A Bilinear Pairing-based Dynamic Key Management and Authentication for Wireless Sensor Networks

Chin-Ling Chen¹, Tzay-Farn Shih¹*, Yu-Ting Tsai¹ and De-Kui Li²

¹Department of Computer Science and Information Engineering, Chaoyang University of Technology, Taichung, Taiwan 41349, ROC.
²Department of Logistics Management, Wuhan Technology and Business University, Wuhan, 430065, Hubei, China
Correspondence should be addressed to Tzay-Farn Shih; tfshih@mail.cyut.edu.tw

Copyright 2014 Chin-Ling Chen 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.

In recent years, wireless sensor networks have been used in a variety of environments; a wireless network infrastructure, established to communicate and exchange information in a monitoring area, has also been applied in different environments. However, for sensitive applications, security is the paramount issue. In this paper, we propose a using bilinear pairing to design dynamic key management and authentication scheme of the hierarchical sensor network. We use the dynamic key management and the pairing-based cryptography (PBC) to establish the session key, and the hash message authentication code (HMAC) to support the mutual authentication between the sensors and the base station. In addition, we also embed the capability of the Global Positioning System (GPS) to cluster nodes to find the best path of the sensor network. The proposed scheme can also provide the requisite security of the dynamic key management, mutual authentication and session key protection. Our scheme can defend against impersonation attack, replay attack, wormhole attack and message manipulation attack.

Index Terms—sensor network; mutual authentication; hierarchical network; bilinear pairing; key management.

1. Introduction

In recent years, wireless sensor networks have been used in a variety of environments; a wireless network infrastructure, established to communicate and exchange information in a monitoring area, has also been applied in different environments, including disaster relief operations, seismic data collecting, monitoring wildlife and collect battlefield information.

Due to their small size, the sensors can be spatially scattered to form an ad hoc network. The sensors have an inherent limitation. The wireless sensor network requires an appropriate encryption or decryption system to protect the collected information [1]. The high cost of an encryption/decryption mechanism (for example: Diffie-Hellman key management [2] or Rivest Shamir Adleman encryption [3]) is unsuitable for use in a wireless sensor network.

In addition, the topology of the network environment is another important issue. The hierarchical pre-distribution protocol [4] allows some of the cluster nodes to aggregate the events of the sensor nodes to communicate with the base station. The hierarchical pre-distribution protocol includes several cluster nodes, sensor nodes and base station; the most common hierarchical networks are two-level, and the two classes of sensor are: sensor node and cluster node. The advantage of this scheme is the easy management of the data aggregation [5-7]. The process of aggregating the data from multiple nodes involves eliminating redundant transmission and providing fused data to the base station. It is also considered as an effectual technique for wireless sensor networks to save energy [8]. The most popular data aggregation algorithms are cluster-based data aggregation algorithms, in which the nodes are grouped into clusters: each cluster consists of a cluster node and some sensors; each sensor transmits data to its cluster node; and each cluster node aggregates the collected data; it then transmits the fused data to the base station.

The key management scheme is divided into four types: the Random Key Pre-distribution Protocol (RKP), the Group-based Key Pre-distribution Protocol (GKP), the Hierarchical Key Pre-distribution Protocol (HKP), and the Pairing-based Protocol (PBC). In 2003, Chan et al. proposed a Random Key Pre-distribution scheme [9]. Since each node randomly picks keys from a large key pool such that any two sensor nodes share at least one common key, ensuring adequate storage space and the range of the network is a challenge. The PIKE scheme [10] addressed the problem of high density deployment requirements in RKP. But in this scheme, the session key is segmented into many key fragments. Therefore, the combination of the session key is complex. However, the PIKE solved the storage problem of RKP. Cheng and Agrawal proposed an improved key distribution mechanism [11]. The IKDM established a session key which used the exchange
information between sensors; it can easily generate a session key by the polynomial function. In recent years, the pairing-based cryptography [12], TinyPBC, is a tiny pairing-based protocol and its computation cost is lower than other corresponding bilinear pairing-based schemes. The pairing-based mechanism was used in the sensor network to accomplish the key management of the sensor’s session key. It can use the sensors’ identity for sensors to send data to each other via the sensor network. After the identity exchange, the sensors key can easily compute the session key via the bilinear pairing. In such design, the security can also be enhanced.

In this paper, we propose a using bilinear pairing to design dynamic key management and authentication scheme of the hierarchical sensor network. We use the dynamic key management mechanism [13,14] and the pairing-based cryptography (PBC) [12,15,16] to establish the session key. We also use the hash message authentication code (HMAC) [15,17] to offer mutual authentication between the sensors and the base station. Moreover, we also involve the capability of the Global Positioning System (GPS) [18,19] to cluster nodes, in order to find the best path of the sensor network.

The remainder of this paper is organized as follows. The preliminaries are presented in Section 2. The proposed scheme is described in Section 3. The security analysis of our scheme is given in Sections 4. And the discussions are offered in Section 5. Finally, conclusions are presented in the last section.

2. Preliminaries

A. Sensor network architecture

Categories of sensor networks that significantly affect key establishment design [4]. The relative capabilities of different sensors. The kind of sensors are divided into the following two classes:

1. Homogeneous: all sensors have the same capabilities.
2. Heterogeneity: there is an inherent hierarchy of sensors with respect to their capabilities (with fewer sensors at higher, more “powerful” levels). The most common hierarchical networks are two-level, where there are two classes of sensor.

We choose the hierarchical sensors network’s model, the architecture is described as follows: a small number high class sensors (cluster node), large number low class sensors (sensor node) and a sink node (base station). High class sensors have more powerful ability, it has equipped with tamper-resistant hardware and GPS capability, the cluster node with powerful ability can plan routing table and achieve more security of sensor network. And the low class sensors do not equip with tamper-resistant hardware and GPS capability.

B. Bilinear pairing

The bilinear map can be constructed on elliptic curves. Each operation for computing \( e(P, Q) \) is a pairing operation [8]. Let \( G \) be a cyclic additive group, and \( G_T \) a cyclic multiplicative group. Both groups \( G \) and \( G_T \) have the same prime order \( q \). The groups \( G \) and \( G_T \) are called bilinear groups. The security of the bilinear pairing based scheme relies on the difficulty of the discrete logarithm problem (DLP), i.e.,

given the point \( Q = aP \), no efficient algorithm exists to obtain \( a \) given \( P \) and \( Q \). The mapping \( e: G \times G \to G_T \) is called a bilinear map if it satisfies the following properties:

1. Bilinear:
   \[ e(Q, P + R) = e(P, Q) \cdot e(R, Q) \]

2. Non-degenerate:
   \[ P, Q \in G \text{ exists such that } e(P, Q) \neq 1_{G_T} \]

3. Computable:
   An efficient algorithm exists to compute \( e(P, Q) \) for any \( P, Q \in G \).

C. Hash Message Authentication Code (HMAC)

We combine the Message Authentication Code [20,21] and the bilinear pairing key to accomplish the Hash-based Message Authentication Code (HMAC); this is a specific construction for computing a Message Authentication Code (MAC) using a cryptographic hash function in combination with a secret key. Both data integrity and authenticity of a message can be achieved by using a hash-based message authentication code in such a technique. We note HMAC (ie. \( H_K(\cdot) \) ) is a HMAC which signifies a one way hash function with pairing key \( K \).

