Research Article

A Lightweight Authenticated Key Agreement Protocol Using Fog Nodes in Social Internet of Vehicles

Tsu-Yang Wu, Xinglan Guo, Lei Yang, Qian Meng, and Chien-Ming Chen

1College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China
2Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin, China

Correspondence should be addressed to Chien-Ming Chen; chienmingchen@ieee.org

Received 2 September 2021; Accepted 27 October 2021; Published 17 November 2021

Copyright © 2021 Tsu-Yang Wu 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.

Recently, there has been rapid growth in the Internet of things, the Internet of vehicles, fog computing, and social Internet of vehicles (SIoV), which can generate large amounts of real-time data. Now, researchers have begun applying fog computing to the SIoV to reduce the computing pressure on cloud servers. However, there are still security challenges in SIoV. In this paper, we propose a lightweight and authenticated key agreement protocol based on fog nodes in SIoV. The protocol completes the mutual authentication between entities and generates the session key for subsequent communication. Through a formal analysis of the Burrows–Abadi–Needham (BAN) logic, real-oracle random (ROR) model, and ProVerif, the security, validity, and correctness of the proposed protocol are demonstrated. In addition, informal security analysis shows that our proposed protocol can resist known security attacks. We also evaluate the performance of the proposed protocol and show that it achieves better performance in terms of computing power and communication cost.

1. Introduction

With the popularization and development of the world wide web, the Internet of things (IoT) [1–3], which is a network of Internet extension and expansion, has emerged. With the continuous development of IoT applications, a “social network of intelligent objects” called social Internet of things (SIoT) [4] has been formed. Internet of vehicles (IoV) [5] is an extension of the concept of SIoT. IoV can realize network connections between the vehicle and vehicle (V2V), vehicle and infrastructure (V2I), and vehicle and pedestrian (V2P) and collect and share the key road information. With the rapid development of network and sensor technology, social connection in urban transportation systems is necessary, so social Internet of vehicles (SIoV) is produced [6–8]. SIoV is an application of SIoT in the field of vehicles and is a combination of vehicular ad hoc networks (VANET) and mobile networks, and it can generate a large amount of real-time data. In SIoV, intelligent vehicles can establish social relationships with other objects and form a specific social network.

For cloud computing processing of road real-time data, there are some problems associated with network delays, transmission efficiency, and others. Because the distance between the cloud computing server and vehicles is far, and the number of vehicles is increasing, the cloud server needs to process more real-time data, which increases the computing burden. Therefore, researchers have introduced fog computing to reduce the computational burden on cloud servers. The data, processing, and application of fog computing are stored on scattered and weak devices, almost outside the cloud, so the computing power is not strong. It can help the cloud server process some data that are not necessary or urgent at that moment. If it encounters data that it cannot process, it reports to the cloud server. Fog nodes can detect unsafe driving behavior in time, issue early warnings for the behavior, and provide the corresponding punishment when necessary. The application of fog node in IoT and IoV environments was mentioned in the articles [9–13]. In 2016, Azimi et al. [11] proposed a medical warning system in IoT based on fog computing. In 2019, Ismail et al. [12] proposed an implication of fog computing on the IoT.
In 2019, Ma et al. [10] proposed a protocol for fog-based IoV networks, which realized authenticated key agreement. In 2021, Eftekhari et al. [9] proposed a pairwise secret key agreement protocol using fog-based IoV, which was a three-part authentication protocol. The SIoV typical architecture based on fog nodes is shown in Figure 1.

However, in the SIoV environment based on fog nodes, there are still great risks related to security issues. For example, it is very challenging to ensure the confidentiality and privacy of data transmission based on ensuring the security of devices deployed on the network edge. The data transmitted through the public channel usually includes sensitive information such as the personal information of vehicle users, which needs to be kept secret. Recently, Ahmed et al. [6] researched a key agreement protocol for V2G in the SIoV environment, which was a two-party authentication protocol. The protocol [6] was based on an elliptic-curve (ECC) point multiplication and had a large computational cost. This shortcoming leads us to propose a more effective protocol.

We propose a lightweight and authenticated key agreement protocol based on three parties using fog nodes in an SIoV environment. In this protocol, vehicles and fog nodes authenticate each other with the help of a cloud server (CS) and establish a secure session key. Owing to the weak computing power of fog nodes, our protocol only uses lightweight primitives, such as hash function and XOR operation. Through formal analysis of the Burrows–Abadi–Needham (BAN) logic, real-oracle random (ROR) model, and ProVerif, the security, validity, and correctness of the proposed protocol are demonstrated. In addition, informal security analysis shows that our proposed protocol can resist known security attacks. We also evaluate the performance of the proposed protocol and show that it has better performance in terms of computing power and communication cost.

The rest of the paper is structured as follows: in Section 2, we review recent research results. The details of our proposed agreement are in Section 3. In Section 4, we use BAN, ROR, and ProVerif to verify the security, validity, and correctness of the proposed protocol. In addition, we conduct an informal security analysis. In Section 5, we compare our method with other protocols in terms of performance and security. Finally, we summarize this paper in Section 6.

2. Related Work

IoV is an open network environment, so this feature may threaten the identity information and relevant sensitive data of vehicle users. For many years, researchers have proposed many protocols to protect the privacy of vehicle users in IoV environments. In 2006, Raya et al. [14] proposed a vehicle communication protocol that stored multiple public and private key pairs and protected the privacy of vehicle users through the certificates stored in OBU. However, in 2008, Lu et al. [15] determined that the protocol [14] had high computing and storage cost because the key was changing at times and proposed a privacy protection protocol for vehicle communication. That same year, Zhang et al. [16] proposed an identity verification protocol for IoV. The protocol [16] realized privacy protection by the tamper-proof device to generate a random pseudoidentity. In 2020, Cui et al. [17] researched a privacy-preserving scheme. The protocol [17] was based on edge computing and used lightweight primitives, such as elliptic-curve cryptography, instead of bilinear pairing-based primitives with high computational cost. Later, Hu et al. [18] proposed a privacy-preserving authentication scheme for IoV.

