Research Article

Securing Topology Control in SDWSNs Using Identity-Based Cryptography

Rui Wang,1,2 Donglan Liu,1 Xin Liu,1 Jianfei Chen,3 Lei Ma,1 Yong Wang,3 Hao Zhang,1 Hao Yu,1 Bing Su,1 Fangzhe Zhang,1 and Lili Sun1

1State Grid Shandong Electric Power Research Institute, Jinan, China
2Shandong Smart Grid Technology Innovation Center, Jinan, China
3State Grid Shandong Electric Power Company, Jinan, China

Correspondence should be addressed to Rui Wang; wangrui_dky@163.com

Received 29 January 2022; Revised 30 April 2023; Accepted 26 June 2023; Published 19 July 2023

In software-defined wireless sensor networks (SDWSNs), topology control is a fundamental procedure to maintain the global network topology. However, the open wireless channels of SDWSNs make it possible for an attacker to eavesdrop, replay, or modify the topology messages, thus posing a great threat to the network operations. The security of SDWSN topology control has not received enough attention yet. Identity-based cryptography (IBC) may be better for SDWSNs due to its capability of generating the public key from the node identity directly, compared with traditional cryptography. In particular, identity-based combined encryption and signature cryptography (IBCES) could encrypt and sign the messages using the same identity. As such, to secure the confidentiality, integrity, and authentication of topology information, we put forward a secure topology control mechanism based on IBCES. First, we use an identity-based encryption authenticated key agreement scheme to implement the authentication of neighbor nodes and hop-to-hop verification via secure neighbor discovery and topology discovery processes. Then through the node admission and key establishment process, the end-to-end secure channels are established between the nodes, sinks, and Controller. Finally, secure topology collection and management processes supporting flat and hierarchical network structures are designed to guarantee the security of topology information. Theoretical analysis shows that our methods could satisfy the security needs of SDWSN topology control and resist several security attacks. The experimental results indicate that our mechanisms are suitable for SDWSNs.

1. Introduction

Nowadays, wireless sensor networks (WSNs) are faced with many inherent problems and many researchers have introduced software-defined networking (SDN) to WSNs and proposed software-defined wireless sensor networks (SDWSNs) [1–4]. SDWSNs can achieve flexible network management and have been applied in smart home [5], smart grids [6], 5G network [7], and other scenarios. In the SDWSN paradigm, the forwarding layer is separated from the control layer and the sensor nodes are managed by a logically centralized Controller. Topology control is one of the most critical procedures of SDWSNs which usually include two processes: topology discovery (TD) (construction) and topology maintenance [8–10]. The Controller periodicity collects the underlying topology messages to form the network topology [11]. Although this is a promising approach, one of the weaknesses of SDWSNs is the lack of authentication and encryption mechanisms [12]. SDWSNs use the open wireless link to transmit the data and control packets [13] which makes topology messages vulnerable to eavesdropping and tampering attacks. Any new node can join the network, which may also lead to many kinds of attacks, such as man-in-the-middle and replay attacks. However, the security of SDWSN topology control has not received enough attention [12, 14].

Cryptography can solve these problems by protecting network communications and defending against external attackers. Cryptography usually includes symmetric cryptography and asymmetric cryptography. Symmetric cryptography is fast and has low-energy consumption, but it has key
management and scalability challenges in large-scale networks [15]. Asymmetric cryptography can easily establish secure communication through key agreement algorithms. But public key infrastructure (PKI) and certificates are usually used for traditional asymmetric cryptography, the complexity and computation of certificate operations make it face high implementation cost challenges in WSNs. In recent years, identity-based cryptography (IBC) has received a lot of attention in the WSN field. Compared with traditional PKI, IBC’s public key can be calculated from the entity’s identity (e.g., IP address, name), making it require fewer resources in terms of processing power, storage, and communication bandwidths [16]. There have been a number of IBC-based WSN and the Internet of Things (IoT) security designs [17].

Because the Controller is trusted by all nodes in the SDWSN network, it can act as the Private Key Generator (PKG) for the IBC. The IBC does not require certificate operations which reduces the communication between nodes and Controller. Therefore, IBC is also fit for SDWSNs. In particular, combining identity-based encryption (IBE) and identity-based signature (IBS) enables the construction of identity-based combined encryption and signature (IBCES) schemes. IBCES can use the same identity for encryption and signature operations, which greatly saves storage space and reduces the number of identities. Based on IBC and IBCES, we propose a secure topology control scheme for SDWSNs, which is used to verify the authenticity of nodes, establish secure communication, construct correct network topology, and authenticate control messages. To the best of our knowledge, we are the first to study the SDWSN secure topology control mechanism based on IBC. Our main contributions are summarized as follows:

1. Using an IBE-based authenticated key agreement mechanism, we present a secure neighborhood discovery mechanism to authenticate neighbor nodes while obtaining the shared keys. Then we present a secure TD scheme to achieve a valid route to the Controller through hop-to-hop verification.
2. We design a scheme of node admission and key establishment to allow new nodes to be authenticated and trusted by the sink and Controller and also setup the end-to-end secure channels between them.
3. We put forward secure topology collection and management schemes based on IBCES that support both flat and cluster network structures. The confidentiality, integrity, and authentication of topology information can be protected.
4. Theoretical analysis shows that our methods could satisfy the security needs of SDWSN topology control and resist several security attacks, such as man-in-the-middle attack, replay attack, and so on. The experimental results on middle-class WSN devices, such as Raspberry Pi, demonstrate the feasibility of our approaches for SDWSNs.

The remainder of this paper is organized as below. Section 2 discusses the related works. Section 3 introduces the basic concept of IBCES and the motivation of this paper. The secure neighbor node discovery and topology control mechanisms are described in detail in Sections 4. Security analysis and experimental results are showed in Sections 5 and 6. In the end, Section 7 summarizes our work.

2. Related Work

2.1. Topology Control in SDWSNs. Topology control plays a very important role in SDWSNs’ network operations [10]. Galluccio et al. [18] presented a state-of-the-art TD protocol for SDWSNs. Each node runs the TD protocol to maintain its current neighbors and the next hops toward the Controller. Theodorou et al. [10] proposed a software-defined topology control scheme to improve the management, control, and operation of WSNs. However, they do not consider security. Malicious nodes can be able to join the network, modify the topology messages, and disturb the network.

On account of SDN-WISE, Anadiotis et al. [19] made use of a TPM module to implement basic security primitives, such as cryptographic key generation and storage, data encryption/decryption, and realized a prototype through a Raspberry Pi equipped with an Infineon Optiga TPM. But, the TPM is mainly used to leverage software and sensor attestation and there is no in-depth study on threats in topology control. Wang et al. [20] designed a centralized trust management mechanism with an energy-efficient topology collection scheme to detect and remove malicious nodes.

Miguel et al. [21] proposed a new SDWSN control plane based on the constrained application protocol (CoAP) which can provide the basic networking functions (TD, topology maintenance, and flow control) of the Controllers. However, they are all based on the assumption that secure communication is already established between nodes and Controller. The security research of topology control in SDWSNs is still in its infancy [10, 22].

2.2. Cryptography in WSNs. Cryptography is the mathematical basis for all security mechanisms [23]. For instance, cryptographic-based strategies are significant methods to achieve privacy data protection [24] and digital forensic inspection [25]. Cryptography techniques can be primarily divided into symmetric and asymmetric cryptography [26]. Symmetric cryptography has been widely used in WSNs due to its low-energy consumption and fast encryption speed. But it has the drawback of scalability because it is hard to realize the key predistribution in large-scale networks. Asymmetric cryptography provides nonrepudiation and easy key distribution. However, due to the complexity and computation overhead of certificate distribution, storage, verification, and cancelation, the traditional asymmetric cryptography mechanisms are not feasible in WSNs. In contrast, the public key of IBC can be directly derived from the identity of the node and key management will be simplified [27]. IBC is considered a suitable choice for WSNs [28].

In recent years, many IBC-based security methods in WSNs and the IoT have been proposed [17]. With the aid of IBC, McCusker and O’Connor [28] constructed a scheme for key distribution and network access in WSNs and...
designed a hardware implementation of IBC to meet the strict energy constraint of sensor node. Utilizing IBC, Li et al. [27] presented a lightweight TLS protocol for end-to-end secure communication in the IP-based IoT. Karantaidou et al. [29] and Mazhar Rathore et al. [30] both conducted IBC feasibility tests on middle-class WSN nodes, such as the Raspberry Pi 3. These results show that IBC is suitable for WSN resource-constrained devices.