D. Pairing-Based Cryptography (PBC)

Since Pairing-Based Cryptography (PBC), based on the Identity-Based Cryptography (IBC) [22,23], is used in many environments of cryptographic protocols and applications [12]. The IBC has some drawbacks; this method needs a Private Key Generator (PKG); it is a trusted entity in charge of generating and escrow user's private keys. In wireless sensor networks, if the sensors need to be deployed in an unattended environment, a sensor node should be a PKG, it is difficult in a wireless sensor network. If we can easily generate a session key via a simple mechanism, it can reduce the complexity. PBC technology does not need a PKG and the sensors can authenticate itself in the wireless sensor network. Therefore, the PBC is the best technology for key management.

3. The proposed scheme

In this paper, we propose a bilinear pairing-based to design a dynamic key management for wireless sensor network. We first introduce the proposed protocol architecture as the Figure 1.

1. Base station broadcasts the starting message to cluster nodes.
2. Cluster nodes respond the message authentication code to the base station.
3. After authentication, the base station sends a response message to allow cluster nodes to rule its group members of the sensor nodes.
4. Cluster node broadcasts the request message to find the members from the neighboring sensor nodes.
5. Sensor nodes reply the request, and respond the message authentication code to the cluster node.
6. In order to get the sensor nodes’ session key, if the cluster node can transmit to the base station enter into step 6.1; else if the cluster node needs to transmit the collected information via the next neighboring; enter into step 6.2.

7. After authentication, the base station sends the corresponding session key of sensor nodes to the cluster nodes.

8. After receiving the session keys, the cluster nodes can verify the message authentication code from the step 5. After that, the cluster nodes send the updated identities to the sensor nodes.

![Fig. 1. The architecture of wireless sensor network](image)

### 3.1 Notations

The following is the introduction to the notations that will be used in our scheme.

- **SN**<sub>i</sub>: the *i*th sensor node
- **CN**<sub>i</sub>: the *i*th cluster node
- **BS**: the base station
- **G**: a cyclic additive group which has the same prime order *q*
- **GT**: a cyclic multiplicative which has the same prime order *q*
- **e(·,·)**: pairing operation *e : G × G → GT*
- **P**<sub>SN</sub><sub>i</sub>: the identity of the *i*th sensor node
- **P**<sub>CN</sub><sub>i</sub>: the identity of the *i*th cluster node
- **P**<sub>BS</sub>: the identity of the base station
- **s, r**: an integer number of secret parameter generated by the base station
- **P**<sub>PUB</sub>: a public parameter, *P**<sub>PUB** = s · P*
- **SP**<sub>1SN</sub>, **SP**<sub>CN</sub>: a secret parameter using a secret number *S* to compute the secret parameter for the *i*th sensor node and cluster node respectively
- **SP**<sub>2CN</sub>, **SP**<sub>CN</sub>: a secret parameter using a secret number *r* to compute the secret parameter for the *i*th cluster node and the *j*th cluster node respectively
- **α, β**: an integer of the secret parameter generated by the cluster node and the base station respectively
- **K**<sub>SN</sub><sub>i</sub>: a session key of the *i*th sensor node
- **K**<sub>P1</sub>: a key pool generated by the base station, *(K**<sub>SN1</sub>, K**<sub>SN2</sub>, ..., K**<sub>SNn</sub>)*
- **K**<sub>P2</sub>: a key pool generated by the base station, *(K**<sub>CN1</sub>, K**<sub>CN2</sub>, ..., K**<sub>CNn</sub>)*
- **Klist**<sub>SN</sub>: a key list, *(Klist**<sub>SN1</sub>, Klist**<sub>SN2</sub>, ..., Klist**<sub>SNn</sub>)*
- **Klist**<sub>CN</sub>: a key list, *(Klist**<sub>CN1</sub>, Klist**<sub>CN2</sub>, ..., Klist**<sub>CNn</sub>)*
- **SNID**<sub>list</sub>: the cluster node collects the sensor's identity to send to the base station
- **n**<sub>A</sub>: a nonce generated by A
- **MAC**: the message authentication code
- **X ? Y**: determine if *X* equal to *Y*
- **H()**: a one way hash function
- **H_K(·)**: a one way hash function with key *K*
- **E_K(M)**: using an asymmetric key *K* to encrypt message *M*
- **D_K(M)**: using an asymmetric key *K* to decrypt message *M*
- **C**<sub>i</sub>: the *i*th encrypted message
- **Msg**<sub>start</sub>: the starting message which is used to start the cluster node which is dominated by the base station
- **Msg**<sub>location</sub>: the location message
- **Msg**<sub>req</sub>: the request message generated by the cluster node to find the sensor node
- **Msg**<sub>rep</sub>: the response message generated by the sensor node to respond to the cluster node request
- **Msg**<sub>finish</sub>: the finished message
- **→••••**: a secure channel
- **→**: an insecure channel
### 3.2 Initialization phase

In this phase, the base station computes the parameters to pre-distribute into the sensor nodes and the cluster node. The overview of the initialization phase is shown in Figure 2.

