Hindawi
Security and Communication Networks
Volume 2018, Article ID 6140978, 10 pages
https://doi.org/10.1155/2018/6140978

Research Article

Secure and Efficient Access Control Scheme for Wireless Sensor Networks in the Cross-Domain Context of the IoT

Ming Luo, Yi Luo, Yuwei Wan, and Ze Wang

School of Software, Nanchang University, Nanchang 330000, China

Correspondence should be addressed to Ming Luo; lmhappy21@163.com

Received 10 September 2017; Revised 22 November 2017; Accepted 24 December 2017; Published 26 February 2018

Academic Editor: Angelos Antonopoulos

Copyright © 2018 Ming Luo et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Nowadays wireless sensor network (WSN) is increasingly being used in the Internet of Things (IoT) for data collection, and design of an access control scheme that allows an Internet user as part of IoT to access the WSN becomes a hot topic. A lot of access control schemes have been proposed for the WSNs in the context of the IoT. Nevertheless, almost all of these schemes assume that communication nodes in different network domains share common system parameters, which is not suitable for cross-domain IoT environment in practical situations. To solve this shortcoming, we propose a more secure and efficient access control scheme for wireless sensor networks in the cross-domain context of the Internet of Things, which allows an Internet user in a certificateless cryptography (CLC) environment to communicate with a sensor node in an identity-based cryptography (IBC) environment with different system parameters. Moreover, our proposed scheme achieves known session-specific temporary information security (KSSTIS) that most of access control schemes cannot satisfy. Performance analysis is given to show that our scheme is well suited for wireless sensor networks in the cross-domain context of the IoT.

1. Introduction

Wireless sensor network (WSN) is a distributed network which contains a large number of sensor nodes. We can collect the target data through the sensor nodes to obtain valuable information. Due to the flexibility and convenience of data capture, WSN has been integrated into the IoT. The integration of WSN applications and low-power sensing nodes with the Internet may be accomplished with various approaches and strategies [1], and the popular integration solutions include cloud-based integration approaches [2, 3], front-end proxy integration approaches [4], architecture frameworks [5], and the integration via standard Internet communication protocols [6, 7]. In the cloud-based integration solution, some important security requirements including privacy, trust, and anonymity cannot be addressed. This approach also does not support the secure integration with data sources from other sensing devices or heterogeneous WSN domains. For the front-end proxy integration solution, the wireless sensor nodes communicate with the Internet hosts through a proxy server; thus this integration approach does not support direct communications between WSN nodes and Internet hosts, and the shortcoming of this approach is that the proxy server is vulnerable to cyberattacks and may become the bottleneck. In the integration solution via standard Internet communication protocols, most of approaches employ specialized middleware layers instead of supporting generic Internet communication mechanisms that can implement heterogeneous applications. However, these proposed solutions developed in the context of the architecture frameworks currently do not support Internet communications in WSN environments. For the integration via standard Internet communication protocols, a large number of access control schemes using public key infrastructure (PKI) are proposed. PKI, however, has a serious problem of certificate management. Subsequently, a series of access control schemes using identity-based cryptography (IBC) or certificateless cryptography (CLC) are designed, and even a new idea of integrating IBC with CLC into an access control scheme is introduced. In particular some access control schemes using heterogeneous signcryption schemes are generated, in which an Internet sender as part of IoT belongs to the CLC environment and a wireless sensor receiver is in the IBC environment. However, almost all of these access

control schemes assume that communication nodes share common system parameters in different network domains, which are not suitable for cross-domain IoT environment in practical situations. Moreover, we find that most of these schemes cannot satisfy known session-specific temporary information security (KSSTIS, which means that the attacker cannot obtain the plaintext message when the ephemeral key and the access request message are leaked). Thus, it is necessary to design a more secure and efficient access control scheme and make it more suitable for wireless sensor networks in the cross-domain context of the IoT.

1.1. Related Work. Zhou et al. proposed an access control scheme for WSNs using elliptic curve (EC) cryptography [8], which is more efficient than the PKI-based schemes. However, to authenticate a sensor node, the scheme of Zhou et al. needed high computational and communicational costs. Next, Huang [9] proposed an efficient access control protocol (EACP) based on the EC, which is quite adequate for low-powered sensor nodes. Consequently, Kim and Lee [10] pointed out that EACP scheme is susceptible to a message replay attack, and they proposed an enhanced access control protocol (ENCP). However, Lee et al. [11] showed that ENCP is subjected to a new node masquerade attack and message forgery attack, and then they proposed a practical access control protocol (PACP). In 2015, Chen et al. [12] claimed that the PACP is susceptible to the adversary attacks and needs huge key storage resources. Recently, Kumar et al. [13] proposed a more secure and efficient scheme for WSNs, which provides robust security and achieves the access control while taking care of the identity privacy. However, these schemes above cannot provide message confidentiality and unforgeability at the same time. In order to simultaneously authenticate the sensor node and protect the confidentiality of messages with a low cost, Yu et al. [14] and Ma et al. [15] proposed the access control schemes using signcryption approach (ACSC). Signcryption performs the signature and the encryption in one logical step. Compared with the signature-then-encryption method, signcryption has less cost. But these above ACSC schemes are based on the public key infrastructure (PKI). In PKI, the certificate authority (CA) generates a digital certificate for each user, which triggers the PKI's certificate management problem. In order to avoid this problem and reduce the burden on traditional PKI, identity-based public key cryptography (IBC) and certificateless public key cryptography (CLC) were proposed, where certificate used in PKI is not needed. Recently, many security mechanisms for WSNs using IBC [16, 17] or CLC [18, 19] have been generated. All the above schemes are homogeneous means that sender and receiver must belong to the same security domain (PKI or IBC or CLC environment). Heterogeneous signcryption allows the sender to send a message to the receiver in different security domain. Huang et al. [20] proposed a heterogeneous signcryption scheme that the sender is in the IBC environment and the receiver belongs to the PKI environment. In 2016, Li et al. [21] proposed a novel access control scheme (NACS) for sensor networks in the context of the IoT. The NACS uses heterogeneous signcryption (HSC) in which an Internet sender as part of IoT belongs to the CLC environment and a wireless sensor receiver is in the IBC environment, which conforms the characteristics of the WSNs in the context of the IoT.

1.2. Our Contribution. In this paper, we propose an access control scheme for WSNs in the cross-domain context of the IoT using heterogeneous signcryption. We define the generic model and security model of the cross-domain heterogeneous signcryption (CDHSC) and then propose a CDHSC scheme that proves to be safe under the Bilinear Inverse Diffie-Hellman Problem (BIDH) and Computational Diffie-Hellman Problem (CDHP) assumptions in the random oracle model. Compared with NACS scheme [21] through performance analysis, our scheme has the following merits: (1) our scheme allows an Internet user in a certificateless cryptography (CLC) environment to communicate a sensor node in an identity-based cryptography (IBC) environment with different system parameters so that it can be used for WSNs in the cross-domain context of the IoT; (2) our scheme has less computation cost (not including precomputation cost). For the Signcryption algorithm, our scheme has the same computation cost as the NACS scheme. But for the Unsigncryption algorithm, our scheme only needs three bilinear pairings computations, while the NACS scheme requires four. As we all know, bilinear pairing computation is the most expensive operation in a signcryption scheme from bilinear pairing; (3) our scheme satisfies the known session-specific temporary information security attribute.

1.3. Organization. The remainder of our paper is organized as follows. The preliminaries for network model, bilinear pairings, and difficult mathematical problems are given in the next section. The third section elaborates on the definition of the cross-domain heterogeneous signcryption (CDHSC), proposes a specific CDHSC scheme, and gives the security analysis of the proposed scheme. In the fourth section we propose a secure and efficient access control scheme for wireless sensor networks in the cross-domain context of the IoT and perform an efficiency analysis on it. In the last section, we make a summary.