2.3. Authentication in WSNs. In WSNs, authentication can be used to avoid attacks against secure communication [31]. At present, authentication algorithms are constantly emerging. Jiang et al. [32] presented a three-factor authentication protocol using the physical unclonable function (PUF) for the Internet of Vehicles. The PUF is introduced in the AKE protocol using the physical unclonable function (PUF) for mobile Ad Hoc networks based on an authenticated identity authentication. In the Internet of Vehicles. The PUF is introduced in the AKE protocol using the physical unclonable function (PUF) for mobile Ad Hoc networks based on an authenticated identity authentication. However, they did not test in an SDN environment, nor they performed a performance analysis. These schemes are still beneficial to the application of IBC in SDWSN scenarios [13].

3. Identity-Based Combined Encryption and Signature Cryptography: An Overview

IBC-based schemes can be implemented with two main cryptographic approaches: Elliptic Curve Cryptography (ECC) and pairing-based cryptography (PBC) [23]. The focus of this paper is PBC. In this section, we briefly introduce the bilinear pairing used in PBC and the related computational problems. Then we describe the IBCES and the motivation of this paper.

3.1. Bilinear Pairing and Related Diffie–Hellman Problems. Let $G_1$ be a cyclic additive group and $G_2$ be a cyclic multiplicative group. Both of them have a large prime order $q$. The bilinear pairing is a map between two groups $\hat{\epsilon}: G_1 \times G_1 \rightarrow G_2$ that satisfies the following properties:

(i) Bilinearity: $\hat{\epsilon}(aP, bQ) = \hat{\epsilon}(P, Q)^{ab}$, for all $P, Q \in G_1$ and $a, b \in \mathbb{Z}_q^*$.

(ii) Nondegeneracy: there exists $P, Q \in G_1$ such that $\hat{\epsilon}(aP, bQ) \neq 1$. It means the map does not send all the pairs in $G_1 \times G_1$ to the identity in $G_2$.

(iii) Computability: there exists an efficient algorithm to compute $\hat{\epsilon}(aP, bQ)$ for all $P, Q \in G_1$.

The security of PBC is usually based on two kinds of problems (alternatively known as assumptions) as:

(i) The computational Bilinear Diffie–Hellman (CBDH) Problem is a problem that given the set of $(P, aP, bP, cP) \in G_1$ for some integers $a, b, c \in \mathbb{Z}_q^*$, compute $\hat{\epsilon}(P, cP) = \hat{\epsilon}(P, Q)^{ab} \in G_2$.

(ii) The decisional Bilinear Diffie–Hellman (DBDH) Problem is a problem that given two sets of $(P, aP, bP, abP)$ and $(P, aP, bP, cP)$ for integer $c \in \mathbb{Z}_q^*$, determine whether $c = ab$.

3.2. Identity-Based Cryptography. IBC mainly includes an encryption system and a signature system. Here we mainly introduce Boneh and Franklin’s [43] identity-based encryption (BF-IBE) and Cha and Cheon’s [44] identity-based signature (CC-IBS). BF-IBE consists of four algorithms.

(i) Setup:

1. Generate an additive group $G_1$ and a multiplicative group $G_2$ with the same prime order $q$, a bilinear pairing map $\hat{\epsilon}: G_1 \times G_1 \rightarrow G_2$.

2. Pick a random generator $P \in G_1$, a random integer $s \in \mathbb{Z}_q^*$, and compute $P_{pub} = sP$.

3. Pick two secure hash functions $H_i, i = 1, 2$. $H_1: \{0, 1\}^n \rightarrow G_1$, $H_2: G_2 \rightarrow \{0, 1\}^n$, $H_3: \{0, 1\}^n \times \{0, 1\}^n \rightarrow \mathbb{Z}_q^*$, $H_4: \{0, 1\}^n \rightarrow \{0, 1\}^n$. Here, $n$ is the length of a message to be encrypted or
signed. The PKG keeps the master key \(s\) secretly and publishes the system public parameters as:

\[
\text{params} = \{G_1, G_2, n, q, \hat{\epsilon}, P, P_{pub}, H_1, H_2, H_3, H_4\}. 
\]

(ii) Extract: given a user’s identity \(ID_U \in \{0, 1\}^*\), the PKG computes:

1. The user’s public key as \(Q_U = H_1(ID_U)\).
2. The user’s private key as \(S_U = sQ_U\).

(iii) Encryption: to encrypt a message \(m \in \{0, 1\}^n\) for \(ID_U\), the sender performs the following steps:

1. Pick a random generator \(\alpha \in \{0, 1\}^n\) at random.
2. Calculate \(r = H_3(\alpha, m)\).
3. Calculate \(V = rP, W = \alpha \oplus H_2(\hat{\epsilon}(P_{pub}, Q_U)^\epsilon)\) and \(T = m \oplus H_4(\alpha)\).
4. Send ciphertext \(c = (V, W, T)\) to the receiver.
5. Decryption: to decrypt \(c\), the receiver \(ID_U\) does the steps below:

   1. Calculate \(\alpha = W \oplus H_2(\hat{\epsilon}(V, S_U))\).
   2. Calculate \(m = T \oplus H_4(\alpha)\).
   3. Set \(r = H_3(\alpha, m)\) and check the equation \(V = rP\).

The receiver will output a correct symbol (\(\top\)) or an error symbol (\(\bot\)) according to whether the above equation is correct or not.

CC-IBS also includes four algorithms:

(i) Setup:

1. Generate an additive group \(G_1\) and a multiplicative group \(G_2\) with the same prime order \(q\), a bilinear pairing map \(\hat{\epsilon} : G_1 \times G_1 \rightarrow G_2\).
2. Pick a random generator \(P \in G_1\), a random integer \(s \in Z_q^*\), and compute \(P_{pub} = sp\).
3. Pick two secure hash functions \(H_i, i = 1, 2, H_1 : \{0, 1\}^n \rightarrow G_1\), \(H_2 : \{0, 1\} \times G_1 \rightarrow Z_q^*\). The PKG keeps the master key \(s\) secretly and publishes the system public parameters as:

\[
\text{params} = \{G_1, G_2, n, q, \hat{\epsilon}, P_{pub}, H_1, H_2\}. 
\]

(ii) Extract: given a user’s identity \(ID_U \in \{0, 1\}^*\), the PKG computes:

1. The user’s public key as \(Q_U = H_1(ID_U)\).
2. The user’s private key as \(S_U = sQ_U\).

(iii) Sign: to sign a message \(m \in \{0, 1\}^n\), the signer \(ID_U\) performs the following steps:

1. Pick a random generator \(r \in Z_q^*\).
2. Calculate \(V = rQ_U, h = H_2(m, V), W = (r + h)S_U\).
3. The signature of \(m\) is \(\sigma = (V, W)\).

(iv) Verify: to verify that \(\sigma\) is a legitimate signature for the message \(m\) and the identity \(ID_U\), the verifier performs two actions:

1. Calculate \(h = H_2(m, V)\).
2. Verify the equation \(\hat{\epsilon}(P_{pub}, V + hQ_U)\) or an error symbol (\(\bot\)).

The verifier will output a correct symbol (\(\top\)) if the above equation is correct. Otherwise, the verifier outputs an error symbol (\(\bot\)).

3.3. Identity-Based Combined Encryption and Signature (IBCES).

The combined public key cryptosystem uses the same key pair in several cryptographic primitives [45], such as encryption and decryption. Li et al. [27] demonstrated that BF-IBE and CC-IBS could be used simultaneously [46] in 2019. As shown in Figure 1, by combining BF-IBE with CC-IBS, an IBCES mechanism is constructed.