![Fig. 2. The overview of the initialization](image)

![Table: Initialization phase parameters](table)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_{SN_i} = H(x) )</td>
<td>Sensor node identity ( P_{SN_i} ) computed from a random number ( x )</td>
</tr>
<tr>
<td>( SP_{SN_i} = sP_{SN_i} )</td>
<td>Secret parameter ( s ) and secret parameter ( sP_{SN_i} )</td>
</tr>
<tr>
<td>( K_{PL} = (K_{SN_1}, K_{SN_2}, \ldots, K_{SN_i}) )</td>
<td>Session key ( K_{PL} ) for cluster node ( CN_i )</td>
</tr>
<tr>
<td>( K_{LIST}^{SN} = (P_{SN_i}, K_{SN_i}) )</td>
<td>Key list for sensor node ( SN_i )</td>
</tr>
<tr>
<td>( P_{CN_i} = H(y) )</td>
<td>Cluster node identity ( P_{CN_i} ) computed from a random number ( y )</td>
</tr>
<tr>
<td>( SP_{1CN_i} = sP_{CN_i} )</td>
<td>Secret parameter ( s ) and secret parameter ( sP_{CN_i} )</td>
</tr>
<tr>
<td>( SP_{2CN_i} = rP_{CN_i} )</td>
<td>Secret parameter ( r ) and secret parameter ( rP_{CN_i} )</td>
</tr>
<tr>
<td>( K_{P2} = (K_{CN_i}, K_{CN_2}, \ldots, K_{CN_n}) )</td>
<td>Session key ( K_{P2} ) for cluster node ( CN_i )</td>
</tr>
<tr>
<td>( K_{LIST}^{CN_i} = (P_{CN_i}, K_{CN_i}) )</td>
<td>Key list for cluster node ( CN_i )</td>
</tr>
</tbody>
</table>

**Step 1:** First, the base station selects a random number \( x \) and computes the sensor node identity \( P_{SN_i} \):

\[
P_{SN_i} = H(x)
\]  

(1)

Then, the base station randomly selects a secret parameter \( s \) and uses the secret parameter \( s \) and sensor node identity \( P_{SN_i} \) to compute the secret parameter \( SP_{1SN_i} \):

\[
SP_{1SN_i} = sP_{SN_i}
\]  

(2)

The base station randomly computes a key pool \( K_{PL} \), where

\[
K_{PL} = (K_{SN_1}, K_{SN_2}, \ldots, K_{SN_i})
\]

and distributes a session key \( K_{SN_i} \) to the \( i \)th sensor node. It then stores the sensor node identity \( P_{SN_i} \) and the \( K_{SN_i} \) in the key list \( K_{LIST}^{SN} \):

\[
K_{LIST}^{SN} = (P_{SN_i}, K_{SN_i})
\]  

(3)

After that, the base station sends the parameters \( (SP_{1SN_i}, P_{SN_i}, K_{SN_i}) \) to the corresponding sensor node.

**Step 2:** The base station selects a random number \( y \) and computes the cluster node identity \( P_{CN_i} \):

\[
P_{CN_i} = H(y)
\]  

(4)

Then the base station randomly selects a secret parameter \( r \) and uses the random secret parameters \( (s, r) \) to compute the secret parameters \( (SP_{1CN_i}, SP_{2CN_i}) \) respectively:

\[
SP_{1CN_i} = sP_{CN_i}
\]  

(5)

\[
SP_{2CN_i} = rP_{CN_i}
\]  

(6)

The base station randomly computes a key pool \( K_{P2} \), where

\[
K_{P2} = (K_{CN_i}, K_{CN_2}, \ldots, K_{CN_n})
\]

and distributes a session key \( K_{CN_i} \) to the \( i \)th cluster node. It then stores the cluster node identity \( P_{CN_i} \) and the \( K_{CN_i} \) in the key list \( K_{LIST}^{CN_i} \):

\[
K_{LIST}^{CN_i} = (P_{CN_i}, K_{CN_i})
\]  

(7)

The base station sends the parameters \( (SP_{1CN_i}, SP_{2CN_i}, P_{CN_i}, K_{CN_i}) \) to the cluster node.

### 3.3 Location-based routing determination phase

1. The starting cluster node process

After the sensors are deployed, we must start the cluster node and get the path routing. In Figure 3, we authenticate the cluster node to confirm the legality of the cluster node. Next, the cluster node can rely on the location-based routing to find the best routing path.
Step 1: First, the base station selects a random number $r_1$ and broadcasts the message $(\text{Msg}_{\text{start}}, r_1)$ to the sensor network.

Step 2: When the cluster node $CN_i$ receives the message, it can select a random number $r_2$ and compute the message authentication code with the key $K_{CN_i}$:

$$MAC_{CN_i} = H_{K_{CN_i}}(r_1 || r_2 || \text{Msg}_{\text{start}})$$ (8)

The cluster node $CN_i$ then sends the message $(P_{CN_i}, r_2, MAC_{CN_i})$ to the base station.

Step 3: Upon receiving the message, the base station can use the identity $P_{CN_i}$ to find the key $K_{CN_i}$ from the $K_{list_{CN_i}}$:

$$K_{list_{CN_i}} = (P_{CN_i}, K_{CN_i})$$ (9)

It then computes the message authentication code $MAC'_{CN_i}$:

$$MAC'_{CN_i} = H_{K_{CN_i}}(r_1 || r_2 || \text{Msg}_{\text{start}})$$ (10)

And checks if it is equal to $MAC_{CN_i}$:

$$MAC_{CN_i} = MAC'_{CN_i}$$ (11)

3.4 Location-based routing phase

The cluster nodes can establish the best route on the basis of receiving the broadcast location message in a monitoring area.

Step 1: After the initialization phase, the sensor nodes and cluster nodes store the operating parameters and then distribute the associated messages within their monitoring environment.

Step 2: The base station broadcasts the starting message $\text{Msg}_{\text{start}}$ to the cluster nodes.

Step 3: Upon receiving the starting message, the cluster node (equipped with a GPS receiver) broadcasts the message $\text{Msg}_{\text{location}}$ concerning its location to the neighbor cluster nodes.

Step 4: After receiving the message $\text{Msg}_{\text{location}}$, the cluster nodes know the location of the source of the neighboring cluster such that it can transmit the monitoring data to the cluster node which is the nearest node to the base station.

For example, in Figure 4, the cluster node $R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8$ can receive the nearest distance message to the base station from the neighbor cluster nodes $R_3, R_2, R_6, R_1, R_5, R_7, R_8$ and $R_8$. It can compare the received location message to select the nearest node from the base station and establish the multi-hop routing path to the cluster node $R_1$. The cluster node $R_1$ will be used to relay communications to the base station, so the best path of the cluster node $R_3$ will be established as follows: $R_3 \rightarrow R_1 \rightarrow BS$. On the basis of the shortest distance between the cluster node and the base station, each cluster node will establish the best routing path.
path for the cluster node \( R_3 \) can be established as follows: \( R_2 \rightarrow R_3 \rightarrow R_1 \rightarrow BS \). In the same way, the cluster node \( R_3 \) can determine the best path: \( R_2 \rightarrow R_3 \rightarrow R_1 \rightarrow BS \). Every pair of nodes along the resulting multi-hop path can establish a pairwise key for encrypted communication in such a way that each intermediate node can relay data towards the base station in a totally secure way. Location awareness also increases the probability that the geographically closest node pairs establish a pairwise session key along the best path to the BS, with the effect of saving energy on all the nodes involved in multi-hop routing.