The protocols proposed by some researchers have high computing power. In 2014, Li et al. [19] proposed a protocol that provided PKC-based privacy protection for IoV and claimed that their protocol could resist replay and stolen smart card attacks. However, Amit et al. [20] revealed that Li et al.’s protocol [19] was susceptible to key compromise impersonation attacks and could not provide user anonymity. To reduce high computing cost caused by the use of PCK in the above protocol, the Trust-Extended Authentication Mechanism (TEAM) protocol was proposed [21]. In 2016, Kumari et al. [22] proposed an authentication protocol that also used TEAM. In 2017, Ying and Nayak [23] proposed an effective and lightweight protocol for an IoV environment, which could provide user anonymity. Chen et al. [5] demonstrated that [23] was vulnerable to replay and offline identity guessing attacks. Therefore, to solve the vulnerability of Ying and Nayak’s protocol [23], Chen et al. [5] proposed a secure authentication scheme for IoV. However, the protocol [5] stored extensive data in the database, so it had high storage cost. In the same year, Mohit et al. [24] proposed an efficient authentication protocol for vehicular systems and deemed their protocol safe. However, Yu et al. [25] pointed out that the protocol [24] of Mohit et al. was susceptible to impersonation attacks and could not provide anonymity, traceability, and mutual authentication. Then, Yu et al. [25] proposed an authenticated protocol in
vehicular communications. In 2020, Sadri et al. [26] demonstrated that Yu et al.’s protocol [25] was susceptible to sensor capture attacks and impersonation attacks and could not provide traceability. Additionally, Sadri and Rajabzadeh Asaar [26] proposed a protocol in the IoV environment, which was based on lightweight primitives. In 2021, Wu et al. [27] proposed a protocol in IoV, and the protocol realized authentication key exchange (AKE).

There are increasingly more vehicles in the IoV environment, and data processing and transmission have become an inevitable challenge. Therefore, researchers began to apply cloud computing to IoV to solve the problem of processing a large amount of data to improve authentication efficiency. In an IoV environment, an authentication scheme based on cloud computing had been widely mentioned and applied in articles [28–31]. For an environment using cloud computing, problems such as network delay and transmission efficiency would exist, and the cloud server would need to process more data, which would increase the computing burden of the cloud server. Therefore, researchers have begun to introduce fog nodes for fog computing to share the pressure of cloud servers. In these papers [10–13], fog computing technology was applied. Ma et al.’s protocol [10] applied fog computing to IoV and proposed an authenticated key agreement protocol. They claimed that the protocol [10] was secure and efficient, but Eftekhar et al. [9] pointed out that Ma et al.’s protocol [10] was vulnerable to internal attacks, stolen smart card attacks, and known session-specific temporary information attacks. Therefore, Eftekhar et al. [9] proposed a more efficient authentication protocol. In 2021, Wu et al. [32] proposed a secure scheme using fog nodes in IoV, and the protocol realized AKE. In the same year, Maria et al. [33] proposed a blockchain-based anonymous authentication scheme, which used bilinear pairing. Some important related works are summarized in Table 1.

3. The Proposed Protocol

In this part, we introduce a lightweight and authenticated key agreement protocol using fog nodes in SIoV. Our protocol is based on the architecture of Figure 1. The protocol includes three entities: vehicle \( V_i \), fog node \( FN_j \), and CS. The symbols used in the protocol are shown in Table 2. The protocol has three phases: vehicle registration phase, fog node registration phase, and login authentication phase.

3.1. \( V_i \) Registration Phase. In the \( V_i \) registration phase, \( V_i \) registers with CS. The phase is shown in Figure 2, and the specific steps are as follows:

1. First, \( V_i \) selects its identity \( ID_i \), password \( PSW_i \), and a random number \( r_i \), calculates its pseudoidentity 
   \[ PID_i = h(ID_i \| r_i) \]
   and then transmits the \( PID_i \) to CS through the secure channel.

2. After receiving the message from \( V_i \), CS calculates the value of 
   \[ HID_i = h(PID_i \| K_{CS}) \]
   initializes the value of \( K_V \) to 0, and stores \{ \( HID_i, K_V \) \} in its database. Finally, CS sends \{ \( HID_i, K_V \) \} to \( V_i \).

3. After receiving the message from CS, \( V_i \) calculates the value 
   \[ a_i = HID_i \oplus h(PSW_i \| r_i) \]
   with the value of \( a_i \), and stores the \{ \( a_i, P_i, r_i, K_V \) \} in its smart card.

3.2. \( FN_j \) Registration Phase. In \( FN_j \) registration phase, \( FN_j \) registers with CS. The phase is shown in Figure 3, and the specific steps are as follows:

1. First, \( FN_j \) selects its identity \( FID_j \) and a random number \( r_j \), calculates its pseudoidentity 
   \[ PFID_j = h(FID_j \| r_j) \]
   and then transmits \{ \( PFID_j, FID_j \) \} to CS through the secure channel.

2. After receiving the message from \( FN_j \), CS first selects a random number \( K_{FN} \), calculates the value of 
   \[ N_j = h(FID_j \| K_{CS}) \oplus R_j, K_{FN} = h(PFID_j \| K_{CS}), HID_j = h(FID_j \| K_{CS}), \]
   and stores \{ \( PFID_j, FID_j, K_{FN}, HID_j \) \} in its database. Finally, CS sends \{ \( K_{FN}, HID_j, N_j, ID_{CS} \) \} to \( FN_j \).