The tasks of key management and storage space of IBCES could be reduced because it can use the same key pair for signature and encryption operations. IBCES mainly includes six algorithms. Based on the previous introduction to BF-IBE and CC-IBS, we give a brief introduction to IBCES as follows:

(i) Setup: the system setup algorithm is performed by PKG. Given a security parameter \(k\), the PKG performs the following:

1. Generate an additive group \(G_1\) and a multiplicative group \(G_2\) with the same prime order \(q\), a bilinear pairing map \(\hat{\epsilon} : G_1 \times G_1 \rightarrow G_2\).
2. Pick a random generator \(P \in G_1\), a random integer \(s \in Z_q^*\), and compute \(P_{pub} = sp\).
3. Pick two secure hash functions \(H_i, i = 1, 2, \ldots, 5\). The PKG keeps the master key \(s\) secretly and publishes the system public parameters as:

\[
\text{params} = \{G_1, G_2, n, q, \hat{\epsilon}, P_{pub}, H_1\}. 
\]

(ii) Extract: the extract algorithm is also performed by PKG. Based on the identity of one user \(ID_U \in \{0, 1\}^*\), the PKG computes:

1. The user’s public key as \(Q_U = H_1(ID_U)\).
2. The user’s private key as \(S_U = sQ_U\).

here, \(H_1 : \{0, 1\}^* \rightarrow G_1\) is a map-to-point operation that takes a string and converts it into a group element. PKG will send \(S_U\) to the user via a secure way.

(iii) Encrypt: the encryption algorithm takes as input

\[
\text{params}, \text{the receiver’s identity } ID_U, \text{ and a plaintext } m, \text{ and then outputs a ciphertext } c. \text{ Here we represent it as } c = E_{ID_U}(m). \]
(iv) Decrypt: the decryption algorithm takes as input *params*, a ciphertext *c*, the receiver’s identity *ID_R*, and private key *S_R* and then outputs the plaintext *m*. Here we represent it as *m* = *D_S_R*(*c*).

(v) Sign: the signature algorithm takes as input the plaintext *m*, the signer’s identity *ID_S*, and private key *S_S* and then outputs the signature *σ*. Here we represent it as *σ* = *S_S*(*m*).

(vi) Verify: the verification algorithm takes as input *params*, the message *m*, the signature *σ*, and the signer’s identity *ID_S* and then outputs a correct symbol (†) or an error symbol (⊥). Here we represent it as † or ⊥ = *V_ID_S*(*m*, *σ*).

3.4. Motivation. The SDWSN Controller periodically collects the underlying topology information generated by nodes to maintain the global topology view and achieve centralized network control. Whereas, the data and control messages of SDWSNs are forwarded in the same wireless link, which makes the transmission insecure [13]. Since SDWSNs lack authentication and encryption mechanisms, any new node can join the network. As shown in Figure 2, the Sybil attackers can fake identities and destroy the normal topology operations of SDWSNs. For example, the Sybil attacker could use multiple identities to forge paths close to the sink to attract network traffic. This disturbs the normal neighborhood detection and topology collections and leads to significant degradation of routing performance [47]. Therefore, the node authentication, information integrity, and confidentiality of the SDWSN topology control procedure should be protected.

As discussed in Section 2.2, the symmetric cryptosystem is difficult for key distribution and management in WSNs. The new node is hard to join in a preexisting sensor network because the currently deployed nodes cannot be sure to own or find pairwise keys for newly added nodes. In addition to the heavy computation and bandwidth overhead, the traditional PKI solutions may also be unsuitable for SDWSNs. Each node must validate the certificates before communicating with its neighbor nodes. However, if a node wants to send a verification request to the certificate authority (CA), it must confirm which neighbor is legal and reliable. This not only burdens the overhead of the control plane but also causes problems under the centralized control of SDWSNs. IBC has the advantage of calculating the node identity as a public key without using a certificate to bind the public key with its identity [37]. There is no need for distributed keys previously and avoids expensive certificate operations. This facility is attractive in SDWSNs.

Particularly, the BF-IBE can realize mutual authentication and share the same symmetric keys between two nodes. The CC-IBS can guarantee the integrity and authentication of topology control packets. Therefore, to protect communication privacy and security, we adopt the above IBCES into the topology control of SDWSNs. We put forward a secure topology control scheme based on IBCES, including secure neighbor discovery, TD, node admission, topology collection, and topology management schemes. Therefore, Controller would maintain the correct topology and manage the underlying network relying on the security of topology information and the authenticity of SDWSN nodes.

4. Design of the Secure Topology Control Mechanism

We first give the system model and security model of our secure topology control mechanism. And then we list the assumptions and notations on the proposed mechanism. Finally, we propose its design in detail.

4.1. System Model and Security Requirements. To guarantee the security of the topology message, we design a secure topology control mechanism. The network model of our secure topology control mechanism is shown in Figure 3. The mechanism supports both flat and cluster network structures and includes four kinds of entities:

(1) The initiator node A: as a sensor node, node A collects environmental data and periodically sends the sensor data with topology information to the sink.
Node A first recognizes its neighbor nodes and initiates an authentication request to node B. And then Node A authenticates itself to the sink and Controller via its trusted neighbor nodes.

(2) The responder node B: node B is the neighbor of node A and needs to respond to node A’s authentication request. In a flat network structure, node B is a normal sensor node like node A. Considering the hierarchical network structure, node B can also be a cluster head and collect the topology information within its cluster and send the aggregated data to the sink.

(3) The Sink: the sink is basically a designated device similar to the normal sensor nodes but more powerful. The sink is responsible for collecting topology information from the sensor nodes. It is also a bridge between the nodes with the Controller.

(4) The Controller: since each SDWSN node trusts the Controller and it is responsible for configuring the network flow rules, it is a natural choice to act as the PKG of IBC (the Controller and PKG are used alternately in the later sections) [13]. Therefore, the Controller is in charge of issuing the identity-based private keys of the nodes and managing the underlying network based on the collected topology information.

The secure topology control mechanism mainly contains six parts: initialization, secure neighbor node discovery, secure TD, node admission with the key establishment, secure topology collection, and secure topology management. Figure 4 shows how our mechanism works in the view of node A and the two operations represented by the dotted line are unique to the cluster structure. The secure topology control mechanism utilizes IBCES to eliminate the binding between the public key and the certificate and the same identity could be used to achieve encryption and signature operations. An identity-based authenticated key exchange (IBAKE) protocol is applied to authenticate the communication entities while establishing the shared symmetric key. The security of topology control messages is guaranteed by an IBS protocol.

The following security requirements are essential for our secure topology control mechanism to achieve adequate security:

1. Mutual Authentication: Before actual communication occurs, both entities in the communication link need to authenticate each other to proceed with secure communications.
2. Forward Secrecy: The forward secrecy of an IBAKE protocol ensures that only protocol participants can compute the shared key. The external attacker cannot calculate the current and previous keys even if the private keys of the communicating parties are leaked.
3. Topology Message Security: To protect topology information from eavesdropping, tampering, and falsification, the data security of the topology message should be guaranteed.
4. Attack Defense: The attacks launched by the eavesdropper and malicious attackers should be defended, such as man-in-the-middle attack, replay attack, Sybil attack, Controller DoS attack, and compromised attack.

### 4.2. Attack Model and Security Model

#### 4.2.1. Attack Model

The capacities of adversary \( \mathcal{A} \) for the topology control mechanism can be summarized as follows:
4.2.2. Security Model. The standard acceptable security notion forIBE is indistinguishability against adaptive chosen ciphertext attacks under chosen identity attacks (IND-ID-CCA2) [46]. The standard security notion for IBS is existential unforgeability against adaptively chosen messages attacks under chosen identity attacks (EUF-ID-CMA) [46]. The adversary can access the decryption oracle and signature oracle at the same time since the same key pair can be used to decrypt a ciphertext and sign a message. Based on security models [48–50], two security models are defined to reflect this capability of the attacker. We give an overview of the security models below. A more detailed description can be obtained in literature [46].

**Definition 1.** As shown in Figure 5, if an adversary attacks a game between and challenger with an advantage at least , runs in maximum time , and makes at most queries, the IBCE’s encryption component is an -attacker. If no -forger exists, the IBCE’s encryption component and signature component is -EUF-IBCES-CMA [46].

**Definition 2.** As shown in Figure 6, if an adversary plays a game between and challenger with an advantage at least , runs in maximum time , and asks at most queries, the IBCE’s encryption component is an -attacker. If no -forger exists, the IBCE’s signature component is -EUF-IBCES-CMA [46].

(1) can eavesdrop, delete, intercept, replay, and modify the messages in the open wireless channel
(2) could forge an identity and attempt to access the network
(3) can obtain previous session keys
(4) can obtain the private key of a node when carrying out the forward secrecy attack.