### 3.5 The authentication phase of the cluster node and the sensor node

The base station sends the broadcast message \( Msg_{\text{start}} \) to the cluster nodes, when the cluster node receives the message, it will broadcast the request message \( Msg_{\text{req}} \) to find the neighboring sensor node to join the group. The overview of the authentication phase of the cluster node and the sensor node is shown in Figure 6.

**Step 1:** When the cluster node \( CN_i \) receives the starting message \( Msg_{\text{start}} \), the cluster node \( CN_i \) selects a nonce \( n_1 \) and sends \((P_{CN_i}, n_1, Msg_{req})\) to the neighboring sensor nodes.

**Step 2:** Upon receiving the message, the sensor node \( SN_i \) selects a nonce \( n_2 \), and uses \((n_1 || n_2 || Msg_{\text{rep}})\) to compute the message authentication code \( MAC_1 \):

\[
MAC_1 = H_{K_{SN_i}}(n_1 || n_2 || Msg_{\text{rep}}) \tag{12}
\]

The sensor node \( SN_i \) sends \((P_{SN_i}, n_2, Msg_{\text{rep}}, MAC_1)\) to the \( i \)th cluster node \( CN_i \).

**Step 3:** The cluster node \( CN_i \) adds the sensor node's identity \( P_{SN_i} \) into the identity list \( SNID_{\text{list}} \):

\[
SNID_{\text{list}} = (P_{SN_1}, P_{SN_2}, ..., P_{SN_i}) \tag{13}
\]

It then sends the cluster node's identity and the sensor node's identity list \( SNID_{\text{list}} \) to the base station. If the cluster node is the nearest base station, then it directly enters into the next section 3.5, the authentication phase of the base station and the cluster node. Otherwise, the cluster node needs to transmit the collected information via the next neighboring cluster node and enters into section 3.6, the authentication phase of the cluster node and the cluster node.

After the authentication and obtaining the session key of the sensor node, the cluster node \( CN_i \) computes the

![Fig. 6. The overview of the authentication phase of the cluster node and the sensor node](image)
message authentication code $MAC'_1$, and checks if it is equal to $MAC_1$ or not:

$$MAC'_1 = H_{K_{SN_i}}(n_i || n'_2 || MSG_{rep})$$

(14)

$$MAC'_2 = MAC'_1$$

(15)

Then the cluster node $CN_i$ selects a random integer number $\alpha$, computes the new parameter

Fig. 7. The overview of the authentication phase of the base station and the cluster node

(\alpha_{P_{CN_i}}, \alpha_{P_{SN_i}}) and updates into $(P_{CN_i}, P_{SN_i})$, respectively:

$$P_{CN_i} = \alpha_{P_{CN_i}}$$

(16)

$$P_{SN_i} = \alpha_{P_{SN_i}}$$

(17)

The cluster node $CN_i$ uses the session key to encrypt the new parameter $P_{SN_i}$ of the sensor node $SN_i$:

$$C_1 = E_{K_{SN_i}}(P_{SN_i})$$

(18)

The cluster node $CN_i$ randomly selects a nonce $n_3$ and computes the message authentication code $MAC_2$:

$$MAC_2 = H_{K_{SN_i}}(n_3 || n'_2 || C_1)$$

(19)

Then the cluster node $CN_i$ sends the message $(P_{CN_i}, C_1, n_3, MAC_2)$ to the sensor nodes.

Step 4: The sensor node $SN_j$ computes the message authentication code $MAC'_2$ and checks if it is equal to $MAC_2$:

$$MAC'_2 = H_{K_{SN_j}}(n'_j || n_2 || C'_1)$$

(20)

$$MAC'_2 = MAC_2$$

(21)

After authentication, the sensor node $SN_j$ decrypts the encrypted message $C_1$:

$$P_{SN_j} = D_{K_{SN_j}}(C_1)$$

(22)

Then the sensor node updates the parameter $P_{SN_j}$:

$$P_{SN_j} = P'_{SN_j}$$

(23)

3.6 The authentication phase of the base station and the cluster node

In this phase, the cluster node sends the message to the base station to find the corresponding sensor node's session key. The overview of the authentication phase of the base station and the cluster node is shown in Figure 7.

Step 1: First, the cluster node $CN_i$ enters collects the sensor nodes identity $P_{SN_i}$ into the identity list $SNID_{list}$:

$$SNID_{list} = (P_{SN_1}, P_{SN_2}, ..., P_{SN_j})$$

(24)
Then the cluster node $CN_i$ uses the pairing function to compute the pairing session key $K_{CN_i-BS}$:

$$K_{CN_i-BS} = e(SP_{2-CN_i}, P_{BS})$$  \hspace{1cm} (25)

It computes the message authentication code $MAC_{CN_i-BS}$:

$$MAC_{CN_i-BS} = H_{K_{CN_i-BS}}(P_{CN_i} \parallel n_{CN_i} \parallel SNID_{list})$$  \hspace{1cm} (26)

Then the cluster node $CN_i$ sends the message $(P_{CN_i}, n_{CN_i}, SNID_{list}, MAC_{CN_i-BS})$ to the base station.

Step 2: After receiving, the base station uses the pairing function to compute the pairing session key $K_{RS- CN_i}$:

$$K_{RS- CN_i} = e(rP_{BS}, P_{CN_i})$$  \hspace{1cm} (27)

The base station computes the message authentication code $MAC_{CN_i-BS}$:

$$MAC_{CN_i-BS} = H_{K_{RS- CN_i}}(P'_{CN_i} \parallel n'_{CN_i} \parallel SNID_{list})$$  \hspace{1cm} (28)

It checks if it is equal to $MAC_{CN_i-BS}$:

$$MAC_{CN_i-BS} = MAC_{CN_i-BS}$$  \hspace{1cm} (29)

After authentication, the base station uses the identity list $SNID_{list}$ to find the corresponding session key $K_{SN_i}$, makes the key list $Klist_{SN_i}$ and enters it into $SNK_{CN_i}$:

$$SNK_{CN_i} = (P_{SN_i}, K_{SN_i})$$  \hspace{1cm} (30)

$$SNK_{CN_i} = (Klist_{SN_i}, Klist_{SN_i}, ..., Klist_{SN_i})$$  \hspace{1cm} (31)

The base station randomly selects a nonce $n_{BS}$ and computes the message authentication code $MAC_{BS-CN_i}$:

$$MAC_{BS-CN_i} = H_{K_{BS-CN_i}}(n_{BS} \parallel n'_{CN_i} \parallel P_{BS})$$  \hspace{1cm} (32)

Then, the base station uses the pairing session key to encrypt the sensor node's session key list $SNK_{CN_i}$:

$$C_2 = E_{K_{BS-CN_i}}(SNK_{CN_i})$$  \hspace{1cm} (33)

The base station sends the message $(P_{BS}, n_{BS}, C_2, MAC_{BS-CN_i})$ to the corresponding cluster node $CN_i$.