3. After receiving the message from CS, \( FN_j \) calculates the value 
   \[ R_j = h(FID_j \| ID_{CS} \| N_j, \beta_j = HID_j \oplus h(R_j \| r_j) \]
   and stores the \{ \( K_{FN}, \beta_j, r_j, N_j \) \} in its database.

3.3. Login and Authentication Phase. In the login and authentication phase, \( V_i, FN_j \), and CS realize authentication and establish session key SK. This phase is shown in Figure 4, and the specific steps are as follows:

1. First, \( V_i \) inserts the smart card into the reader terminal, inputs its identity \( ID_j \), password \( PSW_j \), calculates the login authentication value 
   \[ P_i = h(ID_j \| PSW_j \| r_i) \]
   and then compares \( P_i = P_j \). If equal, \( V_i \) logs in successfully. Otherwise, the login fails. After successful login, \( V_i \) selects a random number \( N_j \), and calculates 
   \[ A_j = h(ID_j \| r_j \| N_j, HID_j = a_j \oplus h(PSW_i \| r_i) \]
   \( V_i = h(HID_j \| K_V) \oplus N_j \). Finally, \( V_i \) sends the login request \( M_1 = \{ A_j, V_i, ID_{CS}, PID_j \} \) to \( FN_j \) through the common channel.

2. After receiving the message \( M_1 \) from \( V_i \), \( FN_j \) first selects a random number \( N_2 \), and then calculates 
   \[ A_2 = h(A_1 \| K_{FN} \| HID_j) \oplus N_2, V_2 = h(A_1 \| K_{FN} \| V_i), \]
   and finally \( FN_j \) transmits the message \( M_2 = \{ PID_j, PFID_j, A_2, V_1, V_2 \} \) to CS.

3. After receiving message \( M_2 \) from \( FN_j \), CS first indexes \( K_{FN} \) according to \( PFID_j \), then calculates \( HID_j = h(PID_j \| K_{CS}), N_1 = h(HID_j \| K_V) \oplus V_1, V_1 = h(HID_j \| K_V) \oplus N_1, \) and compares \( V_1 = V_i \). If it is equal, CS believes that \( V_i \) is legal. Otherwise, the authentication process is terminated. CS calculates 
   \[ V_2 = h(A_2 \| K_{FN} \| V_i) \] and compares \( V_2 = V_1 \). If it is equal, it means that CS believes that \( FN_j \) is legal. Otherwise, the authentication process is terminated. After authenticating \( V_i \) and \( FN_j \), CS calculates 
   \[ A_1 = N_1 \oplus PID_j, HID_j = h(FID_j \| K_{CS}), N_2 = h(A_j \| K_{FN}) \]
Table 1: The summary of authentication protocols.

<table>
<thead>
<tr>
<th>Protocols</th>
<th>Cryptographic techniques</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Li et al. [19]</td>
<td>(1) Utilized digital signature</td>
<td>(1) Does not resist key compromise impersonation attacks</td>
</tr>
<tr>
<td></td>
<td>(2) Utilized asymmetric encryption</td>
<td>(2) Does not provide user anonymity</td>
</tr>
<tr>
<td></td>
<td>(3) Based on anonymous authentication</td>
<td>(1) Does not resist replay attacks</td>
</tr>
<tr>
<td></td>
<td>(1) Utilized one-way hash function</td>
<td>(2) Does not resist offline identity guessing attacks</td>
</tr>
<tr>
<td>Ying and Nayak [23]</td>
<td>(1) Utilized one-way hash function</td>
<td>(3) Does not provide mutual authentication</td>
</tr>
<tr>
<td></td>
<td>(2) Based on Diffie–Hellman problem</td>
<td>(1) Does not resist impersonation attacks</td>
</tr>
<tr>
<td></td>
<td>(3) Based on anonymous authentication</td>
<td>(2) Does not resist sensor capture attacks</td>
</tr>
<tr>
<td>Mohit et al. [24]</td>
<td>(1) Utilized one-way hash function</td>
<td>(3) Does not provide untraceability</td>
</tr>
<tr>
<td></td>
<td>(2) Based on smart card</td>
<td>(1) Does not resist internal attacks</td>
</tr>
<tr>
<td>Yu et al. [25]</td>
<td>(1) Utilized one-way hash function</td>
<td>(2) Does not resist stolen smart card attacks</td>
</tr>
<tr>
<td></td>
<td>(2) Based on smart card</td>
<td>(3) Does not resist known session-specific temporary information attacks</td>
</tr>
<tr>
<td>Ma et al. [10]</td>
<td>(1) Utilized one-way hash function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2) Based on anonymous authentication</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(3) Utilized ECC</td>
<td></td>
</tr>
<tr>
<td>Wazid et al. [34]</td>
<td>(1) Utilized one-way hash function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2) Based on anonymous authentication</td>
<td></td>
</tr>
<tr>
<td>Eftekhari et al. [9]</td>
<td>(1) Utilized one-way hash function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2) Based on anonymous authentication</td>
<td></td>
</tr>
<tr>
<td>Wu et al. [32]</td>
<td>(1) Utilized one-way hash function</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2) Based on smart card</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(3) Utilized ECC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(4) Two-factor</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Notations used in the proposed protocol.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_i$</td>
<td>The $i$-th vehicle</td>
</tr>
<tr>
<td>$FN_j$</td>
<td>The $j$-th fog node</td>
</tr>
<tr>
<td>CS</td>
<td>Cloud server</td>
</tr>
<tr>
<td>$ID_i$, $ID_{FN_j}$, $ID_{CS}$</td>
<td>Identities of $V_i$, $FN_j$, and CS</td>
</tr>
<tr>
<td>$PSW_i$</td>
<td>Password of the $V_i$</td>
</tr>
<tr>
<td>$K_{FN_i}$</td>
<td>Shared key of $FN_j$ and CS</td>
</tr>
<tr>
<td>$K_{CS}$</td>
<td>Secret key of CS</td>
</tr>
<tr>
<td>$K_V$</td>
<td>Counter value of $V_i$</td>
</tr>
<tr>
<td>SK</td>
<td>Session key</td>
</tr>
</tbody>
</table>

Figure 2: $V_i$ registration phase.