4.3. Assumptions and Notations. To make our analysis clear and simple, we present the following assumptions:

(1) The SDWSN network is randomly deployed and static. (2) SDWSN nodes do not have prior knowledge about their neighbor nodes. They may be prone to attacks. (3) All the nodes have enough capabilities to run the IBCE cryptosystem and are homogeneous. (4) The Controller and sink are connected through Ethernet and the communication between them is secure. (5) The sink and Controller are always trusted and well-behaved.

Notations used in this section are listed in Table 1, where the column “Notation” contains the names of variables, and the definition of each variable is presented in the column “Definition”.

4.4. Prior to Deployment Phase. In the setup phase, the PKG runs the setup algorithm and generates the system parameters . Considering that different entities have different roles, such as Controller, sink, and sensor, we set the identity for each entity. For each SDWSN node, the PKG runs the key extraction algorithm to generate the public/private key pair for each node . The PKG places them along with on the nodes prior to deployment [23, 28].

4.5. Secure Neighbor Node Discovery. Due to the lack of unified management in traditional distributed WSNs, each node periodically sends broadcast messages within its transmission range to identify neighbors. Although SDWSNs
have global management capabilities, each node still needs to
discover its neighbor nodes. Considering that nodes may
be deployed in harsh environments, wireless nodes would
identify malicious neighbor nodes if security mechanisms
are not adopted. Therefore, each node needs to start the
neighbor discovery process after deployment and verifies the
authentication of neighbor nodes using an identity authentication
scheme.

Authenticated key exchange (AKE) establishes a session
key in a publicly insecure network, and then uses the established
session key in subsequent communications to protect communications data from eavesdropping or malicious
modification. IBAKE protocol provides AKE using identities
as public credentials which reduces the certificate overhead
[51]. Here, based on the IBAKE protocols of Cakulev et al.
[51] and Han and Li [52], we propose a BF-IBE based mutual
authentication mechanism with key agreement scheme. It
also contains a key confirmation protocol to ensure that
both nodes can compute the same symmetric key.

First, every node would broadcast a hello message con-
taining its identity. When the broadcast message is received,
the neighbor nodes of this node add the corresponding iden-
tity to their unauthenticated neighbor list. Node A \( T_{NB} \)
and node B \( T_{NB} \) (note that \( T_{NB} \) can be a sink or a cluster head) need
to authenticate each other before further communication. As
shown in Figure 7, the mutual authentication mechanism is described in detail as follows:

1. \( N_A \) randomly chooses \( a \in \mathbb{Z}_q^* \) and calculates \( T_A = aP \). Using \( N_B \)’s identity \( ID_B \) and BF-IBC algorithm, \( N_A \) encrypts the message \( C_1 = E_{ID_A}(ID_A||ID_B||T_A||TS_1) \). Here the timestamp \( TS \) is used to prevent the replay attack. Then \( N_A \) sends the neighbor authentication request \( R_1 = Request||C_1 \) to \( N_B \).

2. (a) Upon receiving the request \( R_1 \), \( N_B \) will reject this request if \( ID_A \) is not in the list of its unauthenticated neighbors. Otherwise, it decrypts the \( C_1 \) using its private key \( S_B \) and checks whether \( ID_B \) matches its identity. If so, \( N_B \) randomly chooses \( b \in \mathbb{Z}_q^* \) and calculates \( T_B = bP \). Using the \( N_A \)’s identity \( ID_A \) and BF-IBC algorithm, \( N_B \) encrypts the \( T_A \) together with \( T_B \) getting \( C_2 = E_{ID_A}(ID_B||ID_A||T_B||T_A||TS_2) \).

(b) \( N_B \) calculates the shared secret value \( K_{BA} = bT_A \). The shared key \( SK_{RA} \) can be generated by \( KDF(ID_A||ID_B||K_{BA}) \). \( N_B \) uses the \( SK_{RA} \) to cal-
culate the \( MAC_B \) of \( C_2 \). Then \( N_B \) sends the response \( R_2 = Response||C_2||MAC_B \) to \( N_A \).

3. (a) After receiving \( R_2 \), \( N_A \) decrypts the \( C_2 \) using \( S_A \). If \( T_A \) is consistent with the one it sends, \( N_A \) can ensure it really authenticates with \( N_B \). As the shared value \( K_{AB} = aT_B = (aP) = b(aP) = bT_A = K_{BA} \), the same symmetric key \( SK_{AB} \) can also be generated. By checking \( MAC_B \), \( N_A \) can be sure that \( N_B \) has generated the same \( SK_{AB} \).

---

**Table 1: Notations in this section.**

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( PKG )</td>
<td>Private key generator</td>
</tr>
<tr>
<td>( params )</td>
<td>System parameters</td>
</tr>
<tr>
<td>( \mathbb{G}_1 )</td>
<td>A cyclic additive group</td>
</tr>
<tr>
<td>( \mathbb{G}_2 )</td>
<td>A cyclic multiplicative group</td>
</tr>
<tr>
<td>( \mathbb{Z}_q^* )</td>
<td>A cyclic multiplicative group of module ( q ) (the elements of the group do not include ( 0 ))</td>
</tr>
<tr>
<td>( P )</td>
<td>The generator of ( \mathbb{G}_1 )</td>
</tr>
<tr>
<td>( H_i )</td>
<td>i-th one-way hash function, ( i = 1, 2, 3, 4, 5 )</td>
</tr>
<tr>
<td>( s )</td>
<td>The master key of ( PKG )</td>
</tr>
<tr>
<td>( P_{pub} )</td>
<td>The public key of ( PKG )</td>
</tr>
<tr>
<td>( n )</td>
<td>The length of a message to be encrypted or signed</td>
</tr>
<tr>
<td>( ID_i )</td>
<td>The identity of node ( i )</td>
</tr>
<tr>
<td>( S_i )</td>
<td>The private key of node ( i )</td>
</tr>
<tr>
<td>( Q_i )</td>
<td>The public key of node ( i )</td>
</tr>
<tr>
<td>( T_i )</td>
<td>The ephemeral key of node ( i )</td>
</tr>
<tr>
<td>( TS )</td>
<td>The timestamp of a message</td>
</tr>
<tr>
<td>( K_{AB} )</td>
<td>The shared secret value between nodes A and B</td>
</tr>
<tr>
<td>( KDF )</td>
<td>Key derivation function</td>
</tr>
<tr>
<td>( SK_{AB} )</td>
<td>The shared symmetric key between nodes A and B</td>
</tr>
<tr>
<td>( TN_i )</td>
<td>The trust neighbor node list of node ( i )</td>
</tr>
<tr>
<td>( UN_i )</td>
<td>The untrust neighbor node list of node ( i )</td>
</tr>
<tr>
<td>(</td>
<td></td>
</tr>
<tr>
<td>( MAC )</td>
<td>Message authentication code</td>
</tr>
<tr>
<td>( \rightarrow )</td>
<td>Unicast</td>
</tr>
<tr>
<td>( \Rightarrow )</td>
<td>Broadcast</td>
</tr>
</tbody>
</table>
Using Advanced Encryption Standard (AES) and $K_{AB}$, $N_A$ encrypts $T_B$ and an information Message as $C_3 = AES_{K_{AB}} (T_B\|TS_2\|Message)$. And then $N_A$ feeds back this reply message $R_3 = \text{Reply}\|C_3$.

(4) After obtaining $R_3$ from $N_A$, $N_B$ could decrypt $C_3$ using $K_{BA}$. If $T_B$ is consistent with the sent one, $N_A$ could be authenticated by $N_B$. $N_B$ can also confirm that a key agreement has been reached with $N_A$.

The man-in-the-middle attacker cannot modify $T_A$ or $T_B$, because the encrypted packets could only be decrypted by the other party’s private key. The mutual authentication between $N_A$ and $N_B$ is achieved. A series of symmetric keys appropriate can be generated by applying the Key Derivation Function (KDF) to $K_{AB}$ [53]. The secure communication between $N_A$ and $N_B$ is also established. At last, for the fake of security, the ephemeral secrets are all erased.

$N_A$ adds the authenticated neighbor nodes to the list of trust neighbors after completing all the authentications with its neighbors, i.e., $TN_A = \{ID_B, ID_F\}$. It also adds the failed nodes to the list of untrust neighbors, i.e., $UN_A = \{ID_C, ID_G\}$.