Step 3: When the cluster node $CN_i$ receives the message, it computes the message authentication code $MAC_{BS-CN_i}$:

$$MAC'_{BS-CN_i} = H_{K_{BS-CN_i}}(n_{BS} \parallel n_{CN_i} \parallel P_{BS})$$  \hspace{1cm} (34)

It checks if it is equal to $MAC_{BS-CN_i}$:

$$MAC'_{BS-CN_i} = MAC_{BS-CN_i}$$  \hspace{1cm} (35)

Then, the cluster node $CN_i$ decrypts the encrypted message $C_2$:

$$SNK_{CN_i} = D_{K_{CN_i-BS}}(C_2)$$  \hspace{1cm} (36)

In this phase, the cluster node $CN_i$ obtains the sensor node's session key $K_{SN_i}$ and finishes the mutual authentication with the base station.

3.7 The authentication phase of the $i$th cluster node and the $j$th cluster node

When the cluster cannot directly transmit the message to the base station, it will enter into this phase. The overview of the authentication of the cluster node and the cluster node is shown in Figure 8.

Step1: The cluster node $CN_i$ computes the pairing session key $K_{CN_i-CN_j}$ and randomly selects a nonce $n_1$ to compute the message authentication code $MAC_{CN_i-CN_j}$:

$$K_{CN_i-CN_j} = e(SP_{2-CN_j}, P_{CN_i})$$  \hspace{1cm} (37)

$$MAC_{CN_i-CN_j} = H_{K_{CN_i-CN_j}}(P_{CN_i} \parallel n_1 \parallel Msg_{collect})$$  \hspace{1cm} (38)

Then, the cluster node $CN_i$ sends the message $(P_{CN_i}, n_1, Msg_{collect}, MAC_{CN_i-CN_j})$ to the cluster node $CN_j$.

Step 2: When receiving the message, the cluster node $CN_j$ computes the pairing session key $K_{CN_j-CN_i}$:

$$K_{CN_j-CN_i} = e(SP_{2-CN_i}, P_{CN_i})$$  \hspace{1cm} (39)

Then, the cluster node $CN_j$ uses the pairing session key $K_{CN_j-CN_i}$ to compute the message authentication code $MAC_{CN_j-CN_i}$ and checks if it is equal to $MAC_{CN_j-CN_i}$:

$$MAC'_{CN_j-CN_i} = MAC_{CN_j-CN_i}$$  \hspace{1cm} (40)

$$MAC'_{CN_j-CN_i} = MAC_{CN_j-CN_i}$$  \hspace{1cm} (41)
The cluster node \(CN_j\) randomly selects a nonce \(n_2\) and computes the message authentication code \(MAC_{CN_j-CN_i}^j\):

\[
MAC_{CN_j-CN_i}^j = H_{K_{CN_j-CN_i}}(P_{CN_j} \ || n_1 || n_2)
\] (42)

The cluster node \(CN_j\) sends the message \((P_{CN_j}, n_2, MAC_{CN_j-CN_i}^j)\) to the cluster node \(CN_i\).

Step 3: After receiving the message \((P_{CN_j}, n_2, MAC_{CN_j-CN_i}^j)\) the cluster node \(CN_i\) computes the message authentication code \(MAC_{CN_i-CN_j}^i\) and checks if it is equal to \(MAC_{CN_j-CN_i}^j\) as follows:

\[
MAC_{CN_i-CN_j}^i = H_{K_{CN_i-CN_j}}(P_{CN_i} \ || n_1 || n_2)
\] (43)

\[
MAC_{CN_i-CN_j}^i \neq MAC_{CN_j-CN_i}^j
\] (44)

4. Security analysis

4.1 Mutual authentication

1. The authentication between the cluster node and the sensor node

(1) The cluster node authenticates sensor node

In the authentication phase of the cluster node and the sensor node, when the sensor node \(SN_j\) receives the message \((P_{SN_j}, n_1, Msg_{req}, Msg_{loc})\), the sensor node \(SN_j\) selects a nonce \(n_2\), and uses \((n_1 || n_2 || Msg_{req})\) to compute the message authentication code \(MAC_1\):

\[
MAC_1 = H_{K_{SN_j}}(n_1 || n_2 || Msg_{req})
\] (12)

After the authentication of the base station and cluster node, the cluster node \(CN_i\) obtains the session key of the sensor nodes. The cluster node \(CN_i\) computes the message authentication code \(MAC_1\), and checks if it is equal to \(MAC_1:\)

\[
MAC_1 = H_{K_{SN_j}}(n_1 || n_2 || Msg_{req})
\] (14)

\[
MAC_1 \neq MAC_1
\] (15)

(2) The sensor node authenticates the cluster node

The cluster node \(CN_j\) uses the session key to encrypt the new parameter \(P_{SN_j}\) of the sensor node \(SN_j:\)

\[
C_1 = E_{K_{SN_j}}(P_{SN_j})
\] (18)

The cluster node \(CN_i\) randomly selects a nonce \(n_3\) and computes the message authentication code \(MAC_2:\)

\[
MAC_2 = H_{K_{SN_j}}(n_1 || n_2 || C_1)
\] (19)

Then the cluster node \(CN_i\) sends the message \((P_{CN_i}, C_1, n_3, MAC_2)\) to the sensor nodes.

When the sensor node \(SN_j\) receives the message, it computes the message authentication code \(MAC_2\) and checks if it is equal to \(MAC_2:\)
\[ MAC'_2 = H_{K_{SN_1}} (n_3 \| n_2 \| C'_1) \] (20)
\[ MAC'_2 \oplus MAC'_2 \] (21)

Therefore, our schemes achieve the mutual authentication between the cluster node and sensor node.

2. The authentication between the \( i \)th cluster node and the \( j \)th cluster node

(1) The \( j \)th cluster node authenticates the \( i \)th cluster node

In the authentication phase of the cluster node and the cluster node, the cluster node \( CN_j \) computes the pairing session key \( K_{CN_i-CN_j} \), and randomly selects a nonce \( n_1 \) to compute the message authentication code \( MAC_{CN_i-CN_j} \) as follows:

\[ K_{CN_i-CN_j} = e(SP_2^{CN_i}, P_{CN_j}) \] (37)
\[ MAC_{CN_i-CN_j} = H_{K_{CN_i, CN_j}} (P_{CN_j} \| n_1 \| MSG_{collect}) \] (38)

Then it sends the message \( (P_{CN_j}, n_1, MSG_{collect}, MAC_{CN_i-CN_j}) \) to the sensor nodes.