2. Preliminaries

In this part, we give the basic network model of access control scheme, some prior knowledge of bilinear pairings, and difficult mathematical problems.

2.1. Network Model. In the network model of access control for wireless sensor networks in the cross-domain context of the IoT, there are five types of communication entities including Internet user, a trusted third party called key generation center (KGC) in the CLC environment, WSN node, the other trusted third party named private key generator (PKG) in the IBC environment, and a gateway used to connect the CLC domain with the IBC domain. PKG and KGC are used to complete the registration of WSN nodes and Internet users, respectively. The PKG calculates the public key and a private key for each WSN node. The KGC is responsible for producing a part of the private key

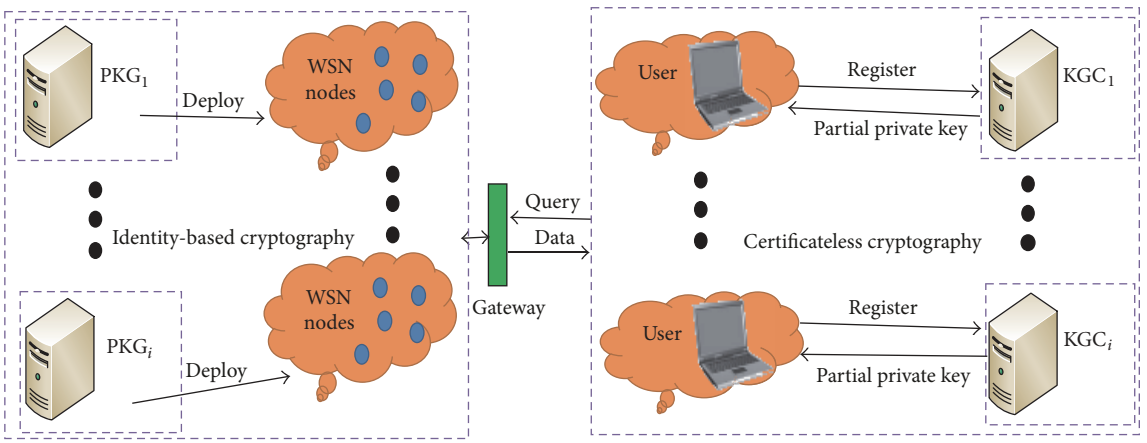


Figure 1: Network model.

of Internet users, and the other part of the private key is generated by the users themselves. In the network model, each PKG and KGC has different system parameters. In the KGC environment, when an Internet user wants to access the information collected by the sensor nodes from WSN, he needs to signcrypt and submit the query message to the gateway. The gateway belonging to this WSN will first authenticate the access request message from the Internet user. If the verification is passed, the gateway will forward the query message to the WSN. Then the WSN transmits the collected data to the Internet user with unsigncryption key. Otherwise, gateway refuses to provide the service.

In the network model of access control, the access request message generated by the Internet user should satisfy confidentiality, integrity, authentication, nonrepudiation, and known session-specific temporary information security (KSSTIS) simultaneously when it is transmitted to the gateway. Figure 1 shows the overview of the network model.

2.2. Bilinear Pairings. Let G_1 and G_2 be two cyclic groups, which have the same prime order q. G_1 is an additive group and G_2 is a multiplicative group. Let P be a generator of G_1. A bilinear pairing is a map $\hat{e} : G_1 \times G_1 \to G_2$ that meets the following properties:

(1) Bilinearity: $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$, for all $P, Q \in G_1$ and $a, b \in Z_q^*$.

(2) Nondegeneracy: there exists $P \in G_1$, such that $\hat{e}(P, P) \neq 1$, where 1 is the identity element of G_2.

(3) Computability: $\hat{e}(P, Q)$ can be successfully computed for all $P, Q \in G_1$.

The security of our scheme relies on the following two hard mathematical problems.

Definition 1. Bilinear Inverse Diffie-Hellman (BIDH) Problem is to compute $\hat{e}(P, P)^{ab^{-1}}$ given (P, aP, bP), where $a, b \in Z_q^*$.

Definition 2. Computational Diffie-Hellman (CDH) Problem is to compute abP given (P, aP, bP), where $a, b \in Z_q^*$.

3. Cross-Domain Heterogeneous Signcryption

For the cross-domain heterogeneous signcryption (CDHSC) which can be used in access control for WSNs in the context of the IoT, we first define the generic model and security model. Then we present the specific CDHSC scheme. Finally we show the correctness analysis and the security proof of the proposed scheme.

3.1. Generic Model. Our CDHSC scheme consists of nine algorithms as follows.

Setup. The trusted third party PKG and KGC execute this probabilistic algorithm to produce a series of system parameters. Firstly they input a security parameter k_i then output their master secret key s_i and corresponding system parameters. Different PKG and KGC use different k_i and output different s_i.

CL-PPUKE. The partial public key extraction algorithm is executed by the KGC in CLC environment, which takes as input a KGC's master secret key s_i and an Internet user's identity ID, and outputs the user's partial public key Q_{ID}.

CL-PPKE. The partial private key extraction algorithm is executed by the KGC in CLC environment, which takes as input a KGC's master secret key s_i and an Internet user's identity ID, and generates the user's partial private key D_{ID}.

CL-SVS. The secret value setup algorithm is performed by the Internet users in CLC environment. Taking as inputs a user's identity ID, the algorithm outputs the user's secret value x_{ID}.

CL-PKG. The main public key generation algorithm is executed by the users in the context of CLC, which takes as input the secret value x_{ID}, and outputs the user's main public key $\mathrm{PK}_{\mathrm{ID}}$.

IB-PKE. The public key extraction algorithm is performed by the PKG in IBC circumstance. Taking as inputs a PKG's master secret key s_j and the user's identity ID, the algorithm outputs the user's public key Q_{ID}.

IB-KE. The key extraction algorithm is performed by the PKG in IBC environment. Taking as inputs a PKG's master secret key s_j and the user's identity ID, the algorithm outputs the user's private key D_{ID}.

SC. The signcryption algorithm is performed by an Internet user under the circumstance of CLC. Taking as inputs the plaintext message m, sender's partial private key D_{ID_s}, sender's secret value x_{ID_s}, sender's identity ID_s, and the receiver's identity ID_r, the algorithm outputs the ciphertext σ.

USC. The unsigncryption algorithm is performed by the receiver in IBC environment. Taking as inputs the ciphertext σ, the sender's identity ID_s and the main public key $\mathrm{PK}_{\mathrm{ID}_s}$, sender's partial public key Q_{ID_s}, and the receiver's private key D_{ID_r} and identity ID_r, the algorithm outputs the plaintext m if σ is a valid ciphertext. Otherwise the output is the symbol $\perp$.

Note that the ciphertext σ should meet the need of the public verifiability with confidentiality. That is to say, ciphertext verification process of the unsigncryption algorithm can be performed by any verifier (generally the WSN gateway) without the knowledge of the plaintext message m.

3.2. Security Model. The standard security notion for a CDHSC scheme is confidentiality and unforgeability. In the following definitions, Definition 3 describes the confidentiality and the unforgeability is depicted in Definition 5.

Definition 3 (*confidentiality*). A CDHSC scheme is semantically secure against adaptive chosen ciphertext attacks property (IND-CDHSC-CCA2) if no probabilistic polynomially time adversary A has a nonnegligible advantage in the following Game 1.

Game 4.

Initial. The challenger C runs Setup algorithm with a parameter k, then he returns the system parameters to A.

Phase 1. A executes a polynomially bounded number of queries.

Partial Public Key Extraction (PPUKE) Queries. A chooses an identity ID and forwards it to C. Then C executes CL-PPUKE algorithm and forwards the corresponding partial public key Q_{ID} to A.