\( V_i \) registration phase.

Select \( ID_i \), \( PSW_i \), \( r_i \)

Compute: \( HID_i = h(ID_i \| r_i) \)

\( a_i = HID_i \oplus h(PSW_i \| r_i) \)

\( P_i = h(ID_i \| PSW_i \| r_i) \)

\( V_i \) stores \( \{a_i, P_i, r_i, K_V\} \) in the smart card

\( CS \)

\( \{PID_i\} \)

\( HID_i = h(PID_i \| K_{CS}) \)

\( CS \) keeps \( \{PID_i, K_V\} \) in its database
Select \( FID_j, r_j \)
\[ PFID_j = h(FID_j \|| r_j) \]
\[ R_j = h(FID_j \|| ID_{CS}) \oplus N_j \]
\[ \beta_j = HID_j \oplus h(R_j \|| r_j) \]
\[ \text{FN}_j \text{ stores } \{ K_{PN}, \beta_j, r_j, N_j \} \] in its database

\[ C_{PN} = h(PFID_j \|| K_{CS}) \]

\[ CS \text{ keeps } \{ PFID_j, K_{PN}, FID_j \} \] in its database

---

**Figure 3:** \( \text{FN}_j \) registration phase.

---

\[ V_i, FN_j, CS \]

- **Enter:** \( ID_x, PSW_i \)
- **Compute:** \( P^* = h(ID_x \|| PSW_i \|| r_i) \)
- **Check:** \( P^* = P_i \)
- **Generate:** \( N_1 \)
- **Compute:** \( A_1 = h(A_1 \|| K_{PN} \|| HID_j) \oplus N_2 \)
- **Compute:** \( V_2 = h(A_2 \|| K_{PN} \|| V_i) \)
- **Compute:** \( V_1 = h(HID_j \|| K_{CS} \|| N_1) \)
- **Compute:** \( V_2^* = h(HID_j \|| K_{PN} \|| SK) \)
- **Check:** \( V_2^* = V_2 \)
- **Update:** \( K_{Y} = K_{Y} + 1 \)

\[ N_1 \oplus N_2 \oplus N_3 \oplus HID_j = h(HID_j \|| N_1) \oplus N_2 \oplus N_3 \oplus HID_j \]
\[ SK = h(N_1 \oplus N_2 \oplus N_3 \oplus HID_j \|| SK) \]
\[ V_1^* = h(HID_j \|| K_{SF} \|| V_i) \]
\[ V_4^* = h(HID_j \|| K_{CS} \|| V_i) \]

\[ N_1, N_2, V_1, V_4 \]

---

**Figure 4:** Login and authentication phase.

\( \text{HID}_j \oplus A_2, \) selects a random number \( N_3, \) and calculates
\( N_1 \oplus N_2 \oplus N_3 \oplus HID_j, \)
\( N_Y = h(HID_j \|| N_1) \oplus N_2 \oplus N_3 \oplus N_Y \)
\( SK = h(N_1 \oplus N_2 \oplus N_3 \oplus HID_j \|| SK) \)
\( V_3 = h(HID_j \|| K_{SF} \|| SK) \)
\( V_4 = h(HID_j \|| K_{CS} \|| SK) \)

\( V_1^* = h(HID_j \|| K_{PN} \|| SK) \)
\( V_4^* = h(HID_j \|| K_{CS} \|| V_i) \)
\( K_Y = K_Y + 1 \)

\( (5) \) After receiving message \( M_4 \) from \( \text{FN}_j, V_i \) calculates
\( N_4 \oplus N_3 \oplus HID_j = h(HID_j \|| N_1) \oplus N_2 \oplus N_3 \oplus HID_j, \)
\( V_4^* = h(HID_j \|| K_{PN} \|| SK), \)
and compares \( V_1^* = V_4^* \). If it is equal, it means that \( V_i \) believes that \( \text{FN}_j \) and \( CS \) are legal. Otherwise, the authentication process is terminated. Finally, \( V_i \) updates \( K_Y = K_Y + 1 \).

**4. Security Analysis**

4.1. BAN Logic. BAN logic is a formal security analysis method [35]. In this part, we use BAN logic to prove that vehicles, fog nodes, and cloud servers share a session key \( SK \) and further prove the correctness of our protocol. The rules used in BAN logic are shown in the references.
4.1.2. Goals.

1. Message-meaning rule: \((P) \equiv P \rightarrow^{K} Q, P \triangleleft \{X\}_K \rightarrow (P \mid Q \rightarrow X)\)
2. Freshness rule: \((P) \equiv \$ (X))/\((P) \equiv \$ (X, Y))\)
3. Nonce-verification rule: \((P) \equiv \$ (X), P \equiv Q \mid X) /\((P) \equiv Q \equiv X)\)
4. Jurisdiction rule: \((P) \equiv \$ (X), P \equiv Q \mid X) /\((P) \equiv Q \equiv X)\)
5. Belief rule: \((P) \equiv X, P \equiv Y) /\((P) \equiv (X, Y))\)
6. Session key rule: \((P) \equiv \$ (X), P \equiv Q \equiv X) /\((P) \equiv P \rightarrow^{K} Q)\)