Upon receiving the message \( (P_{CN_j}, n_1, MSG_{collect}, MAC_{CN_i-CN_j}) \), the cluster node \( CN_i \) computes the pairing session key \( K_{CN_i-CN_j} \):

\[ K_{CN_i-CN_j} = e(SP_2^{CN_j}, P_{CN_j}) \] (39)

Then, the cluster node \( CN_j \) uses the pairing session key \( K_{CN_i-CN_j} \) to compute the message authentication code \( MAC'_{CN_i-CN_j} \) and checks if it is equal to \( MAC_{CN_i-CN_j} \) or not:

\[ MAC'_{CN_i-CN_j} = H_{K_{CN_i-CN_j}} (P_{CN_i} \| n'_1 \| MSG_{collect}) \] (40)
\[ MAC'_{CN_i-CN_j} \oplus MAC_{CN_i-CN_j} \] (41)

(2) The \( i \)th cluster node authenticates the \( j \)th cluster node

Upon receiving the message \( (P_{CN_j}, n_1, MSG_{collect}, MAC_{CN_i-CN_j}) \), the cluster node \( CN_j \) randomly selects a nonce \( n_2 \) and computes the message authentication code \( MAC_{CN_j-CN_i} \):

\[ MAC_{CN_j-CN_i} = H_{K_{CN_j, CN_i}} (P_{CN_j} \| n_1 \| n_2) \] (42)

Then it sends the message \( (P_{CN_j}, n_2, MAC_{CN_j-CN_i}) \) to the cluster node \( CN_i \).

When the cluster node \( CN_i \) receives the message, it computes the message authentication code \( MAC_{CN_i-CN_j} \) and checks if it is equal to \( MAC_{CN_i-CN_j} \):

\[ MAC'_{CN_j-CN_i} = H_{K_{CN_j, CN_i}} (P_{CN_j} \| n'_1 \| n'_2) \] (43)
\[ MAC'_{CN_j-CN_i} \oplus MAC_{CN_j-CN_i} \] (44)

Therefore, our scheme achieve mutual authentication among the cluster nodes.

3. The authentication between the base station and the cluster node

(1) The base station authenticates the cluster node

In the authentication phase of the base station and the cluster node, when the cluster node \( CN_i \) receives the message \( (P_{CN_i}, n_{CN_i}, SNID_{list}, MAC_{CN_i-BS}) \), the cluster node \( CN_i \) uses the pairing function to compute the pairing session key \( K_{CN_i-BS} \):

\[ K_{CN_i-BS} = e(SP_2^{CN_i}, P_{BS}) \] (25)

It computes the message authentication code \( MAC_{CN_i-BS} \):

\[ MAC_{CN_i-BS} = H_{K_{CN_i-BS}} (P_{CN_i} \| n_{CN_i} \| SNID_{list}) \] (26)

Then it sends the message \( (P_{CN_i}, n_{CN_i}, SNID_{list}, MAC_{CN_i-BS}) \) to the base station.

When the base station receives the message, the base station uses the pairing function to compute the pairing session key \( K_{BS-CN_i} \):

\[ K_{BS-CN_i} = e(rP_{BS}, P_{CN_i}) \] (27)

The base station computes the message authentication code \( MAC_{BS-CN_i} \):

\[ MAC_{BS-CN_i} = H_{K_{BS-CN_i}} (P_{CN_i} \| n_{CN_i} \| SNID_{list}) \] (28)

It checks if it is equal to \( MAC_{CN_i-BS} \):

\[ MAC'_{BS-CN_i} = MAC_{CN_i-BS} \oplus MAC_{CN_i-BS} \] (29)

(2) The cluster node authenticates the base station

When the base station receives the message \( (P_{CN_i}, n_{CN_i}, SNID_{list}, MAC_{CN_i-BS}) \), the base station randomly selects a nonce \( n_{BS} \) and computes the message authentication code \( MAC_{BS-CN_i} \):

\[ MAC_{BS-CN_i} = H_{K_{BS-CN_i}} (P_{CN_i} \| n_{CN_i} \| P_{BS}) \] (32)

Then it sends the message \( (P_{BS}, n_{BS}, C_2, MAC_{BS-CN_i}) \) to the base station.

Then, the cluster node \( CN_i \) receives the message and computes the message authentication code \( MAC_{BS-CN_i} \):

\[ MAC_{BS-CN_i} = H_{K_{BS-CN_i}} (n_{BS} \| n_{CN_i} \| P_{BS}) \] (34)

It checks if it is equal to \( MAC_{BS-CN_i} \):

\[ MAC'_{BS-CN_i} = MAC_{BS-CN_i} \oplus MAC_{BS-CN_i} \] (35)

Therefore, we complete the mutual authentication.

4.2 Dynamic key management

Our scheme offers random pairwise keys pre-distribution. After completing the information transmission, the cluster nodes and the sensor nodes update the session key for each session. It can prevent the replay attack. We divide into two
parts to analyze this process: the cluster node to sensor node, and the cluster node to cluster node.

1) The cluster node to sensor node

For example, if the sensor node $SN_i$ wants to communicate with the cluster node $CN_i$, it computes the dynamic key $K_{SN_i-CN_i}$:

$$K_{SN_i-CN_i} = e(SP_{SN_i}, P_{CN_i})$$  \hspace{1cm} (45)

Then, encrypts the collected data with the key $K_{SN_i-CN_i}$ and sends the encrypted message $E_{K_{SN_i-CN_i}}(M)$ to the cluster node $CN_i$.

Upon receiving the message, the cluster node $CN_i$ computes the session key $K_{CN_i-SN_i}$:

$$K_{CN_i-SN_i} = e(SP_{CN_i}, P_{SN_i})$$  \hspace{1cm} (46)

Then, decrypts the message $E_{K_{SN_i-CN_i}}(M)$ and gets the collected data.

After the transaction, the cluster node computes a new integer parameter $\alpha_{new}$ to update the identity of the cluster node and sensor nodes.

$$P_{CN_i} = \alpha_{new}P_{CN_i}$$  \hspace{1cm} (47)

$$P_{SN_i} = \alpha_{new}P_{SN_i}$$  \hspace{1cm} (48)

So, our scheme updates a new session key in each section.