Public Key (PK) Queries. On a new public key query for identity ID, C executes CL-SVS and CL-PKG algorithm to compute the user's secret value x_{ID} and main public key $\mathrm{PK}_{\mathrm{ID}}$ then adds $(\mathrm{ID}, x_{\mathrm{ID}}, \mathrm{PK}_{\mathrm{ID}})$ to the list L_k. Finally, C returns $\mathrm{PK}_{\mathrm{ID}}$ to A.

Public Key Replacement (PKR) Queries. A can replace a main public key $\mathrm{PK}_{\mathrm{ID}}$ with a value selected by himself.

Corruption Query. On a corruption query, C checks the list L_k and returns the secret value x_{ID}.

Partial Private Key Extraction (PPKE) Queries. A chooses an identity ID and forwards it to C. Then C executes CL-PPKE algorithm and forwards the corresponding partial private key D_{ID} to A.

Public Key Extraction (PKE) Queries. When receiving an identity ID from A, C executes IB-PKE algorithm and forwards the corresponding public key Q_{ID} to A.

Key Extraction (KE) Queries. When receiving an identity ID from A, C executes IB-KE algorithm and forwards the corresponding private key D_{ID} to A.

Signcryption Queries. A submits a plaintext m, a sender's identity ID_i, and a receiver's identity ID_j. Firstly, C performs *Corruption* query and PPKE query with ID_i to obtain x_{ID_i} and D_{ID_i}, performs PKE query with ID_j to obtain $\mathrm{Q}_{\mathrm{ID}_j}$, and then executes SC algorithm to get the signcryption $\sigma = \mathrm{SC}(m, x_{\mathrm{ID}_i}, D_{\mathrm{ID}_i}, \mathrm{ID}_i, \mathrm{Q}_{\mathrm{ID}_j})$. If the sender's public key $\mathrm{PK}_{\mathrm{ID}_i}$ has been replaced, the sender's secret value x_{ID_i} is provided by A. Finally C returns ciphertext σ to A.

Unsigncryption Queries. A submits a signcryption σ, a sender's identity ID_i, and a receiver's identity ID_j. Firstly, C performs PPUKE query and PK query with ID_i to obtain $\mathrm{Q}_{\mathrm{ID}_i}$ and $\mathrm{PK}_{\mathrm{ID}_i}$, performs KE query with ID_j to obtain D_{ID_j}, and then executes USC algorithm to check the validity of ciphertext σ. If the ciphertext is valid, C sends plaintext $m = \mathrm{USC}(\sigma, \mathrm{PK}_{\mathrm{ID}_i}, \mathrm{Q}_{\mathrm{ID}_i}, D_{\mathrm{ID}_j}, \mathrm{ID}_j)$ to A; otherwise it outputs character $\perp$.

Challenge. After Phase 1, A outputs two plaintexts (m_0, m_1) which are of the same length, a sender's identity ID_s and a receiver's identity ID_r on which he wants to be challenged. Note that ID_r cannot be the identity that has been used for KE query in Phase 1. C randomly takes a bit of $\beta \in \{0, 1\}$ and calculates $\sigma^* = \mathrm{SC}(m_\beta, x_{\mathrm{ID}_s}, D_{\mathrm{ID}_s}, \mathrm{ID}_s, \mathrm{Q}_{\mathrm{ID}_r})$ and forwards it to A.

Phase 2. A can make a polynomially bounded number of queries just like in Phase 1, whereas, it cannot make a KE query on ID_r and cannot perform an *Unsigncryption* query on ciphertext σ^* under ID_s and ID_r to obtain the plaintext m unless the sender's public key $\mathrm{PK}_{\mathrm{ID}_s}$ is replaced after the challenge phase.

Guess. A outputs a bit of $\beta' \in \{0, 1\}$. If $\beta' = \beta$, he wins the game.

We define the advantage of A to be $\mathrm{Adv}(A) = |2\Pr[\beta' = \beta] - 1|$.

For unforgeability, there are two types of adversaries named A_{I} and A_{II} since the signcryption is generated in the CLC environment. Type I adversary A_{I} does not know the KGC's master key, but he is able to replace public keys of arbitrary identities with other public keys of his choice. In contrast, Type II adversary possesses the KGC's master secret key, while he cannot replace public key of any user during the game.

Definition 5 (*unforgeability*). A CDHSC scheme is existentially unforgeable against an adaptive chosen-message attacker (EUF-CDHSC-CMA) if no probabilistic polynomially time adversary $A_{i\ (i=\mathrm{I,II})}$ has a nonnegligible advantage in the following Game 2.

Game 6.

Initial. The challenger C runs the *Setup* algorithm defined in generic model and gives the resulting system parameters to the adversary A_i. For Type II adversary, C sends him the master secret keys of PKG and KGC in addition to the system parameters.

Probing. The challenger is probed by the adversary A_i who executes a polynomially bounded number of queries just like Phase 1 of the confidentiality game. Note that A_{II} does not need to perform PKR, PPKE, and KE queries.

Forge. The adversary A_i returns a ciphertext σ^*, a sender's identity ID_s, and a receiver's identity ID_r. Let the tuple $(m^*, \mathrm{ID}_s, \sigma^*)$ be the result of unsigncrypt algorithm under the private key corresponding to ID_r. A_i wins the game if the tuple $(m^*, \mathrm{ID}_s, \sigma^*)$ satisfies the following requirements:

(1) This ciphertext is a valid one, when the result of unsigncrypt algorithm is not character $\perp$ but m^*.

(2) A_i has never asked the secret value of the user with identity ID_s.

(3) A_i has never asked the *Signcryption* query on $(m^*, \mathrm{ID}_s, \mathrm{ID}_r)$.

3.3. CDHSC Scheme. In this section, we propose a CDHSC scheme based on bilinear pairings. We follow the generic model of a general CDHSC scheme that we presented in Section 3.1, and we add KSSTIS property to it. The scheme is described below.

Setup. Given a parameter k_0, the KGC chooses an additive group G_{1-0} and a multiplicative group G_{2-0} which have the same prime order q_0, a generator P_0 of G_{1-0}, a bilinear map $\widehat{e}_0 : G_{1-0} \times G_{1-0} \to G_{2-0}$, and three hash functions $H_{1-0} : \{0,1\}^* \to Z^*_{q_0}$, $H_{2-0} : G_{1-0} \times G_{2-0} \times \{0,1\}^* \to \{0,1\}^n$, and $H_{3-0} : \{0,1\}^n \times {G_{1-0}}^2 \times \{0,1\}^* \to Z^*_{q_0}$. Similarly, given a security parameter k_1, the PKG chooses an additive group G_{1-1} and a multiplicative group G_{2-1} which have the same prime order q_1, a generator P_1 of G_{1-1}, a bilinear map $\widehat{e}_1 : G_{1-1} \times G_{1-1} \to G_{2-1}$, and one hash function $H_{1-1} : \{0,1\}^* \to Z^*_{q_1}$. The KGC randomly chooses a master secret key $s_0 \in Z^*_{q_0}$ and calculates the master public key $P_{\mathrm{pub}_0} = s_0 P_0$. Then KGC outputs the system parameters $\{G_{1-0}, G_{2-0}, q_0, \widehat{e}, n, P_0, P_{\mathrm{pub}_0}, H_{1-0}, H_{2-0}, H_{3-0}\}$ and keeps s_0 secret. Similarly, the PKG outputs the system parameters $\{G_{1-1}, G_{2-1}, q_1, \widehat{e}_1, P_1, P_{\mathrm{pub}_1} = s_1 P_1, H_{1-1}\}$ and keeps s_1 secret.