4.1.2. Goals.

\[ G1 \ V_i \rightarrow CS: \{PID_i, A_1, V_i\} \]
\[ G2 \ FN_j \rightarrow CS: \{PFID_i, A_2, V_2\} \]
\[ G3 \ CS \rightarrow FN: \{N_{V_3}\} \]
\[ G4 \ CS \rightarrow V_j: \{N'_{V_3}, V_j\} \]

4.1.3. Idealizing Communication.

\[ M1 \ V_i \rightarrow CS: [PID_i, A_1, V_i] \]
\[ M2 \ FN_j \rightarrow CS: [PFID_i, A_2, V_2] \]
\[ M3 \ CS \rightarrow FN: [N_{V_3}, V_3] \]
\[ M4 \ CS \rightarrow V_j: [N'_{V_3}, V_j] \]

4.1.4. Initial State Assumptions

\[ A1 \ V_i \equiv \$ (N_1) \]
\[ A2 \ FN_j \equiv \$ (N_2) \]
\[ A3 \ CS \equiv \$ (N_3, h(ID_i)\{K_{V_3}\}) \]
\[ A4 \ CS \equiv V_i \rightarrow^{h(ID_i)\{K_{V_3}\}} \equiv CS \]
\[ A5 \ CS \equiv \$ (N_1) \]
\[ A6 \ CS \equiv V_j \rightarrow N_1 \rightarrow^{h(ID_i)\{K_{V_3}\}} \equiv CS \]
\[ A7 \ CS \equiv FN_j \rightarrow \equiv N_2 \]
\[ A8 \ CS \equiv \$ (N_2) \]
\[ A9 \ CS \equiv FN_j \rightarrow \equiv N_2 \]
\[ A10 \ CS \equiv HID_i \]
\[ A11 \ CS \equiv HID \rightarrow h(ID_i)\{N_1\} \]
\[ A12 \ FN_j \equiv \equiv FN_j \rightarrow \equiv CS \]
\[ A13 \ FN_j \equiv CS \rightarrow \equiv N_3 \]
\[ A14 \ FN_j \equiv CS \rightarrow \equiv HID_j \]
\[ A15 \ FN_j \equiv \$ (N_1) \]
\[ A16 \ FN_j \equiv \$ (N_3) \]
\[ A17 \ FN_j \equiv \$ (HID_j) \]
\[ A18 \ V_i \equiv V_i \rightarrow \equiv CS \]
\[ A19 \ V_j \equiv CS \rightarrow \equiv N_3 \]

\[ A20 \ V_i \equiv CS \rightarrow \equiv HID_j \]
\[ A21 \ V_j \equiv \$ (N_2) \]
\[ A22 \ V_j \equiv \$ (N_3) \]
\[ A23 \ V_i \equiv \$ (HID_j) \]

4.1.5. Detailed Steps. By considering the message M1 and using the seeing rule, we get

\[ S1: CS \equiv \{V_i: \langle N_1\rangle_{h(ID_i)\{K_{V_3}\}}, PID_i, A_i\} \]

Using S1, we get

\[ S2: CS \equiv \{N_2\}_{h(ID_i)\{K_{V_3}\}} \]

Under the premise of assuming A4, using S2, and the message-meaning rule, we get

\[ S3: CS \equiv V_i \rightarrow N_1 \]

In the case of conclusion S3, using assumption A5, the freshness rule, and the nonce-verification (N-V) rule, we get

\[ S4: CS \equiv V_i \equiv N_1 \]

In the case of conclusion S4, using assumption A6, and the jurisdiction rule, we get

\[ S5: CS \equiv N_1 \]

In addition, considering the message M2, we get

\[ S6: CS \equiv \{PID_i, A_2: \langle N_2\rangle_{h(ID_i)\{K_{V_3}\}}, HID_j\} \]

Using S6, we get

\[ S7: CS \equiv \{N_2\}_{h(ID_i)\{K_{V_3}\}} \]

Under the premise of assuming A7, using S7, and the message-meaning rule, we get

\[ S8: CS \equiv FN_j \rightarrow N_2 \]

In the case of conclusion S8, using assumption A8, the freshness rule, and the nonce-verification (N-V) rule, we get

\[ S9: CS \equiv FN_j \equiv N_2 \]

In the case of conclusion S9, using assumption A9, and the jurisdiction rule, we get

\[ S10: CS \equiv N_2 \]

Because \(SK = h(N_1 \oplus N_2 \oplus N_3 \oplus HID_i \oplus HID_j)\), according to the conclusions A10, A11, S10, and S5 and the belief rule, we get

\[ S11: CS \equiv V_i \rightarrow^{h(ID_i)\{N_1\}} \equiv (G5) \]

Using A5, S11, and the SK rule, we get

\[ S12: CS \equiv V_i \equiv V_i \rightarrow^{h(ID_i)\{N_1\}} \equiv (G6) \]
Using A8, S11, and the SK rule, we get
\[ S13: \text{CS} \equiv \text{FN}_j \equiv V_{r}^{\text{SK}} \text{FN}_j, \quad (G7). \] (13)

Using S14, we get
\[ S14: \text{FN}_j \equiv \left\{ V_{r}^{N_3}: \langle N_1, N_3, \text{HID}_r \rangle_h \left( \text{HID}_j || N_j \right) V_3 \right\}. \] (14)

By considering the message M3 and using the seeing rule, we get
\[ S15: \text{FN}_j \equiv \left\{ \langle N_1, N_3, \text{HID}_r \rangle_h \left( \text{HID}_j || N_j \right) V_3 \right\}. \] (15)

By considering the message M3 and using the seeing rule, we get
\[ S16: \text{FN}_j \equiv \text{CS} \sim \left\{ \langle N_1, N_3, \text{HID}_r \rangle \right\}. \] (16)