FIGURE 7: The node mutual authentication scheme.
4.6. Secure Topology Discovery. After each node completes its neighbor discovery, it creates its neighbor list. Different from traditional WSNs, SDWSNs are capable of global network management. To achieve this goal, the Controller periodically launches the TD procedure and collects the topology information to manage the network. As a state-of-the-art TD protocol, the Controller of SDN-WISE periodically broadcasts TD packets via the sinks to their neighbor nodes, which initiate a flood of TD packets to discover the rest nodes. Compared to the existing TD protocols, SDN-WISE can help sensor nodes to establish their surviving neighbor nodes’ information and the next hop toward the Controller [18]. Therefore, we design a secure TD mechanism based on the TD protocol of SDN-WISE. The security depends on the trust neighbor list created by the neighbor discovery process.

The TD packet contains the identity Sink_i, the broadcast time, a signature, a battery level, and the current hops [18].

\[ TD = ID_{Sink} \parallel Time \parallel \sigma_{Sink} \parallel Battery \parallel Hops, \]  

where \( \sigma_{Sink} = S_{Sink} (ID_{Sink} \parallel Time) \). (4)

The CC-IBS can be used to realize authentication and message integrity. Therefore, the freshness of this TD process is characterized by the broadcast time, and the authenticity of the TD packet is proved by the signature of broadcast time \( \sigma_{Sink} \). After receiving a new TD packet from \( N_B \) (note that B can also be a sink), \( N_A \) does the following operations:

1. \( N_A \) first checks whether \( N_B \) is in its \( TN_A \). If yes, \( N_A \) updates \( N_B \) along with the current RSSI and the battery level. Otherwise, it adds \( N_B \) to the list of unauthenticated neighbors. And \( N_B \) then discards this packet.
2. \( N_A \) only participates in the TD process of sinks in its trusted sink list. If the TD packet from Sink_i is received for the first time, \( N_A \) checks the time and the signature to ensure the authenticity of this message. If \( T = V_{Sink} (ID_{Sink} \parallel Time, \sigma_{Sink}) \), it discards this packet. Otherwise, it adds the Sink_i to its trust sink list.
3. \( N_A \) sets its next hop toward the Controller equal to \( N_B \) if \( N_B \) has a close path to the sink.
4. \( N_A \) adds one to the current hop value and broadcasts the updated TD packet over the wireless channel.

\( N_A \) only communicates with the authenticated neighbor nodes and sinks, which ensures the security of TD. If necessary, \( N_A \) can authenticate the uncertified new nodes after completing the TD process. The node may receive multiple TD packets from different sinks and it just selects the nearest one.

4.7. Node Admission and Key Establishment. On the basis of secure TD, all nodes can establish a secure route to the Controller at the end of the TD process. Further, each node needs to join the network and establish a secure link with the Controller for secure communication, such as sending routing requests when communicating with an unfamiliar node. Node admission is a process that allows new nodes to be authenticated and trusted by the sinks and Controller [13]. Whether it is to send topology reports safely or use security services, secure channels must be established with the Controller and the selected sink [39]. To protect end-to-end communication security, our modified IBAKE protocol could achieve authentication and key establishment between two remote nodes, such as the sink and the node.

4.7.1. Key Establishment. At the end of the TD phase, each node has ready received the TD packets from the sinks and created a trusted sink list. As shown in Figures 4 and 5, to establish a secure channel between \( N_A \) and its selected sink, the steps can be summarized as follows:

1. \( N_A \) sends an authentication request packet with an ephemeral key \( T_A \).
2. Intermediate secure nodes relay the request packet to the sink.
3. The sink recognizes the node, generates its ephemeral key \( T_B \), calculates the symmetric key \( SK \), and sends back a response packet including a MAC.
4. Intermediate secure nodes relay the response packet to \( N_A \).
5. \( N_A \) authenticates the response packet and generates the same symmetric key. The MAC is used to confirm that the shared symmetric key is correctly calculated. Then \( N_A \) sends a symmetrically encrypted reply packet to the sink.
6. Intermediate secure nodes relay the reply packet to the sink.
7. The sink receives the packet, checks for the authenticity of \( N_A \), and confirms the completion of the key agreement.

Nowadays, \( N_A \) confirms that it can reach the Controller through the sink and send secured data to the sink. The sink adds \( N_A \) to the list of secure nodes to prevent receiving forged topology messages from the unauthorized nodes.

4.7.2. Node Admission. After the authentication with the sink node is completed, the new node needs to finish the mutual authentication and establish a secure channel with the Controller. The authentication process is similar to the sink nodes and the detailed process is omitted here. If successful, the Controller and node authenticate each other. \( N_A \) is allowed to enter the network and could enforce security services. The Controller inserts \( N_A \) into its database. Only such nodes are authorized to participate in the security-enabled routes and request security services.
In the above two processes, we can observe that the intermediate nodes play a major role in the secure forwarding of topology packets. Benefiting from the secure neighbor node discovery mechanism, the malicious neighbor nodes would be isolated from the network and only the authenticated node can forward the packets. A valid route to the Controller is guaranteed via the secure TD mechanism. Finally, the secure links between the nodes and sinks, as well as the nodes and Controller have been established.

4.8. Secure Topology Collection. In order to control the underlying nodes, the Controller needs to carry out a topology collection process to collect the topology information of each node, such as node status and communication links between nodes. In an insecure network environment, the Controller collects malicious topology reports from unauthenticated nodes, which poses a serious threat to network management. Since $N_A$ has already been authenticated by the Controller, they will share the uniform symmetric key $SK_{AB}$. After finishing the TD process, each node periodically sends a Topology Report (TR) packet to the sink and Controller. The TR packet includes its trust neighbor list $TNA$, untrust neighbor list $UNA$, and current state. Considering the flat and hierarchical network structures, topology collection includes the secure topology report and topology aggregation processes, respectively.

4.8.1. Secure Topology Report. For the SDWSN with a flat network structure, such as SDN-WISE [18], each node only needs to forward the TR message to the next-hop node. Taking into account different security requirements, there are three methods to report topology information.

1. No encryption: $N_A$ directly sends the TR message without any cryptographic operations. This has the smallest calculation overhead but also the lowest security.

2. Authenticity and integrity: by generating the MAC of the TR packet, $N_A$ only needs to ensure that its TR packet is not tampered with:
   - (i) $TR_A = Current\_State_A||TNA||UNA||TS$
   - (ii) $MAC_A = HMAC_{SK_{AB}}(TR_A)$
   - (iii) $Message_A = TR||MAC_A$

3. Authenticity, integrity, and confidentiality. $N_A$ further encrypts the message to avoid eavesdropping:
   - (i) $TR_A = Current\_State_A||TNA||UNA||TS$
   - (ii) $MAC_A = HMAC_{SK_{AB}}(TR_A)$
   - (iii) $Message_A = AES_{SK_{AB}}(TR_A)||MAC_A$

Considering the potential security risks of the above E&M (encryption and MAC) approach, a safer authenticated encryption with associated data (AEAD) scheme can be adopted. AEAD is a form of encryption that simultaneously assures the confidentiality, integrity, and authenticity of data. The AEAD algorithms, such as AES-128-GCM and ChaCha20-IETF-Poly1305, can be employed if the software or hardware environment supports them.

4.8.2. Secure Topology Aggregation. In WSN, energy-efficient data collection strategies are important [54]. As shown in Figure 8, for the SDWSN with a hierarchical network structure, (e.g., ETMRM [20] and Sec-SWSN [11]), the cluster head is responsible for collecting and aggregating the messages of cluster nodes and delivering them to the Controller. Note that the question of how to elect the cluster heads is beyond the scope of this paper. A fuzzy attribute-based joint integrated scheduling and tree formation (FAJIT) technique for tree formation and parent node selection using fuzzy logic in a heterogeneous network will help to solve this problem [54]. But the cluster head election confirmation mechanism is supported and explained in Section 4.9.