CL-PPUKE. This algorithm accepts an identity ID_A of an Internet user and generates the partial public key $Q_A = (a + s_0)P_0$ for the user, where $a = H_{1-0}(\mathrm{ID}_A)$. Then the KGC runs the CL-PPKE algorithm.

CL-PPKE. After executing the CL-PPUKE algorithm, the KGC calculates the partial private key $D_A = (a + s_0)^{-1} P_0$ for the user. Finally, the KGC sends (Q_A, D_A) securely to the user.

CL-SVS. This algorithm accepts an identity ID_A of an Internet user and randomly chooses a secret value $x_A \in Z^*_{q_0}$ for the user. Then the user runs the CL-PKG algorithm.

CL-PKG. After executing the CL-SVS algorithm, the user calculates his public key $\mathrm{PK}_A = x_A Q_A$.

IB-PKE. This algorithm accepts an identity ID_B of a WSN node and generates the public key $Q_B = (b + s_1)P_1$ for the node, where $b = H_{1-1}(\mathrm{ID}_B)$. Then the PKG runs the IB-KE algorithm.

IB-KE. After executing the IB-PKE algorithm, the PKG calculates the private key $D_B = (b + s_1)^{-1} P_1$ for the node. Finally, the PKG sends (Q_B, D_B) securely to the WSN node.

SC. To signcrypt a message m using the partial private key D_A, secret value x_A, and the receiver's identity ID_B, a sender with identity ID_A performs the following steps:

(1) Selecting $r \in Z^*_{q_0}$ randomly.

(2) Calculating $U = (r + x_A)Q_B$ and $T = \widehat{e}(P_1, P_1)^{r + x_A}$.

(3) Calculating $Z = H_{2-0}(U, T, \mathrm{ID}_B)$.

(4) Calculating $C = m \oplus Z$.

(5) Calculating $t = H_{3-0}(C, U, \mathrm{PK}_A, \mathrm{ID}_A)$.

(6) Calculating $V = tD_A + x_A U$.

(7) Outputting the ciphertext $\sigma = (C, U, V)$.

USC. To unsigncrypt the ciphertext $\sigma = (C, U, V)$ using the private key D_B, sender's partial public key Q_A, and the main public key PK_A, the receiver with identity ID_B performs the following steps:

(1) Calculating $t = H_{3-0}(C, U, \mathrm{PK}_A, \mathrm{ID}_A)$.

(2) Checking if $\widehat{e}(V, Q_A) = \widehat{e}(P_0, P_0)^t \widehat{e}(\mathrm{PK}_A, U)$ holds. If the equation holds, the receiver executes the following step. Otherwise, he rejects this ciphertext and outputs the symbol $\perp$.

(3) Calculating $T = \widehat{e}(U, D_B)$.

(4) Calculating $Z = H_{2-0}(U, T, \mathrm{ID}_B)$.

(5) Recovering the message $m = C \oplus Z$.

Note that any user can verify the ciphertext $\sigma = (C, U, V)$ by computing $t = H_{3-0}(C, U, \mathrm{PK}_A, \mathrm{ID}_A)$ and verifying whether $\widehat{e}(V, Q_A) = \widehat{e}(P_0, P_0)^t \widehat{e}(\mathrm{PK}_A, U)$, where nothing about the plaintext message m will be lost. Thus, we can shift the computational cost of signcryption verification to the WSN gateway (he just needs to obtain the public parameters and the ciphertext σ) in the cross-domain context of the IoT.

3.4. Correctness. The consistency of the CDHSC scheme is easy to verify.

(1) In the signcryption verification stage,

$$\begin{aligned}\widehat{e}(V, Q_A) &= \widehat{e}(tD_A + x_A U, Q_A)\\ &= \widehat{e}(tD_A, Q_A)\,\widehat{e}(x_A U, Q_A)\\ &= \widehat{e}\left(t(a+s_0)^{-1} P_0, (a+s_0) P_0\right)\widehat{e}(U, x_A Q_A)\\ &= \widehat{e}(P_0, P_0)^{t}\,\widehat{e}(\mathrm{PK}_A, U).\end{aligned} \tag{1}$$

(2) In the signcryption decryption stage,

$$\begin{aligned}T &= \widehat{e}(U, D_B) = \widehat{e}\left((r + x_A)(b + s_1) P_1, (b + s_1)^{-1} P_1\right)\\ &= \widehat{e}(P_1, P_1)^{r + x_A}.\end{aligned} \tag{2}$$

3.5. Security Proof. In this section, we use some mathematical difficult problems to prove the confidentiality and unforgeability of the CDHSC scheme in the random oracle model. In addition, we demonstrate that our scheme satisfies the known session-specific temporary information security (KSSTIS). In our scheme, the generation algorithms of public key Q_B and private key D_B for the node ID_B in the IBC environment are the same as the generation algorithms of partial public key Q_A and partial private key D_A for the user ID_A in the CLC environment, so the KGC can act as the roles of KGC and PKG simultaneously in a small wireless sensor networks in a single domain context of the IoT. Moreover, in the following security proofs of our proposed scheme, for reasons of proof brevity, we assume that the KGC plays the roles of the KGC and PKG at the same time in a single domain.

(1) Confidentiality

Theorem 7. *Our CDHSC scheme is indistinguishable against any IND-CDHSC-CCA2 adversary A in the random oracle model assuming that the BIDH problem in G_2 is intractable.*

Proof. Let C be a BIDH problem attacker. Then A is an adversary who interacts with C following Game 1. C is given (P, aP, bP) as an input to the BIDH problem and aims to compute $\widehat{e}(P, P)^{ab^{-1}}$, where $a, b \in Z_q^*$ and $P \in G_1$.

Initial. C sets $P_{\mathrm{pub}} = sP$. The value s is the master key of the KGC, which is unknown to C, and C gives system parameters to A.

Phase 1. We show that C can use A to solve the BIDH problem. C needs to maintain three lists L_1, L_2, and L_3 that are initially empty and are used to keep track of answers to queries asked by A to oracles H_1, H_2, and H_3, respectively. What is more, C maintains two lists: L_k and L_p of the queries made by A to oracles PK and PPUKE (or PKE), respectively. Subsequently, C simulates the challenger and plays the game described in Definition 3 with the adversary A as follows.

Partial Public Key Extraction (PPUKE) or Public Key Extraction (PKE) Queries. Suppose that A makes at most q_p queries to this oracle. First, C chooses $\gamma \in \{1, 2, \ldots, q_p\}$ randomly. When A makes this query on ID_i ($i \in \{1, 2, \ldots, q_p\}$), if $i = \gamma$ (we let $\mathrm{ID}_i = \mathrm{ID}_\gamma$ at this point), C returns $Q_{\mathrm{ID}_\gamma} = bP$ and adds $(\mathrm{ID}_\gamma, Q_{\mathrm{ID}_\gamma} = bP, D_{\mathrm{ID}_\gamma} = b^{-1}P, \perp)$ to L_p (C cannot compute $D_{\mathrm{ID}_\gamma} = b^{-1}P$; he just considers D_{ID_γ} to be $b^{-1}P$). Otherwise C picks a random $w_i \in Z_q^*$, returns $Q_{\mathrm{ID}_i} = w_i P$, and adds $(\mathrm{ID}_i, Q_{\mathrm{ID}_i} = w_i P, D_{\mathrm{ID}_i} = w_i^{-1} P, w_i)$ to L_p.

H_1 Queries. When A makes this query on ID_i, C forwards e_i to A if L_1 has the entry (ID_i, e_i). If the list L_1 does not contain (ID_i, e_i), C randomly picks $e_i \in Z_q^*$, returns e_i, and adds (ID_i, e_i) to L_1.