By considering the message M3 and using the seeing rule, we get
\[ S17: \text{FN}_j \equiv \text{CS} \equiv \left\{ N_1, N_3, \text{HID}_r \right\}. \] (17)

Applying this for each component, we get
\[ S18: \text{FN}_j \equiv \text{CS} \equiv N_1, \]
\[ S19: \text{FN}_j \equiv \text{CS} \equiv N_3, \]
\[ S20: \text{FN}_j \equiv \text{CS} \equiv \text{HID}_r. \] (18)

In the case of conclusion S18, using assumption A15, and the jurisdiction rule, we get
\[ S21: \text{FN}_j \equiv N_1. \] (19)

In the case of conclusion S22, using assumption A16, and the jurisdiction rule, we get
\[ S22: \text{FN}_j \equiv N_1. \] (20)

In the case of conclusion S23, using assumption A17, and the jurisdiction rule, we get
\[ S23: \text{FN}_j \equiv \text{HID}_r. \] (21)

Because \( \text{SK} = h(N_1 @ N_2 @ N_3 @ \text{HID}_r @ \text{HID}_r) \), according to the conclusions S21, S22, S23 and the belief rule, we get
\[ S24: \text{FN}_j \equiv V_{r}^{\text{SK}} \text{FN}_j. \quad (G3). \] (22)

Using A15, S24, and the SK rule, we get
\[ S25: \text{FN}_j \equiv \text{FN}_j \equiv V_{r}^{\text{SK}} \text{FN}_j. \quad (G4). \] (23)

Applying this for each component, we get
\[ S26: \left\{ V_{r}^{N_3}: \langle N_2, N_3, \text{HID}_r \rangle_h \left( \text{HID}_j || N_j \right) V_4 \right\}. \] (24)

Using S26, we get
\[ S27: V_{r} \equiv \left\{ \langle N_2, N_3, \text{HID}_r \rangle_h \left( \text{HID}_j || N_j \right) V_4 \right\}. \] (25)

Under the premise of assuming A18, using S27, and the message-meaning rule, we get
\[ S28: V_{r} \equiv \text{CS} \sim \left\{ \langle N_2, N_3, \text{HID}_j \rangle \right\}. \] (26)

Applying this for each component, we get
\[ S30: V_{r} \equiv \text{CS} \equiv N_2, \]
\[ S31: V_{r} \equiv \text{CS} \equiv N_3, \]
\[ S32: V_{r} \equiv \text{CS} \equiv \text{HID}_r. \] (28)

Because \( \text{SK} = h(N_1 @ N_2 @ N_3 @ \text{HID}_r @ \text{HID}_r) \), according to the conclusions S31, S32, and S33 and the belief rule, we get
\[ S34: V_{r} \equiv V_{r}^{\text{SK}} \text{FN}_j. \quad (G1). \] (32)

Using A21, S34, and the SK rule, we get
\[ S35: V_{r} \equiv \text{FN}_j \equiv V_{r}^{\text{SK}} \text{FN}_j. \quad (G2). \] (33)

### 4.2. Formal Security Analysis

In this part, we use the ROR model to formally prove the security of our proposed protocol. The ROR model judges the security of the protocol by calculating the session key SK probability of an ordinary situation [36, 37].

#### 4.2.1. ROR Model

The protocol consists of three entities: vehicle, fog node, and cloud server. In the ROR model, we use \( \Pi_{r}^{\text{V}}, \Pi_{r}^{\text{FN}}, \) and \( \Pi_{r}^{\text{CS}} \) to represent the \( x \)-th communication of the \( V_{r} \), the \( y \)-th communication of the \( \text{FN}_j \), and the \( z \)-th communication of the \( \text{CS} \), respectively. We also define that the attacker, \( A \) can have the following query capabilities, where \( Z = \{ \Pi_{r}^{\text{V}}, \Pi_{r}^{\text{FN}}, \Pi_{r}^{\text{CS}} \} \).
Execute ($Z$): by performing this query operation, $A$ can intercept the messages transmitted on the public channel.

Hash (string): by performing this query operation, $A$ can obtain the hash value of the input string.

Send ($Z, M$): by performing this query operation, $A$ can send message $M$ to $Z$ and receive the response from $Z$.

Corrupt ($Z$): by performing this query operation, $A$ can obtain a party’s secret values, such as some values in the smart card, long-term key, or temporary information.

Test ($Z$): by performing this query operation, $A$ flips a coin $c$. If $c = 1$, $A$ can obtain an accurate session key; if $c = 0$, $A$ can obtain a random string of the same length as the session key.

4.2.2. Theorem. In the ROR model, assume $A$ can perform execute, hash, send, corrupt, and test queries. Then, the probability that $A$ can break the proposed protocol $P$ in polynomial time is $\text{Adv}^P_A(\xi) \leq q_{\text{send}}(2^{-1}) + (3q_{\text{hash}}^22^2) + 2\max(C'q_{\text{send}} + q_{\text{send}}2^2) + (q_{\text{exe}} + q_{\text{send}}2^2)/p$, where $q_{\text{hash}}$ represents the number of times hash queries are executed, $q_{\text{send}}$ represents the number of times send queries are executed, $q_{\text{exe}}$ represents the number of times execute queries are executed, $I$ represents the bits of biological information, and $C'$ and $s$ are constants in Zipf’s law.