Since the cluster heads are the neighbor nodes of $N_A$, they share the same symmetric key $SK$. $N_A$ can safely forward the TR message to its cluster head just like the secure topology report mechanism as:

\[ ID_A \rightarrow ID_{Cluster}: Topology\_Report. \] (5)

After receiving the TR packets from its members, the cluster head first verifies the messages, aggregates its own report with its members’ legitimate reports, and forwards them to the Controller as:

\[ ID_{Cluster} \rightarrow ID_{Controller}: Aggregated\_Topology\_Report. \] (6)

4.8.3. Secure Topology Analysis. The Controller decrypts the encrypted report messages using the corresponding $SK_{AB}$. During the transmission, the security of the TR packets can be guaranteed. On this basis, the Controller can establish the global network topology and the connectivity of nodes according to the underlying topology information.

Due to the symmetric encryption algorithm, the entire process of topology collection will not introduce too much computation overhead. The secure topology report mechanism is relatively simple and easy to implement, while the secure topology aggregation mechanism will greatly reduce
the transmission overhead. A suitable method can be chosen according to different network scales.

4.9. Secure Topology Management. During the operation of wireless sensor networks, network nodes are not fixed. Some nodes shut down due to depleted battery resources. Some nodes may be captured and become malicious. The network manager also deploys some new nodes. Topology management can preserve the integrity of network connectivity during network running [10]. The Controller can add new nodes, remove the malicious nodes, and announce the election of the cluster heads based on the global network topology. We put forward a CC-IBS based message authentication mechanism to defend against the tamper attack. The integrity and authenticity of topology management messages would be protected.

4.9.1. Node Removal. All nodes are legitimate at deployment time, but some may be captured and corrupted at run time and initiate malicious behavior, such as greyhole and blackhole attacks. Using certain trust management solutions, the internal attackers could be found by the nodes and reported to the Controller [20]. In addition, network security detection methods based on artifical intelligence, such as LSTM and CNN, are also very helpful for SDWSNs to discover network attackers [55, 56]. To the alarm report message, \( N_A \) signs its digest using \( S_A \) and the CC-IBS algorithm, and sends it to the Controller along with its signature as:

(i) \( \text{Message}_\text{Alarm} = ID_E || \text{Behavior\_Description} || \text{TS} \)
(ii) \( \sigma = S_A(\text{Hash} (\text{Message}_\text{Alarm})) \)
(iii) \( ID_A \rightarrow ID_{\text{Controller}} : \text{Message}_\text{Alarm} || \sigma \)

After confirming the alarm messages, the SDWSN Controller can learn about the underlying security situation. With the help of anomaly detection algorithms, the Controller can identify the malicious nodes and then generate the node removal messages containing the identities of the malicious nodes, a timestamp, operations, and a signature. The topology management information will be broadcasted to the sinks.

(i) \( \text{Message}_\text{Removal} = \text{Attacker\_Type} || ID_E || ID_G || \text{Operations} || \text{TS} \)
(ii) \( \sigma = S_{\text{Controller}}(\text{Hash} (\text{Message}_\text{Removal})) \)
(iii) \( ID_{\text{Sink}} \Rightarrow ID_\star : \text{Message}_\text{Removal} || \sigma \)

\( N_A \) would check the signature to make sure that the control message is not modified when the broadcast packet is received as:

\( ID_A : \top \) or \( \bot = V_{ID_{\text{Controller}}} (\text{Message}_\text{Removal}, \sigma). \) \hfill (7)

\( N_A \) would add \( ID_E \) and \( ID_G \) to the blacklist if the control packet is legitimate. The flow rules related to the malicious nodes are all removed.

4.9.2. Node Addition. During the running of the SDWSNs, some nodes may run out of power and the SDWSNs need to deploy additional nodes to the network. At the same time, certain sinks may be deployed when network congestion occurs. In order to reduce the communication and TD overhead, the Controller can sign the identity of these newly added nodes and broadcast their identities to the whole network as:

(i) \( \text{Message}_\text{Addition} = \text{Sink} || ID_{\text{H}} || ID_{\text{F}} || \text{Node} || ID_{\text{F}} || ID_{\text{E}} || \text{TS} \)
(ii) \( \sigma = S_{\text{Controller}}(\text{Hash} (\text{Message}_\text{Addition})) \)
(iii) \( ID_{\text{Sink}} \Rightarrow ID_\star : \text{Message}_\text{Addition} || \sigma \)

The correctness of the signature is verified by the CC-IBS algorithm and each node will add the legitimate nodes into its trust sink list or node list as:

\( ID_A : \top \) or \( \bot = V_{ID_{\text{Controller}}} (\text{Message}_\text{Addition}, \sigma). \) \hfill (8)

When secure communication is required, an authenticated key agreement process can be subsequently initiated between them.

4.9.3. Cluster Head Election Announcement. For the hierarchical network structure, announcing the elected cluster head is an important topology management operation under the centralized management of SDWSNs [57]. When the cluster heads are selected by the Controller, the Controller can sign the election messages using the CC-IBS algorithm and send them to the cluster heads safely. Each cluster head can broadcast this message to its neighbor nodes as:

(i) \( \text{Message}_\text{Cluster} = \text{Cluster} || ID_{\text{Cluster}} || \text{TS} \)
(ii) \( \sigma = S_{\text{Controller}}(\text{Hash} (\text{Message}_\text{Cluster})) \)
(iii) \( ID_{\text{Cluster}} \Rightarrow ID_\star : \text{Message}_\text{Cluster} || \sigma || \text{Current\_Status} \)

Upon receipt of messages, each node verifies the signatures of these messages and joins the cluster head with the highest remaining energy around it as:

(i) \( ID_A : \top \) or \( \bot = V_{ID_{\text{Controller}}} (\text{Message}_\text{Cluster}, \sigma) \)
(ii) \( ID_A \rightarrow ID_{\text{Cluster}}, \text{Join\_In} \)

5. Security Analysis

We analyze the security of our mechanism from five aspects including mutual authentication, forward secrecy, data security, attack defense, and security property comparison to verify that the mechanism meets the security requirements.

5.1. Mutual Authentication. At each step of the node authentication process, IBE is used with the public key of the receiver to guarantee that only the intended recipient of the message can decrypt the message. Therefore, the \( aP \) is encrypted by \( N_B \)'s public key and can only be decrypted by \( N_B \)'s private key \( S_B \). \( N_A \) can ensure that it communicates with the real \( N_B \) when it receives the correct \( aP \). Equally, \( N_B \) can verify the authenticity of \( N_A \) through the returned \( bP \).
Finally, the mutual authentication between $N_A$ and $N_B$ is achieved. Thanks to the IBAKE mechanism and Diffie–Hellman key agreement protocol, the two participants also agree on the key material used to secure future communication.

5.2. Forward Secrecy. The forward secrecy property is that even if $N_A$ and $N_B$’s private keys are compromised, the previously used keys should not be recovered [58]. In the key establishment process, the adversary needs to compute $abP$ from $aP$ and $bP$ even if the private keys are compromised. This is a CDH problem. In other words, given $aP$ and $bP$, it is computationally hard to compute $abP$, so the accidental leakage of present private keys does not compromise the past keys.

5.3. Topology Message Security. Our proposed schemes can protect the information security of topology messages. Confidentiality ensures that information is kept secret from everyone except those who are allowed to see it. Integrity ensures that messages have not been altered by unauthorized persons. Authentication ensures that the communicating party is the correct party to its claim [46]. As discussed in Section 4.2.2, the encryption component of IBCES is indistinguishably against IND-IBCES-CCA2 and the signature component is existential unforgeability against EUF-IBCES-CMA in the random oracle model. Therefore, the Cha and Cheon’s IBS and Boneh and Franklin’s IBE are securely used with the same key pair [46]. Furthermore, our encrypted communications and message signatures are secure under the same identity.

5.3.1. Data Confidentiality. We use the shared symmetric key $K_{AB}$ between two SDWSN nodes to encrypt the topology report message. The topology messages are transmitted in ciphertext format which can prevent unauthorized nodes from accessing sensitive data. The symmetric key $K_{AB}$ is usually not available to the adversary. If the adversary can guess $K_{AB}$, it will be able to break the CDH problem.

5.3.2. Data Integrity. The MAC derived from the shared symmetric key $K_{AB}$ is added to the end of the message. The digest of topology alarm messages is signed by the reported node. Using these methods, we can ensure that messages are transmitted without being modified by the malicious attackers.