H_2 Queries. When A makes H_2 query on $(U_i, T_i, \mathrm{ID}_i)$, if the list L_2 has the entry $(U_i, T_i, \mathrm{ID}_i, Z_i)$, C answers Z_i to A. Otherwise, C randomly picks $Z_i \in \{0, 1\}^n$ as the output and inserts $(U_i, T_i, \mathrm{ID}_i, Z_i)$ into the list L_2.

H_3 Queries. For H_3 query on $(C_i, U_i, \mathrm{PK}_{\mathrm{ID}_i}, \mathrm{ID}_i)$, if the list L_3 has the entry $(C_i, U_i, \mathrm{PK}_{\mathrm{ID}_i}, \mathrm{ID}_i, t_i)$, C answers t_i to A. Otherwise, C randomly picks $t_i \in Z_q^*$ as the output and inserts $(C_i, U_i, \mathrm{PK}_{\mathrm{ID}_i}, \mathrm{ID}_i, t_i)$ into the list L_3.

Partial Private Key Extraction (PPKE) or Key Extraction (KE) Queries. When A makes this query on ID_i, if $\mathrm{ID}_i = \mathrm{ID}_\gamma$, C aborts the simulation and returns $\perp$. Otherwise, the list L_p should have the entry $(\mathrm{ID}_i, Q_{\mathrm{ID}_i} = w_i P, D_{\mathrm{ID}_i} = w_i^{-1} P, w_i)$. C returns D_{ID_i} to A.

Public Key (PK) Queries. When A makes this query on ID_i, if the list L_k has the entry $(\mathrm{ID}_i, \mathrm{PK}_{\mathrm{ID}_i}, x_{\mathrm{ID}_i})$, then C answers $\mathrm{PK}_{\mathrm{ID}_i}$ to A. Otherwise, C selects $x_{\mathrm{ID}_i} \in Z_q^*$ randomly, computes $\mathrm{PK}_{\mathrm{ID}_i} = x_{\mathrm{ID}_i} Q_{\mathrm{ID}_i}$, and then returns $\mathrm{PK}_{\mathrm{ID}_i}$ and adds $(\mathrm{ID}_i, \mathrm{PK}_{\mathrm{ID}_i}, x_{\mathrm{ID}_i})$ to L_k.

Corruption Queries. We assume that C has made PK query on ID_i before this query. The list L_k should have the entry $(\mathrm{ID}_i, \mathrm{PK}_{\mathrm{ID}_i}, x_{\mathrm{ID}_i})$. C returns x_{ID_i} to A.

Public Key Replacement (PKR) Queries. A can replace the main public key $\mathrm{PK}_{\mathrm{ID}_i}$ of user ID_i with a value he selects. When A executes a *public key replacement* query with the entry $(\mathrm{ID}_i, \mathrm{PK}'_{\mathrm{ID}_i})$, C updates the list L_k with entry $(\mathrm{ID}_i, \mathrm{PK}'_{\mathrm{ID}_i}, \perp)$.

Signcryption Queries. When A asks for a *Signcryption* query on a message m with a sender's identity ID_i and a receiver's identity ID_j, if $\mathrm{ID}_i = \mathrm{ID}_\gamma$, C aborts the simulation and returns $\perp$. Otherwise, C first executes *Corruption* query and PPKE query with ID_i to obtain x_{ID_i} and D_{ID_i}, performs PPUKE query with ID_j to obtain Q_{ID_j}, and then executes $\mathrm{SC}(m, x_{\mathrm{ID}_i}, D_{\mathrm{ID}_i}, \mathrm{ID}_i, Q_{\mathrm{ID}_j})$ algorithm to return the signcryption $\sigma = (C, U, V)$. If the sender's public key $\mathrm{PK}_{\mathrm{ID}_i}$ has been replaced, the sender's secret value x_{ID_i} is provided by A. Finally C returns ciphertext σ back to A.

Unsigncryption Queries. When A asks for this query on a signcryption $\sigma = (C, U, V)$ with a sender's identity ID_i and

a receiver's identity ID_j, if $\mathrm{ID}_j = \mathrm{ID}_\gamma$, C aborts the simulation and returns $\perp$. Otherwise, C computes $t = H_{3-0}(C, U, \mathrm{PK}_{\mathrm{ID}_i}, \mathrm{ID}_i)$ and checks if $\hat{e}(V, Q_{\mathrm{ID}_i}) = \hat{e}(P_0, P_0)^t \hat{e}(\mathrm{PK}_{\mathrm{ID}_i}, U)$ holds. If not, C aborts the simulation and returns $\perp$. Otherwise, C executes the KE query to get D_{ID_j} and calculates $T = \hat{e}(U, D_{\mathrm{ID}_j})$. Then C executes H_2 query to obtain $Z = H_2(U, T, \mathrm{ID}_j)$ and finally calculates and returns $m = C \oplus Z$.

Challenge. A outputs two plaintexts (m_0, m_1) which have the same length and picks a sender's identity ID_s and a receiver's identity ID_r on which he wishes to be challenged. Note that C fails if A has asked a KE query on ID_r during the first stage. If $\mathrm{ID}_r \neq \mathrm{ID}_\gamma$, C aborts the simulation and returns $\perp$. Otherwise C selects a random number $\beta \in \{0, 1\}$ and generates the challenge ciphertext $\sigma^* = (C^*, U^*, V^*)$ as follows. At first, C chooses the value $V^* \in G_1$. Then he sets $U^* = aP$ and computes $Z^* = H_2(U^*, \mu^*, \mathrm{ID}_\gamma)$ and $C^* = m_\beta \oplus Z^*$, in which $\mu^* = \hat{e}(U^*, D_\gamma) = \hat{e}(aP, b^{-1}P) = \hat{e}(P, P)^{ab^{-1}}$ (μ^* is the candidate answer for the BIDH problem). Finally C forwards the ciphertext $\sigma^* = (C^*, U^*, V^*)$ to A.

Phase 2. A then performs a second series of queries, and C can handle these queries as in the first stage. Whereas, it cannot make a KE query on ID_r and cannot perform an *Unsigncryption* query on ciphertext σ^* under ID_s and ID_r to obtain the plaintext unless the sender's public key $\mathrm{PK}_{\mathrm{ID}_s}$ is replaced after the *challenge* phase.

Guess. A produces a bit of β'. If $\beta' = \beta$, C then answers 1 as the result to the BIDH problem since he has generated a valid signcrypted message of m_β using the knowledge of μ^*. Otherwise, C answers 0.

So, the adversary A can defeat the signcryption by means of analyzing the ciphertext, and at the same time he can solve the BIDH problem with nonnegligible advantage. But we all know that there is no algorithm that can be used to work out the BIDH problem in the probabilistic polynomial time; hence our scheme has the indistinguishability against adaptive chosen ciphertext attack. □

(2) Unforgeability

Theorem 8. *Our CDHSC scheme is existentially unforgeable against any EUF-CDHSC-CMA adversary $A_{i\ (i=\mathrm{I,II})}$ in the random oracle model assuming that the CDH problem in G_1 is intractable.*

Proof. Let C be an CDH problem attacker. Then A_i is the adversary who interacts with C following Game 2. C is given (P, aP, bP) as an input to the CDH problem and aims to compute abP, where $a, b \in Z_q^*$ and $P \in G_1$.

Initial. The challenger C runs the *Setup* algorithm defined in generic model and gives the resulting system parameters to the adversary A_i. For the Type II adversary, C sends him the master secret keys of KGC in addition to the system parameters.

Probing. We show that C can use A_i to solve the CDH problem. C needs to maintain three lists L_1, L_2, and L_3 that are initially empty and are used to keep track of answers to queries asked by A_i to oracles H_1, H_2, and H_3, respectively. What is more, C maintains two lists: L_k and L_p of the queries made by A_i to oracles PK and PPUKE (or PKE), respectively. Subsequently, C simulates the challenger and plays the game described in Definition 5 with the adversary A_i. The A_i performs a polynomially bounded number of the following queries, and A_{II} does not need to perform PKR, PPKE, and KE queries.

