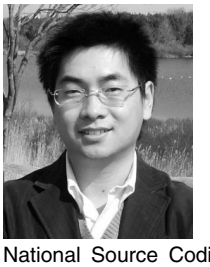
[48] Certicom Research, "Standards for Efficient Cryptography C SEC 2: Recommended Elliptic Curve Domain Parameters," http://www.secg.org/collateral/sec2_final.pdf, Sept. 2000.

[49] PBC Library, http://crypto.stanford.edu/pbc/times.html, 2010.

[50] TinyECC Library, http://discovery.csc.ncsu.edu/software/TinyECC/index.html, 2010.

[51] Imote2: High-performance wireless sensor network node, http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Imote2_Datasheet.pdf, 2009.

[52] TelosB mote platform, http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/TelosB_Datasheet.pdf, 2009.



**Shucheng Yu** received the BE degree in computer science and engineering from Nanjing University of Post & Telecommunication, China, in 1999 and the MS degree in computer science and engineering from Tsinghua University, China, in 2004. He is a PhD candidate in the Electrical and Computer Engineering Department at Worcester Polytechnic Institute. In the past, he has also worked as a Research Engineer at Cadence Design Systems, Chinese National Source Coding Center and Bright Oceans Corporation. His research interests include network security and applied cryptography. His current research interests include security and privacy in cloud computing, attribute-based cryptography, and wireless sensor network security. He is a member of the IEEE.

**Kui Ren** received the BEng and MEng degrees from Zhejiang University in 1998 and 2001, respectively, and the PhD degree in electrical and computer engineering from Worcester Polytechnic Institute in 2007. He is an assistant professor in the Electrical and Computer Engineering Department at Illinois Institute of Technology. In the past, he has worked as a research assistant at Shanghai Institute of Microsystem and Information Technology, Chinese Academy of Sciences, at Institute for Infocomm Research, Singapore, and at Information and Communications University, South Korea. His research interests include network security & privacy and applied cryptography with current focus on security & privacy in cloud computing, lower-layer attack and defense mechanisms for wireless networks, and sensor network security. His research is sponsored by the US National Science Foundation. He is a member of the IEEE and the ACM.

**Wenjing Lou** received the BE and ME degrees in computer science and engineering from Xi'an Jiaotong University in China, the MASc degree in computer communications from Nanyang Technological University in Singapore, and the PhD degree in electrical and computer engineering from the University of Florida. From December 1997 to July 1999, she worked as a research engineer at Network Technology Research Center, Nanyang Technological University. She joined the Electrical and Computer Engineering Department at Worcester Polytechnic Institute as an assistant professor in 2003, where she is now an associate professor. Her current research interests are in the areas of ad hoc, sensor, and mesh networks, with emphases on network security and routing issues. She has been an editor for the *IEEE Transactions on Wireless Communications* since 2007. She was named Joseph Samuel Satin Distinguished fellow in 2006 by WPI. She is a recipient of the US National Science Foundation Faculty Early Career Development (CAREER) award in 2008. She is a senior member of the IEEE.