5.3.3. Authentication. The data integrity and source authentication are guaranteed by the HMAC mechanism. The Controller sign the topology management messages and the SDWSN nodes only accept the correct topology management messages. These can protect the messages from impersonation attack.

5.4. Attack Defense

5.4.1. Man-in-the-Middle Attack. The man-in-the-middle attacker between $N_A$ and $N_B$ may obtain the transmitted messages from the sender. However, without knowing the corresponding private keys $S_A$ or $S_B$, it can not eavesdrop, tamper with, or forge the messages without attracting the attention of the receiver. Although an attacker might change the plaintext of the message (such as ID), the modified message is dropped because it does not match the identities in the encrypted message. As a result, the man-in-the-middle attack can be resisted by our secure communication scheme.

5.4.2. Replay Attack. A replay attacker replies with a message to the receiver, pretending that the original sender sends the message again. In our scheme, this replay attack is avoided by a timestamp $TS$. The replay attacker can not decrypt and change the value of $TS$ due to that $TS$ in the message is encrypted by the sender. The topology packets whose $TS$ stays at the same time would be dropped by the receiver.

5.4.3. Sybil Attack. Malicious nodes can take on multiple identities to launch Sybil attacks. Our mutual authentication scheme does not allow Sybil attackers to join the network. By authenticating the identities of each node and sink, the SDWSN node only communicates with the entities in its list of trusted sinks and nodes. As a result, this attack is no longer viable.

5.4.4. Controller DoS Attack. As the network intelligence is centralized in the Controller, the attacker can send lots of illegal packets, exhausting the resource of the Controller and crashing the whole network. In our scheme, the sinks and cluster heads only accept the packets from the authenticated nodes and also check the timestamp to defend the replay packets. The flooding packets from malicious nodes will be discarded at the sinks and cluster heads. As a consequence, our scheme can weaken the Controller DoS attack to a certain extent.

5.4.5. Internal Attack. Captured nodes can be corrupted and launch internal attacks, such as selective forwarding attacks. While our mechanisms can defend against external attacks, internal attackers can be detected by trust management schemes, which are outside our scope. However, each node communicates separately from the sink and Controller. The compromised node does not affect communication with other nodes. Therefore, our alarm reporting mechanism helps the Controller to discover and clear attackers in the SDWSN network.

5.4.6. Node Capture Attack. Because the sensor nodes are usually deployed in unattended environments and are not equipped with tamper-resistant hardware. Therefore, the sensor nodes are easy to be captured by adversaries, leading to typical node capture attacks in user authentication schemes for WSNs. The captured sensor node will reveal sensitive information about other noncompromised sensor nodes or affect previous communications. As discussed in Section 5.2, our scheme satisfies the forward secrecy property which means even if the private key of a sensor node is compromised, the previously used keys could not be recovered. Therefore, our scheme can resist the node impersonation attack.

5.5. Security Property Comparison. Table 2 compares our scheme with the previous works on wireless network security according to the node and message authentication, key
<table>
<thead>
<tr>
<th></th>
<th>Node auth</th>
<th>Message auth</th>
<th>Key agreement</th>
<th>Secure neighbor and topology discovery</th>
<th>Secure topology management</th>
<th>Cryptosystem</th>
<th>Forward secrecy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chakrabarty et al. [59]</td>
<td>Yes</td>
<td>Yes</td>
<td>—</td>
<td>Not discussed</td>
<td>Not discussed</td>
<td>Symmetric</td>
<td>None</td>
</tr>
<tr>
<td>Salman et al. [60]</td>
<td>Mentions</td>
<td>Not discussed</td>
<td>Not discussed</td>
<td>Not discussed</td>
<td>Not discussed</td>
<td>PKI certificate</td>
<td>Not informed</td>
</tr>
<tr>
<td>de Toledo et al. [39]</td>
<td>Mentions</td>
<td>Mentions</td>
<td>Mentions</td>
<td>Not discussed</td>
<td>Not discussed</td>
<td>Implicit certificate</td>
<td>Yes</td>
</tr>
<tr>
<td>Alves et al. [13]</td>
<td>Mentions</td>
<td>Mentions</td>
<td>Mentions</td>
<td>Mentions</td>
<td>Not discussed</td>
<td>Implicit certificate</td>
<td>Yes</td>
</tr>
<tr>
<td>Prasad and Salman Ali [42]</td>
<td>Yes</td>
<td>Not discussed</td>
<td>Yes</td>
<td>Not discussed</td>
<td>Not discussed</td>
<td>IBC</td>
<td>None</td>
</tr>
<tr>
<td>McCusker and O’Connor [28]</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>IBC</td>
<td>None</td>
</tr>
<tr>
<td>Wei et al. [61]</td>
<td>Yes</td>
<td>Yes</td>
<td>None</td>
<td>Not discussed</td>
<td>Not discussed</td>
<td>IBC</td>
<td>None</td>
</tr>
<tr>
<td>Li et al. [27]</td>
<td>Yes</td>
<td>None</td>
<td>Yes</td>
<td>—</td>
<td>—</td>
<td>IBC</td>
<td>Yes</td>
</tr>
<tr>
<td>Ours</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>IBC</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Mentions, but not detailed; —, this property is not considered.
agreement, secure topology control mechanisms, cryptosystem, and forward secrecy. Alves et al. [13], de Toledo et al. [39], and Chakrabarty et al. [59] mainly detect the authenticity and integrity of packets through MAC or AEAD. In contrast, we also use a digital signature mechanism in addition to symmetric cryptography. Although McCusker and O’Connor [28] and Prasad and Salman Ali [42] implement IBC-based key agreement mechanisms, none of their proposed mechanisms are forward-secure. The symmetric keys used in the past would be recovered if the private keys are compromised. Compared to other research, our work is the only one that fulfills all the properties at the same time.

6. Performance Analysis

In this section, we first analyze the computation and communication cost of our topology control scheme and then evaluate the performance on different platforms with different security levels. The notations used in this section are listed in Table 3.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_h$</td>
<td>Computation cost of map-to-point operation in $G_1$</td>
</tr>
<tr>
<td>$C_p$</td>
<td>Computation cost of pairing operation on the bilinear map</td>
</tr>
<tr>
<td>$C_m$</td>
<td>Computation cost of point multiplication in $G_1$</td>
</tr>
<tr>
<td>$C_e$</td>
<td>Computation cost of exponentiation operation in $G_2$</td>
</tr>
<tr>
<td>$</td>
<td>G_1</td>
</tr>
<tr>
<td>$</td>
<td>m</td>
</tr>
</tbody>
</table>

The computation cost of the sender is the same as that of the receiver during the node authentication process. The node authentication process does not bring much computation overhead to the node and usually happens at the deployment phase. The computation cost of the verifier is higher than that of the signer during the message authentication process. However, the message authentication processes usually occur at the topology management phase, such as the detection of malicious nodes, some new nodes joining the network, or the selection of new cluster heads, which does not burden the SDWSN nodes. In fact, the cryptosystem generates some communication overhead, and the communication cost of the node authentication process is higher than that of the message authentication process. Considering that the operation of $C_h$ is relatively time-consuming, the ID and corresponding public key can be cached after the two entities have communicated, omitting the map-to-point operation to reduce overhead.

6.2. Performance Test. Our secure topology control mechanism is implemented by the PBC library [62] (we employ PBC-0.5.14) and OpenSSL library [63] (we employ OpenSSL-1.1.1d) for both the SDWSN nodes and the Controller. The PBC library is a free portable library that allows the rapid prototyping of pairing-based cryptosystems. OpenSSL is used for symmetric encryption and MAC calculation. We use the medium-class WSN device, Raspberry Pi 4B, to simulate each SDWSN node. It is equipped with 1.5-GHz Cortex-A72 CPU, 2 GB RAM, and Raspberry Pi OS with Linux kernel 5.4.42. The Controller is placed on a desktop with a 3.4-GHz Intel Core i7-6700 CPU, 8 GB RAM, and Ubuntu 18.04. We use Type A pairings of PBC constructed from the curve $y^2 = x^3 + x$ over the field $\mathbb{F}_q$ for some $q = 3 \mod 4$. We conduct experiments with three security levels 80-, 112-, and 128-bit which are in line with different type A curve parameters [64].