Partial Public Key Extraction (PPUKE) or Public Key Extraction (PKE) Queries. When A_i makes this query on ID_i, if the list L_p has the entry $(\mathrm{ID}_i, Q_{\mathrm{ID}_i} = w_i P, D_{\mathrm{ID}_i} = w_i^{-1} P, w_i)$, then C answers Q_{ID_i} to A_i. Otherwise, C selects $w_i \in Z_q^*$ randomly, computes $Q_{\mathrm{ID}_i} = w_i P$ and $D_{\mathrm{ID}_i} = w_i^{-1} P$, then returns Q_{ID_i}, and adds $(\mathrm{ID}_i, Q_{\mathrm{ID}_i} = w_i P, D_{\mathrm{ID}_i} = w_i^{-1} P, w_i)$ to L_p.

Partial Private Key Extraction (PPKE) or Key Extraction (KE) Queries. We assume that C has made PPUKE (or PKE) query on ID_i before this query. The list L_p should have the entry $(\mathrm{ID}_i, Q_{\mathrm{ID}_i} = w_i P, D_{\mathrm{ID}_i} = w_i^{-1} P, w_i)$. C returns D_{ID_i} to A.

Public Key (PK) Queries. We assume that A_i makes at most q_k queries to this oracle and C has made PPUKE (or PKE) query on ID_i before this query. First, C chooses $\gamma \in \{1, 2, \ldots, q_k\}$ randomly. When A_i makes this query on ID_i ($i \in \{1, 2, \ldots, q_k\}$), if $i = \gamma$ (we let $\mathrm{ID}_i = \mathrm{ID}_\gamma$ at this point), C returns $\mathrm{PK}_{\mathrm{ID}_\gamma} = w_\gamma bP$ and adds $(\mathrm{ID}_\gamma, \mathrm{PK}_{\mathrm{ID}_\gamma}, \perp)$ to L_k. Otherwise C selects $x_{\mathrm{ID}_i} \in Z_q^*$ randomly, computes $\mathrm{PK}_{\mathrm{ID}_i} = x_{\mathrm{ID}_i} Q_{\mathrm{ID}_i}$, then returns $\mathrm{PK}_{\mathrm{ID}_i}$, and adds $(\mathrm{ID}_i, \mathrm{PK}_{\mathrm{ID}_i}, x_{\mathrm{ID}_i})$ to L_k.

Corruption Queries. When A_i makes this query on ID_i, if $\mathrm{ID}_i = \mathrm{ID}_\gamma$, C aborts the simulation and returns $\perp$. Otherwise, the list L_{k} should have the entry $(\mathrm{ID}_i, \mathrm{PK}_{\mathrm{ID}_i}, x_{\mathrm{ID}_i})$. C returns x_{ID_i} to A_i.

H_1, H_2, H_3, PKR, and Signcryption Queries. This proof is the same as the proof of Theorem 7.

Unsigncryption Queries. When A asks for this query on a signcryption $\sigma = (C, U, V)$ with a senders identity ID_i and a receivers identity ID_j, C computes $t = H_{3-0}(C, U, \mathrm{PK}_{\mathrm{ID}_i}, \mathrm{ID}_i)$ and checks if $\hat{e}(V, Q_{\mathrm{ID}_i}) = \hat{e}(P_0, P_0)^t \hat{e}(\mathrm{PK}_{\mathrm{ID}_i}, U)$ holds. If not, C aborts the simulation and returns $\perp$. Otherwise, C executes the KE query to get D_{ID_j} and calculates $T = \hat{e}(U, D_{\mathrm{ID}_j})$. Then C executes H_2 query to obtain $Z = H_2(U, T, \mathrm{ID}_j)$ and finally calculates and returns $m = c \oplus Z$.

Forgery. A_i outputs a ciphertext $\sigma^* = (C^*, U^* = aP, V^*)$ and picks a sender's identity ID_s and a receiver's identity ID_r on which he wishes to be challenged. If $\mathrm{ID}_s \neq \mathrm{ID}_\gamma$, C aborts the simulation and returns $\perp$. Otherwise, he runs the H_3 simulation algorithm to obtain $t^* = H_3(C^*, U^*, \mathrm{PK}_{\mathrm{ID}_\gamma}, \mathrm{ID}_\gamma)$; we can have $V^* = t^* D_{\mathrm{ID}_\gamma} + \omega$ and $\omega = V^* - t^* D_{\mathrm{ID}_\gamma}$ (ω is a candidate for the CDH problem). Finally, C checks if

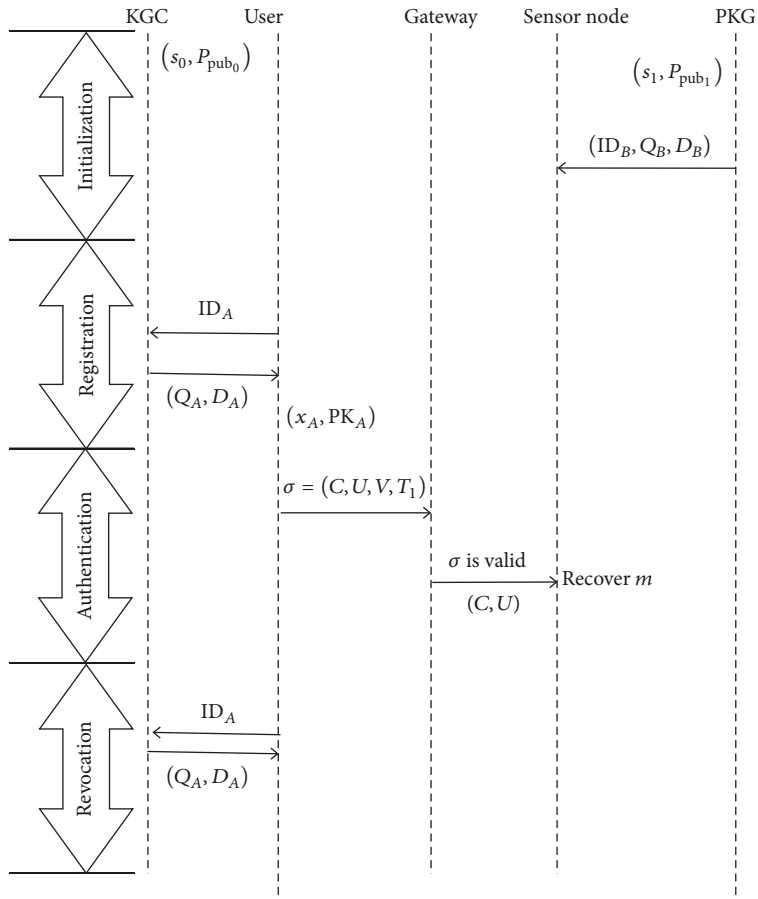


Figure 2: The proposed access control scheme.

$\hat{e}(V^*, Q_{ID_\gamma}) = \hat{e}(P_0, P_0)^{t^*} \hat{e}(PK_{ID_\gamma}, U^*)$, if the condition is not satisfied, C fails and outputs 0; otherwise, C answers 1.