4.2.3. Proof. We played five rounds of games, which were expressed as follows: GM$_0$ to GM$_6$. Succ$_A^{GM_i}$ represents the event that $A$ can win in the game GM$_i$. Succ$_A^{GM_i}$ is the advantage of $A$ for winning GM$_i$. Pr $\left[ \text{Succ}_A^{GM_i} \right]$ represents the probability that $A$ has in breaking the security of SK for protocol $P$. The specific steps of GM$_i$ are as follows:

GM$_0$: GM$_0$ is the first-round game in the ROR model and a real attack. We choose a coin $c$ to start the round. Therefore, in GM$_0$, we can obtain the probability that $A$ can successfully break $P$ as

$$\text{Adv}^P_A = \left| 2\text{Pr}[\text{Succ}_A^{GM_0}] - 1 \right|.$$  

GM$_1$: GM$_1$ adds an execute query to GM$_0$. In GM$_1$, $A$ can only obtain the messages transmitted on the public channel. After GM$_1$, $A$ will query the session key SK through the test, but $A$ cannot obtain five values $\{\text{N}_1, \text{N}_2, \text{N}_3, \text{HID}_1, \text{HID}_2\}$, so the probability that GM$_0$ is equal to that of GM$_1$ is

$$\text{Pr}[\text{Succ}_A^{GM_1}] = \text{Pr}[\text{Succ}_A^{GM_0}].$$  

GM$_2$: GM$_2$ adds a send query to GM$_1$. According to Zipf’s law [38], we obtain

$$\left| \text{Pr}[\text{Succ}_A^{GM_1}] - \text{Pr}[\text{Succ}_A^{GM_2}] \right| \leq (q_{\text{send}}/2^2).$$  

GM$_3$: GM$_3$ adds the hash query to GM$_2$. The maximum probability of text collision in transmission is $(q_{\text{exe}} + q_{\text{send}})^2/2p$, and we can obtain

$$\left| \text{Pr}[\text{Succ}_A^{GM_2}] - \text{Pr}[\text{Succ}_A^{GM_3}] \right| \leq \left( \frac{q_{\text{exe}} + q_{\text{send}}}{2} \right)^2.$$

GM$_4$: in this round, we verify the security of the session key SK using two events. One is to obtain the long-term key of $\Pi^{\text{CS}}_2$ to verify the perfect forward security, and the other is to obtain temporary information to verify that the protocol can resist the known session-specific temporary information attacks.

(1) Perfect forward security: using $\Pi^{\text{CS}}_2$, $A$ attempts to obtain the private key $K_{\text{CS}}$ of CS, or $A$ uses $\Pi^{\text{V}}_i$ or $\Pi^{\text{FN}}_i$ to obtain some secret values in the registration phase.

(2) Known session-specific temporary information attacks: $A$ uses one of $\Pi^{\text{V}}_i$ or $\Pi^{\text{FN}}_i$ or $\Pi^{\text{CS}}$ to attempt to obtain temporary information.

For the first event, if $A$ obtains the private key $K_{\text{CS}}$ of CS, or the secret value of $\Pi^{\text{V}}_j$ and $\Pi^{\text{FN}}_j$ in the registration phase, but $A$ cannot get the random number $\text{N}_1, \text{N}_2, \text{N}_3, \text{HID}_1$, it cannot calculate session key SK, where $\text{SK} = h(\text{N}_1 \oplus \text{N}_2 \oplus \text{N}_3 \oplus \text{HID}_1 \oplus \text{HID}_2)$. For the second event, if $A$ can obtain $\text{N}_1$, but the values of $\text{N}_2$ and $\text{N}_3$ are confidential, the SK cannot be calculated. Similarly, if $\text{N}_2$ and $\text{N}_3$ are leaked, SK cannot be calculated by $A$. Therefore, the probability of this round is

$$\left| \text{Pr}[\text{Succ}_A^{GM_3}] - \text{Pr}[\text{Succ}_A^{GM_4}] \right| \leq \frac{q_{\text{hash}}}{2^2}. $$

GM$_5$: in this round of the game, $A$ uses the corrupt query to obtain the parameter $\{a_i, P_i, r_i, K_V\}$ stored in the smart card, so $A$ wants to conduct the offline key guessing attacks. $V_j$ uses random numbers and passwords for registration, so $A$ must guess $P_i = h(\text{ID}_i \parallel \text{PSW}_i \parallel r_i)$, but the probability of guessing a random number is $1/2^i$, which can be ignored. Using Zipf’s law [38], we can obtain

$$\left| \text{Pr}[\text{Succ}_A^{GM_4}] - \text{Pr}[\text{Succ}_A^{GM_5}] \right| \leq \max \left( C'q_{\text{exe}} + q_{\text{send}}^2 \right).$$

GM$_6$: this round of the game is to verify that protocol $P$ can resist the impersonation attacks, $A$ uses $h(\text{N}_1 \oplus \text{N}_2 \oplus \text{N}_3 \oplus \text{HID}_1 \oplus \text{HID}_2)$ to query, and the game is terminated. Therefore, the probability that $A$ can guess SK is

$$\left| \text{Pr}[\text{Succ}_A^{GM_5}] - \text{Pr}[\text{Succ}_A^{GM_6}] \right| \leq \frac{q_{\text{hash}}}{2^2}. $$

Because the probability of success and failure of the GM$_6$ is $1/2$, we obtain

$$\left| \text{Pr}[\text{Succ}_A^{GM_5}] - \text{Pr}[\text{Succ}_A^{GM_6}] \right| \leq \frac{q_{\text{hash}}}{2^2}. $$