2) The base station to cluster nodes

If the base station wants to update the session key, it can compute a new integer $r_{new}$ to generate a new secret parameter $SP^{\alpha_{new}}_{CN_i}$:

$$SP^{\alpha_{new}}_{CN_i} = r_{new}P_{CN_i}$$  \hspace{1cm} (49)

Then, encrypts the new secret parameter $SP^{\alpha_{new}}_{CN_i}$ with key $K_{BS-CN_i} = e(rP_{BS}, P_{CN_i})$, and sends to the corresponding cluster node $CN_i$. This mechanism can prevent the cluster node from being captured. If a malicious attacker gets the cluster node and intercepts the secret parameters in the sensor network, we can change the secret parameter via the base station.

4.3 Provides session key protection (elliptic curve discrete logarithm problem)

The security of our scheme relies on the difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP) concerning bilinear groups. We compute the parameter $SP_{SN_i}$, given the point $SP_{SN_i} = sP_{SN_i}$, it is difficult to obtain the secret parameter $s$ by given the secret parameter $SP_{SN_i}$ and the $P_{SN_i}$. If an attacker steals the transferred traffic information, the attacker cannot crack the session key to decrypt the ciphertext.

4.4 Impersonation attack

In the impersonation attack, if the attacker tries to steal the information between the sensors' communications, our scheme can defend against the information being used to conduct falsification, modification, replacement and re-transmission. In order to prevent the impersonation attack, the session key is generated by using mutual authentication. In the mutual authentication phase, we use $H_{K_{CN_i-CN_j}}(\cdot)$, and a one way hash function with key $K_{CN_i-CN_j}$ to implement message authentication; the key is difficult to crack and calculate. The related information is shown as follows:

$$K_{CN_i-CN_j} = e(SP^{2}_{CN_i}, P_{CN_j})$$  \hspace{1cm} (37)

$$MAC_{CN_i-CN_j} = H_{K_{CN_i-CN_j}}(P^{2}_{PUBLIC}) || n || Msg_{collect}$$  \hspace{1cm} (38)

So, the attacker cannot accomplish the impersonation attack.

4.5 Replay attack

For the replay attack, we use dynamic key management to update the session key in each transaction, we change the message authentication code in each section as follows: If an attacker tries to steal information to resend the same information to the target sensor node, it is impossible to pass the authentication.

For example, in each section, the sensor node uses $n_{1_{new}}$, $n_{2_{new}}$ and $Msg_{rep}$ to compute the new message authentication code $MAC_{1_{new}}$; the cluster node uses $n_{3_{new}}$, $n_{2_{new}}$ and $C_1$ to compute a new message authentication code $MAC_{2_{new}}$:

$$MAC_{1_{new}} = H_{K_{SN_i}}(n_{1_{new}} || n_{2_{new}} || Msg_{rep})$$  \hspace{1cm} (50)

$$MAC_{2_{new}} = H_{K_{SN_i}}(n_{3_{new}} || n_{2_{new}} || C_1)$$  \hspace{1cm} (51)

If the attacker uses the message authentication code, the verifiers can verify the legality as follows:

$$MAC \neq MAC_{1_{new}}$$  \hspace{1cm} (52)

$$MAC \neq MAC_{2_{new}}$$  \hspace{1cm} (53)

Therefore, the attacker cannot successfully achieve the replay attack.

4.6 Wormhole attack

In a wormhole attack, an attacker records a packet in one location of the network and sends it to another location, creating a tunnel between the attacker's nodes. The packet is retransmitted to the network under the attacker's control [24,25]. In the location-based routing determination phase of our scheme, the cluster nodes can establish the best route on the basis of the received broadcast location message in a monitoring area.
In Figure 9, if an attacker deploys a malicious cluster node \( A \), it can collect the message from the sensor nodes \( S_4, S_5, S_7 \) and successfully intercept the messages. But, in our schemes, we involve the starting cluster node process in the location-based routing determination phase to build the communication connections initially. Afterward, we also involve the mutual authentication between the cluster nodes and sensor nodes. So, the attacker cannot successfully complete the wormhole attack.

4.7 Message manipulation attack

In a message manipulation attack, an attacker may drop, modify, or even forge exchanged messages in order to interrupt the communication process [15].

In Figure 10, an attacker deploys a malicious cluster node \( A \) and forges a fake cluster node; the malicious node \( A \) can receive messages from the cluster nodes, and the attacker may drop, modify or even forge exchanged messages in order to interfere the normal communication process. If a malicious cluster node \( A \) wants to interfere with a path among \( C_1, C_2, C_4 \), the cluster node \( C_4 \) communicates with the malicious cluster node \( A \), it cannot pass the mutual authentication successfully, because it is difficult to compute the HMAC's key. Moreover, the routing path is established in the location-based routing determination phase. It is impossible for an attacker to interfere with the routing path and message.

5. Discussions

In Table 1, our scheme can prevent more attacks than other related schemes. In Table 2, the cluster node only needs to store 1 session key and 2 identity parameters; we can use the bilinear pairing function to calculate the session key between the clusters or the session key between the cluster and the base station. We use the GPS to support the path planning agreements, and use the location-based routing determination to build the network routing path. The dynamic key management protocol can update the session key to enhance the security.

The proposed scheme provides complete authentication. In Table 3, we make the computation cost of the session key agreement according four stages:

1. Sensor node to sensor node: TinyPBC is a tiny pairing-based protocol and the computation cost is lower than the bilinear pairing based protocol. In this stage, the scheme TinyPBC can use the cost \( 2T_n + 2T_p + 1T_E + 1T_D + 2T_A \) to generate a session key. We have more cost \( 2T_n + 2T_A \) than the TinyPBC scheme does. Our scheme inherits the advantage of the TinyPBC: we use the sensor level to build the hierarchical sensor network, i.e. we use TinyPBC's topology (sensor node to sensor node) to our scheme (cluster node to cluster node), it provides more powerful key management in WSN. It can also easily carry out message data aggregation and generate the session key between the cluster node and sensor nodes. So, our scheme can prevent more attacks, such as wormhole and message manipulation attacks.

---

**TABLE 1. THE COMPARISONS OF THE PREVENTION ATTACKS**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Against impersonation attack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Against replay attack</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Against wormhole attack</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Against message manipulation attack</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
The cost of the pairing-based cryptography is the same as the TinyPBC scheme, but our scheme has better performance and security.

(2) Cluster node to sensor node: According to the comparison of the KMTD. The sensor network is more convenient, complete and secure. In order to achieve more security and easy key management, we use the bilinear pairing to generate the session key. We need not use the encryption and decryption to generate the session key or the base station's help. The computation cost is reduced $3T_h + T_E + T_D$ and $2T_A + T_e + T_n$ added to help the session key generation. This method can defend against more attacks and also has the path planning agreement.