So, the adversary A_i can forge a valid signcryption by means of analyzing ciphertext, and he can solve the CDH problem with nonnegligible advantage at the same time. But we all know that there is no algorithm that can be used to work out the CDH problem in the probabilistic polynomial time; hence our scheme is existentially unforgeable against adaptive chosen-message attacker A_i. □

(3) Known Session-Specific Temporary Information Security (KSSTIS). Assume that at the ith communication, session-specific temporary key r and signcryption $\sigma = (C, U, V)$ are leaked. In our CDHSC scheme, the encryption key is $Z = H_2(U, T, ID_B)$. An external adversary can get U through the ciphertext σ and the identity ID_B of the receiver, but he cannot acquire the secret value x_A of the sender or the private key D_B of the receiver. So, the external adversary is hard to obtain the plaintext m since he cannot compute $T = \hat{e}(P_2, P_2)^{r+x_A} = \hat{e}(U, D_B)$. Hence, our scheme can achieve the KSSTIS attribute. But in NACS [21], when the external adversary obtains the temporary key r of ith communication, he can compute $T = \hat{e}(P_{pub}, Q_{ID_r})^r$ easily, and it is easy for the adversary to obtain the message $m = C \oplus H_2(U, T, ID_r)$ with the ciphertext $\sigma = (U, C, V)$.

4. An Efficient Access Control Scheme

We come up with a secure and efficient access control scheme for wireless sensor networks in the cross-domain context of the IoT using our CDHSC scheme. The proposed scheme includes four phases: system initialization, user registration, authentication with key establishment, and leaked key revocation phase. The workflow of access control scheme is shown in Figure 2.

4.1. System Initialization Phase. Without loss of generality, we select two domains and assume that the KGC and PKG are in different communication domains and they generate different system parameters. The KGC outputs the system parameters $\{G_{1-0}, G_{2-0}, q_0, \hat{e}, n, P_0, P_{pub_0}, H_{1-0}, H_2, H_3\}$ and obtains his master secret key s_0 and the master public key $P_{pub_0} = s_0 P_0$. The PKG gets his master secret key s_1 and the master public key $P_{pub_1} = s_1 P_1$, and he runs the IB-PKE and IB-KE algorithms to generate the users public key Q_U and private key D_U for each WSN node in the IBC environment. The PKG loads the system parameters, Q_U, D_U, and ID_U, into a smart card and issues this card to the WSN node.

4.2. User Registration Phase. At this stage, an Internet user with identity ID_A in the context of CLC wants to register for his partial public/private key pair; he submits his identity

Table 1: Comparisons of performance.

Scheme	KISSTIS	Cross-domain	Computational cost			Sensor communication cost	
			Signcryption	Unsigncryption	Energy cost	Received data	Energy cost
NACS [21]	N	N	$3M + 1E$	$4P$	260.16 mJ	$\|m\| + \|G_1\|$	1.03 mJ
Ours	Y	Y	$3M + 1E$	$3P + 1E$	234 mJ	$\|m\| + \|G_1\|$	1.03 mJ

ID_A to the KGC. KGC examines if the user's information (e.g., the user's IP address) is reasonable. If the information is incorrect, the KGC will reject the user's request. Otherwise, the KGC executes CL-PPUKE and CL-PPKE algorithms to get partial public key $Q_A = (a + s_0 + \chi_A)P_0$ and partial private key $D_A = (a+s_0+\chi_A)^{-1}P_0$, where $a = H_{1-0}(\mathrm{ID}_A)$ and $\chi_A \in Z_q^*$ selected by the KGC randomly. The KGC returns (Q_A, D_A) to user. After receiving (Q_A, D_A), the Internet user executes the CL-SVS and CL-PKG algorithms to obtain his secret value x_A and public key PK_A. The user can store the public parameters in a plaintext file and (x_A, D_A) in a ciphertext file.

4.3. Authentication with Key Establishment Phase. When an Internet user ID_A wants to access the information collected by a sensor node ID_B from WSN, the user firstly acquires the current time stamp T_1 in order to detect the replay attack, then generates the query request message, and performs a signcryption operation on it. The ciphertext is $\sigma = (C, U, V, T_1)$, where $V = tD_A + x_A U$ and $t = H_{3-0}(C, U, \mathrm{PK}_A, \mathrm{ID}_A \oplus T_1)$. Then he sends the ciphertext to the gateway belonging to the destination WSN. The gateway first calculates $t = H_{3-0}(C, U, \mathrm{PK}_A, \mathrm{ID}_A \oplus T_1)$ and examines whether $\hat{e}(V, Q_A) = \hat{e}(P_0, P_0)^t \hat{e}(\mathrm{PK}_A, U)$ and T_1 is fresh or not. If not, the gateway denies the access to the WSN. Otherwise, the user passes the authentication, and the gateway sends (C, U) to the WSN node. The WSN node calculates $T = \hat{e}(U, D_B)$, $Z = H_2(U, T, \mathrm{ID}_B)$ and gets query request message $m = C \oplus Z$. After that, the WSN node can encrypt the response data using symmetric encryption algorithm with the session key Z. In this process, confidentiality, nonrepudiation, and KSSTIS are all achieved according to the security proof of Section 3.5. The message integrity is ensured by using the hash value t. The functions of authentication and session key establishment are implemented by verifying the signature (U,V) and calculating the session key Z, respectively.

4.4. Leaked Key Revocation Phase. Assume that the partial private key D_A of an Internet user with identity ID_A is leaked; then the user should send a key revocation request message to the KGC for a new key. The user submits his identity ID_A to the KGC. KGC randomly chooses another value $\chi_A' \in Z_q^*$ to compute the new partial public key $Q_A = (a + s_0 + \chi_A')P_0$ and partial private key $D_A = (a + s_0 + \chi_A')^{-1}P_0$ with the same identity ID_A. Then the KGC returns (Q_A, D_A) to user.

4.5. Performance Evaluation. We compare the performance of our method with the NACS [21]. The comparative result is shown in Table 1. In the table, we use M, E, and P as abbreviations for point multiplications in G_1, exponentiations in G_2, and pairing operations, respectively. Moreover, we use notations KSSTIS as abbreviations for whether the scheme achieves known session-specific temporary information security.

From the computational point of view, the signcryption operation of our CDHSC scheme needs three point multiplications in G_1 and one exponentiation in G_2 which is the same as NACS [21]. The unsigncryption operation of our CDHSC scheme needs one exponentiation and three pairings, but NACS requires four pairings. As we all know, the pairing operation is several times more expensive than the exponentiation. So the computational cost of our CDHSC scheme is more efficient than the NACS. In addition to efficiency improvement, our CDHSC scheme also enhances the security, since it achieves KSSTIS attribute. Most importantly, our scheme allows an Internet user under the circumstance of CLC to communicate a sensor node in IBC environment with different system parameters.

For energy consumption, according to [21], a point multiplication in G_1 (or an exponentiation in G_2) operation and a pairing operation consume 19.44 mJ and 45.6 mJ, respectively. Therefore, the computational energy cost of NACS and our scheme are $4 * (19.44 + 45.6) = 260.16$ mJ and $5 * 19.44 + 3 * 45.6 = 234$ mJ, respectively, and the communication energy cost of NACS and our scheme are the same for the sensor node (the cost is 1.03 mJ [21]).

Hence, consider the wireless sensor networks in the single-domain or cross-domain context of the IoT; it may be that our access control scheme is more applicable.

5. Conclusion

In this paper, we proposed a cross-domain heterogeneous signcryption scheme that allows a sender in the CLC environment to send the request signcryption message to a recipient in the IBC environment with different system parameters, and we proved that it has the confidentiality under the BIDH problem and unforgeability under the CDH problem in the random oracle model. Based on the CDHSC scheme, we designed a secure and efficient access control scheme for wireless sensor networks in the cross-domain context of the IoT.

Compared with NACS, our scheme not only needs less computation costs but also has stronger security since it achieves KSSTIS attribute. We believe that the proposed access control scheme can be feasible in many practical single-domain or cross-domain WSN applications.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors thank the anonymous referees for their valuable suggestions and comments. This work is supported by the National Natural Science Foundation of China (nos. 61662046 and 61601215) and the Science and Technology Research Project of Jiangxi Province of China (no. 20171BCB23014 and no. 20142BBE50019).