6.2.1. Comparison between the Controller and Node. Before executing the topology control scheme, we measured the execution time of the basic PBC cryptographic operations on both devices. The results are listed in Table 5. We can observe that the Raspberry Pi node is slower than the Controller, up to 7.4 times in a certain scenario. The security level has a significant impact on the running time and a higher security level results in an increased workload on both
devices. Users can choose the different security parameters according to their needs, such as 80-bit for legacy security or 128-bit for modern security. In terms of cryptographic operations, the map-to-point operation \((H_1\text{ function})\) that converts a hash value into a group element takes the most time. Nevertheless, it runs no more than 128 ms, which is acceptable. Besides, this operation will only be used when two nodes communicate for the first time to generate an IBC public key. The identities and public keys of other nodes can be stored to save calculation time.

To test the feasibility of IBCES on the Controller and the nodes, we test the performance of cryptographic algorithms with different security levels and the results are illustrated in Figure 9. The setup and exact algorithms are performed only on the Controller. Encryption takes a longer time than decryption and verification consumes more time than signing. From 80 to 128 bits, the basic operations become time-consuming in large groups of elliptic curve elements, and therefore the execution time has become longer [29]. For the Controller, the exact algorithm takes the longest time because it needs to do a map-to-point operation to generate the public key. For the sensor node, the verification takes the longest time because its two pairing operations take up part of the time. But compared with other types in PBC, the pairing operation of Type A works fastest. The verification time under the strongest security is around 120 ms, which is still within reasonable values.

6.2.2. Overhead of Node and Message Authentication. Based on the above tests, we evaluate the computation and communication cost of the two types of authentication mechanisms involved in the topology control process. Node authentication is mainly used in the neighbor node discovery and node admission processes. Message authentication consists of node signing and verification and is mainly used in the secure TD and topology management processes. The computation costs for the authentication process are shown in Figure 10. They are first tested under the scenario where two nodes communicate for the first time. It can be observed that even with the 128-bit security level, the time of the first node authentication process (node auth-first) is less than 680 ms and the time of the first message authentication process (message auth-first) is around 300 ms, both are acceptable. It can also be observed that the total time-cost of node authentication (node auth) and message authentication (message auth) both become much smaller in subsequent communications.

We assume the ID length is 10 bytes and use the compression method to reduce the size of \(G_1\). The communication cost for each authentication process is shown in Figure 11. For instance, when taking the 128-bit key size of AES security level, the size of \(p\) is equal to 3,072 bits, and \(|G_1|\)
can be reduced to 193 bytes. Then the node authentication process costs extra 1,656 bytes. By contrast, at the same security level, the size of RSA certificate is 1,667 bytes and ECC certificate is 839 bytes. To complete the same function, the authentication overhead is greatly reduced. Relatively speaking, the communication cost of the message authentication process is less. Its communication overhead is twice the size of $|G_1|$ and consumes at most 386 bytes and 130 bytes at least.

6.2.3. Time-Cost of the First Topology Discovery. Compared with other topology control processes, the first TD process requires a message verification process, which takes a certain amount of time. Therefore, to make a clear understanding of our topology control scheme, we measure the time-cost of the first TD process by varying the number of delivery hops. As depicted in Figure 12, the delivery time rises linearly as the number of hops increases. Similarly, with the improvement of the security level, the delivery time of TD also increases. Nevertheless, in the case of the highest security level, the time-cost of a medium-scale network (up to 7 hops) is less than 1,800 ms, which is still within a reasonable value. The subsequent TD process will become faster as the identities of the legal sinks are cached. Specifically, the verification overhead of the cluster head announcement is the same as the delivery time when the number of hops is equal to 1. These experimental results indicate that our proposed topology control mechanism is fit for the SDWSN scenario.

6.3. Performance Analysis

6.3.1. Comparison with Other Authentication Schemes. Authentication is one of the most important security needs of WSNs. The traditional cryptographic approaches are infeasible to resource-constrained sensor nodes due to their high memory, communication, and computational overheads [17]. End-to-end authentication schemes for constrained WSN nodes using IBC and elliptic curve cryptography (ECC) have been suggested. The security strength of ECC depends on the strength of the elliptic curve discrete logarithm problem (ECDLP). Compared with traditional public key cryptography, ECC requires a shorter key size and has better computational efficiency. Although ECC is emerging as a promising security solution for WSN, ECC-based two-party authenticated key agreement protocols still use PKI and require heavy computation and management of public key certificate [65]. By contrast, IBC can provide authentication without using expensive certificates and digital signatures [65].

6.3.2. Comparison with Other IBAKE Schemes. At present, in the field of SDWSN topology control, most of the research focuses on energy-efficient routing, efficient TD protocol, trusted routing, etc., while ignoring the considerations related to cryptography [66, 67]. In the first communication scenario, we compare our work with typical identity-based authenticated key exchange mechanisms (IBAKE), although they were not applied to the SDWSN topology control domain directly. Several mechanisms can be used to achieve topology control and may have less computation, such as [28, 30] and [53]. However, they do not satisfy the forward secrecy requirement and have certain security risks. The shared secret key of Oliveira et al. [53] is calculated by $K_{AB} = e(S_A, Q_B) = e(Q_A, S_B)$. If the private key of node A or node B is compromised, the attacker who has stored the early data can compute the secret key and decrypt the early data. Although Rathore et al. [30] generate the ephemeral secrets randomly and McCusker and O’Connor [28] adopt the signature mechanism, respectively, they both have the same problem. The shared keys of Oliveira et al. [53] and McCusker and O’Connor [28] are fixed and the previous information will also be leaked when the keys are compromised.
By contrast, Wei et al. [61] and our work both support IBC encryption and signature mechanisms at the same time. However, the time cost of Wei et al. [61] is higher than that of ours. This is because it involves multiple signature verification operations which are time-consuming. Besides, the access authentication of Wei et al. [61] is a key transmit mechanism in which the initiator directly generates the session key and securely transmits it to the responder. It also has the forward secrecy issue. Our work uses a key agreement mechanism in which the two parties jointly participate in key generation through the exchange of parameters. Overall, our work is more secure and can achieve more convenient network management. Of course, there are drawbacks to our experiments. Limited by the number of experimental nodes, we can experiment with up to seven hops. If conditions permit, we can emulate a larger network. At the same time, simulation experiments can also be conducted on more types of nodes.

7. Conclusion

Topology control is a fundamental procedure of SDWSNs to maintain the global topology view of the network. However, the open wireless channels make it possible for the attacker to eavesdrop, replay, or modify the topology messages, thus posing a great threat to the SDWSN network. It is necessary to eavesdrop, replay, or modify the topology messages, thus the open wireless channels make it possible for the attacker to perform a great threat to the SDWSN network. It is necessary to strengthen SDWSN topology control’s security. As such, to secure the confidentiality, integrity, and authentication of topology information, we propose a secure topology control mechanism based on IBACCS. Mutual authentication with authenticated key agreement scheme is adopted to achieve authentication and secure communication between the neighbor nodes. Then the end-to-end secure communication among the node, sink, and Controller is established through a node admission with a key establishment process. At last, the basic information security needs are respectively satisfied by BF-IBE and CC-IBS during the topology collection and management processes. Theoretical analysis shows that our methods could satisfy the basic security needs of SDWSN topology control and resist several security attacks, such as man-in-the-middle attack, reply attack, and Sybil attack. The experimental results indicate that our mechanisms are suitable for SDWSNs.

At present, Raspberry Pi 4B nodes are mainly used in our experiment, and there are not too many kinds of network nodes actually deployed. In future work, simulation experiments can also be conducted on more types of nodes. We plan to investigate a lightweight IBC framework and carry out the feasibility tests on the low-class WSN platforms, e.g., MICAz motes, which are more resource-constrained. Furthermore, signcryption can achieve the signature and encryption simultaneously. To get a lower cost, we will conduct an in-depth study of the IBC cryptosystem and intend to introduce signcryption into our topology control mechanism.

Data Availability

No data were used to support this study.

Disclosure

A preliminary version of this paper was presented at the 15th International Conference on Wireless Algorithms, Systems, and Applications [68].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is sponsored by the project of State Grid Shandong Electric Power Company Science and Technology Program, Project Name: Research on Key Technologies of Smart Grid 5G Secure Access and Trusted Data Sharing—Topic 1: Research on Key Technologies of Smart Grid 5G Terminals and Network Security Detection and Risk Assessment, ERP number: 520626220016.

References