**TABLE 2. THE COST COMPARISON OF THE STORED KEY AND METHOD USED**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Stored cost (cluster node)</td>
<td>1SK+2IDs</td>
<td>N/A</td>
<td>2SKs+1ID</td>
<td>1SK+2IDs</td>
</tr>
<tr>
<td>Stored cost (sensor node)</td>
<td>2SKs+1ID</td>
<td>1SK+2IDs</td>
<td>2SKs+1ID</td>
<td>1SK+2IDs</td>
</tr>
<tr>
<td>New node algorithm</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Detail security analysis</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Group-based protocol</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sensor's homogeneity</td>
<td>homogeneity</td>
<td>homogeneity</td>
<td>hierarchical</td>
<td>hierarchical</td>
</tr>
<tr>
<td>GPS capability (cluster node)</td>
<td>No</td>
<td>No</td>
<td>hierarchical</td>
<td>hierarchical</td>
</tr>
<tr>
<td>Path planning agreements</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Dynamic key management</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

$SK$ : the session key.

$ID$ : the identity.

**TABLE 3. THE COMPUTATION COST OF THE SESSION KEY**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensor node to sensor node</td>
<td>N/A</td>
<td>$2T_h + 2T_p + 1T_E + 1T_D + 2T_n$</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Cluster node to sensor node</td>
<td>N/A</td>
<td>N/A</td>
<td>$7T_h + 2T_E + 2T_D + 2T_n$</td>
<td>$4T_h + 1T_a + 1T_E + 1T_D + 3T_n + 2T_A$</td>
</tr>
<tr>
<td>Cluster node to cluster node</td>
<td>$4T_h + 2T_p + nT_d + 1T_E + 1T_D$</td>
<td>N/A</td>
<td>N/A</td>
<td>$4T_h + 2T_p + 1T_E + 1T_D + 2T_n + 2T_A$</td>
</tr>
<tr>
<td>Cluster node to base station</td>
<td>N/A</td>
<td>N/A</td>
<td>$6T_h + 3T_E + 4T_D + 2T_n + 2T_A$</td>
<td>$4T_h + 2T_p + 1T_E + 1T_D + 2T_n + 2T_A$</td>
</tr>
</tbody>
</table>

$T_p$ : the time cost of a pairing operation
$T_a$ : the time cost of a Additive group $G$
$T_h$ : the time cost of a hash operation
$T_E$ : the time cost of a encryption
$T_D$ : the time cost of a decryption
$T_n$ : the time cost of generation a nonce
$T_{pf}$ : the time cost of the polynomial function
$T_d$ : the time cost of the combining key fragment degree
$T_A$ : the time cost of the authentication
(3) Cluster node to cluster node: According to the comparison of the IKDM, the computation cost is reduced $2T_{T'} + nT_d$ and in changing to the bilinear pairing $2T_p + 2T_A + 2T_a$. The polynomial function easily generates the session key between the cluster nodes. However, the IKDM generates a session key which is unsuitable for large scale sensor network. The construction methods of the session key need more key material of the cluster node to combine, so we chose the bilinear pairing to generate the session key in the cluster nodes and enhance security. It can more easily complete the session key.

(4) Cluster node to base station: According to the comparison of the KMTD, the computation cost is reduced $2T_h + 2T_e + 3T_d$ and $2T_p$ is added. We combine the Message Authentication Code and the bilinear pairing key to accomplish the HMAC. The security of our scheme relies on ECDLP; the attacker cannot compute the secret key, and this increases the security between the base station and the cluster node. The session key and the mutual authentication are generated by the bilinear pairing function. It has the characteristic of ECDLP; the attacker cannot compute the secret key and pass verification.

Based on these concepts, we use the hierarchical topology which has more power and can easily implement key management. We combine the Message Authentication Code and the bilinear pairing key to accomplish the message authentication.

6. Conclusion

We used bilinear pairing to design a dynamic key management and authentication of the hierarchical sensor network. We used the dynamic key management, pairing-based cryptography, hash message authentication code and the GPS capability’s cluster nodes to establish the secure agreement of the wireless sensor network. Our scheme achieves the following goals:

1. Proposed the dynamic key management to update the session key.
2. Overcomes the sensor node inherent limitations, we use the hierarchical network protocol in the wireless sensor network. It is more suitable for the large monitoring range in a wireless sensor network.
3. Provided the mutual authentication among the sensor nodes, cluster nodes and the base station.
4. Used the characteristics of the Discrete Logarithm Problem to generate the session key, so that its security could be enhanced.

Acknowledgements

This research was supported by the Ministry of Science and Technology, Taiwan, R.O.C., under contract number MOST 103-2221-E-324-023, MOST103-2632-E-324-001-MY3 and and Collaborative Innovation Center for Modern Logistics and Business of Hubei (Cultivation).

References


De-Kui Li was born in 1979. He received M.S., and Ph.D. degrees in Management Information System from Kwangwoon University, Seoul, South Korea, in 2012. He is currently a associate professor in the Department of Management, Wuhan Technology and Business University, Wuhan, Hubei, China. His research interests include business intelligence and data mining.

Chin-Ling Chen was born in Taiwan in 1961. He received the B.Sc. degree in Computer Science and Engineering from Feng Cha University in 1991; the M.Sc. degree and Ph.D. in Applied Mathematics at National Chung Hsing University, Taichung, Taiwan, in 1999 and 2005 respectively. He is a member of the Chinese Association for Information Security. From 1979 to 2005, he was a senior engineer at Chunghwa Telecom Co., Ltd. He is currently a professor in the Department of Computer Science and Information Engineering at Chaoyang University of Technology, Taiwan. His research interests include cryptography, network security, communication network and electronic commerce. Dr. Chen has published over 50 articles on the above research fields in SCI/SSCI international journals.

Tzay-Farn Shih received the Ph.D. degree in Electrical Engineering from National Taiwan University, Taiwan, in 2006. He is presently an assistant professor of Computer Science and Information Engineering at Chaoyang University of Technology, which he initially joined in August 2006. His research interests include computer simulation, computer networks routing protocol, wireless networks, mobile ad hoc networks and wireless sensor networks.

Yu-Ting Tsai was born in Taiwan in 1988. He received the B.S. degree in Department of Communication and Information Science from Tunghnan University of Technology, Taipei, Taiwan in 2011. He received his master’s degree at the Department of Computer Science and Information from Chaoyang University of Technology in 2013. His research interests include cryptography, especially in WSN model communication and session key agreement.