References

[1] J. Granjal, E. Monteiro, and J. S. Silva, "Security in the integration of low-power Wireless Sensor Networks with the Internet: A survey," *Ad Hoc Networks*, vol. 24, pp. 264–287, 2015.

[2] H. Kumarage, I. Khalil, A. Alabdulatif, Z. Tari, and X. Yi, "Secure data analytics for cloud-integrated internet of things applications," *IEEE Cloud Computing*, vol. 3, no. 2, pp. 46–56, 2016.

[3] A. Botta, W. de Donato, V. Persico, and A. Pescapé, "Integration of cloud computing and internet of things: a survey," *Future Generation Computer Systems*, vol. 56, pp. 684–700, 2016.

[4] M. Khan, B. N. Silva, and K. Han, "A web of things-based emerging sensor network architecture for smart control systems," *Sensors*, vol. 17, no. 2, article no. 332, 2017.

[5] K. C. Serdaroglu and S. Baydere, "WiSEGATE: wireless sensor network gateway framework for internet of things," *Wireless Networks*, vol. 22, no. 5, pp. 1475–1491, 2016.

[6] V. C. Thang and N. V. Tao, "A performance evaluation of improved IPv6 routing protocol for wireless sensor networks," *International Journal of Intelligent Systems and Applications*, vol. 8, no. 12, pp. 18–25, 2016.

[7] S. Raza, L. Seitz, D. Sitenkov, and G. Selander, "S3K: Scalable security with symmetric keys - DTLS key establishment for the internet of things," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 3, pp. 1270–1280, 2016.

[8] Y. Zhou, Y. Zhang, and Y. Fang, "Access control in wireless sensor networks," *Ad Hoc Networks*, vol. 5, no. 1, pp. 3–13, 2007.

[9] H.-F. Huang, "A novel access control protocol for secure sensor networks," *Computer Standards & Interfaces*, vol. 31, no. 2, pp. 272–276, 2009.

[10] H.-S. Kim and S.-W. Lee, "Enhanced novel access control protocol over wireless sensor networks," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 2, pp. 492–498, 2009.

[11] H. Lee, K. Shin, and D. H. Lee, "PACPs: practical access control protocols for wireless sensor networks," *IEEE Transactions on Consumer Electronics*, vol. 58, no. 2, pp. 491–499, 2012.

[12] C.-Y. Chen, A. D. Yein, T.-C. Hsu, J. Y. Chiang, and W.-S. Hsieh, "Secure access control method for wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 11, no. 7, Article ID 261906, 2015.

[13] P. Kumar, A. Gurtov, J. Iinatti, M. Sain, and P. Ha, "Access control protocol with node privacy in wireless sensor networks," *IEEE Sensors Journal*, vol. 16, no. 22, pp. 8142–8150, 2016.

[14] H. Yu, J. He, T. Zhang, P. Xiao, and Y. Zhang, "Enabling end-to-end secure communication between wireless sensor networks and the Internet," *World Wide Web*, vol. 16, no. 4, pp. 515–540, 2013.

[15] C. Ma, K. Xue, and P. Hong, "Distributed access control with adaptive privacy preserving property for wireless sensor networks," *Security and Communication Networks*, vol. 7, no. 4, pp. 759–773, 2014.

[16] N.-W. Lo and J.-L. Tsai, "An efficient conditional privacy-preserving authentication scheme for vehicular sensor networks without pairings," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1319–1328, 2016.

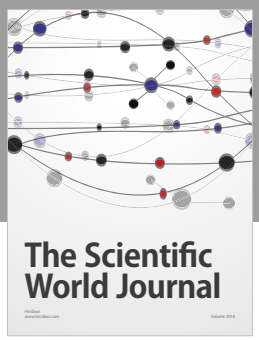
[17] S. Bala, G. Sharma, and A. K. Verma, "PF-ID-2PAKA: pairing free identity-based two-party authenticated key agreement protocol for wireless sensor networks," *Wireless Personal Communications*, vol. 87, no. 3, pp. 995–1012, 2016.

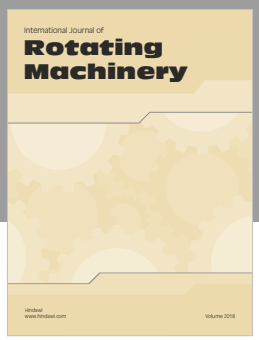
[18] G. Sharma, S. Bala, and A. K. Verma, "An improved RSA-based certificateless signature scheme for wireless sensor networks," *International Journal of Network Security*, vol. 18, no. 1, pp. 82–89, 2016.

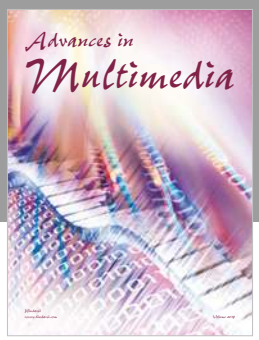
[19] A. A. Omala, N. Robert, and F. Li, "A provably-secure transmission scheme for wireless body area networks," *IEEE Cloud Computing*, vol. 40, no. 11, article no. 247, 2016.

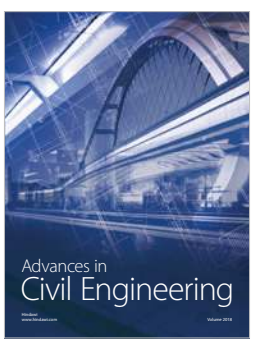
[20] Q. Huang, D. S. Wong, and G. Yang, "Heterogeneous signcryption with key privacy," *The Computer Journal*, vol. 54, no. 4, pp. 525–536, 2011.

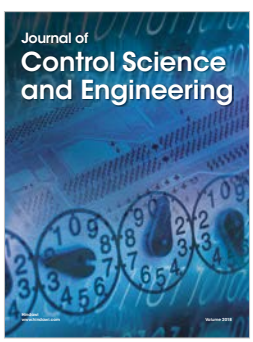
[21] F. Li, Y. Han, and C. Jin, "Practical access control for sensor networks in the context of the Internet of Things," *Computer Communications*, vol. 89, no. 1, pp. 154–164, 2016.

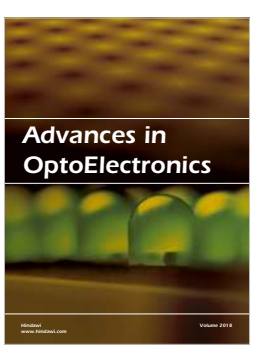


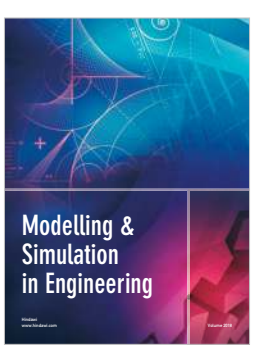


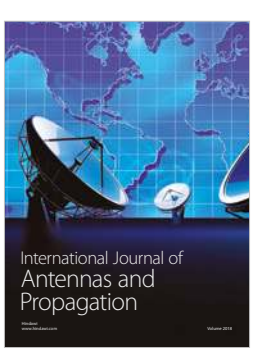


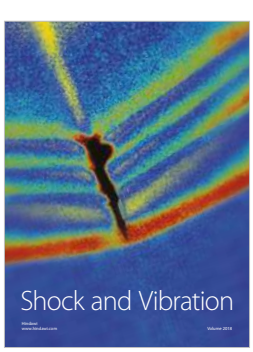


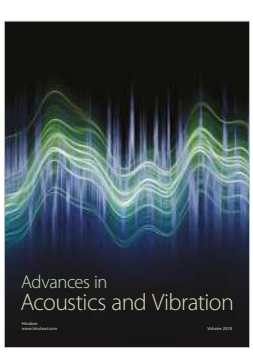












Hindawi

Submit your manuscripts at
www.hindawi.com