GM$_7$: in this round of the game, $A$ uses the final query to obtain the secret value of $\Pi^{\text{CS}}_i$ to verify the perfect forward security.
\[
\frac{1}{2} \text{Adv}^A_P = \frac{1}{2} \Pr\left[\text{Succ}_{GM^0}^A\right] - \frac{1}{2} \Pr\left[\text{Succ}_{GM^1}^A\right]
\]
\[
= \Pr\left[\text{Succ}_{GM^0}^A\right] - \Pr\left[\text{Succ}_{GM^1}^A\right]
\]
\[
= \Pr\left[\text{Succ}_{GM^1}^A\right] - \Pr\left[\text{Succ}_{GM^0}^A\right]
\]
\[
\leq \sum_{i=0}^{5} \Pr\left[\text{Succ}_{GM, i}^A\right] - \Pr\left[\text{Succ}_{GM}^A\right]
\]  
(41)
\[
\frac{q_{\text{send}}}{2^{l+1}} + \frac{3q_{\text{hash}}^2}{2^{l+1}} + \max \left\{ C', q_{\text{send}} \right\} + \frac{(q_{\text{hash}} + q_{\text{send}})^2}{2p}
\]

Finally, we can obtain
\[
\text{Adv}^A_P \leq \frac{q_{\text{send}}}{2^{l+1}} + \frac{3q_{\text{hash}}^2}{2^{l+1}} + 2 \max \{ C' \}
\]
\[
\cdot q_{\text{send}} + \frac{q_{\text{hash}} + q_{\text{send}}^2}{2p}
\]  
(42)

4.3. ProVerif. ProVerif is a formal automatic verification tool, which can verify confidentiality, identity, anonymity, and so on [39, 40]. In this paper, we use the ProVerif code to achieve vehicle registration, fog node registration, and authentication between the two parties and the CS and verify the security and effectiveness of our proposed protocol through ProVerif.

ProVerif demonstrates that the specific operation works as follows. Our protocol includes three entities: vehicle, fog node, and cloud server. The symbols and operation definitions used in ProVerif are shown in Figure 5.

The proof contains six events, as shown in Figure 6. The six events are vehicle started, vehicle authored, cloudserver vehicle (CS), cloudserver vehicle fognode (Fog), fognode cloudserver (Fog), and vehicle cloudserver (VC), indicating that the vehicle starts certification, the vehicle completes certification, the cloud server completes the vehicle certification, and the cloud server completes the fog node certification, respectively. The fog node completed the certification of the cloud server, and the vehicle completed the certification of the cloud server.

Then, we use ProVerif to query whether A can interpret the common key SK through the data transmitted on the common channel. The query operation is shown in Figure 7.

Finally, we get the verification result using the ProVerif tool, as shown in Figure 8. The result shows that A cannot interpret the common key SK of the V_i, FN_j, and CS.

4.4. Informal Security Analysis. This part is an informal security analysis of our proposed agreement. We have proved that the protocol can meet common security requirements. The specific proof is as follows.

4.4.1. Mutual Authentication. In the authentication phase, with the help of CS, mutual authentication between V_i and FN_j is realized. V_i in message M_1 is the value CS uses to authenticate V_i, V_2 in message M_2 is the value CS uses to authenticate FN_j, and V_3 and V_4 in message M_3 are the values CS uses to authenticate FN_j and V_i, respectively. Therefore, the mutual authentication among V_i, FN_j, and CS is realized in the authentication phase.

4.4.2. Replay Attacks. In this protocol, we use cumulative value K_v to resist replay attacks. In the V_i registration phase, we initialize K_v to 0. As the session progresses, it carries out +1 operation on the value K_v, saves it to its database after CS authenticates V_i, FN_j, and carries out the necessary calculation. After CS authenticates V_i and generates the session key, it also carries out the +1 operation on the value K_v and saves it to the smart card. In this manner, K_v on both sides is synchronous and equal, and the session process is completed smoothly. If A repeatedly sends message M_1 intercepted in the public channel, CS continues to calculate the value K_v + 1 in the authentication phase. Value V_4 generated using K_v is not equal to value V_4 calculated by V_i using K_v, stored in its smart card, because the value K_v in the smart card of V_i cannot keep up with the CS update speed, so the authentication fails. Thus, our protocol can resist replay attacks.

4.4.3. Man-in-the-Middle Attacks. Suppose that A can intercept the message M_1 = \{A_1, V_1, ID_{CS}, PID_{i}\} transmitted on the public channel between V_i and FN_j. Since A cannot obtain the information \{a_{\text{rand}}, K_v, r_i\} in the smart card and the identity ID of V_j, A cannot calculate the values \{K_v, HID, N_1\} required for V_3, where V_1 = h(HID || K_v) || N_1. Therefore, after A tampers with M_1, it cannot pass the authentication of FN_j. Similarly, because the privacy value is unknown, A cannot calculate the authentication value V_j, V_3, or V_4 and cannot complete the verification after intercepting the information M_2, M_3, or M_4. Therefore, our protocol can resist man-in-the-middle attacks.

4.4.4. User Anonymity. The real identities of V_i and FN_j are transmitted on the secure channel and are protected by pseudoidentity PID and PFID in the authentication phase. The anonymity of V_i and FN_j is ensured. Therefore, our protocol can provide user anonymity.

4.4.5. Untraceability. If A wants to trace the V_j, it intercepts the messages \{M_1, M_2, M_3, M_4\} transmitted on the common channel. Since the random numbers \{N_1, N_2, N_3\} are used, this means that messages \{M_1, M_2, M_3, M_4\} are different during each session. In addition, A cannot obtain the random numbers \{N_1, N_2, N_3\}, so A cannot be traced back to V_j. Therefore, our protocol can provide untraceability.

5. Security and Performance Comparisons

In this part, we compare our protocol with those of Ma et al. [10], Wazid et al. [34], Eftekhari et al. [9], and Wu et al. [32] in terms of security, computational cost, and communication cost.
5.1. **Security Comparisons.** When comparing protocol security, we use ✔ to indicate that the protocol can resist the attacks and × to indicate that the protocol cannot resist the attacks. The results of comparing protocol security are shown in Table 3. It can be seen that our protocol can resist known attacks and have better security. Ma et al.’s protocol [10] cannot provide user anonymity and untraceability and is vulnerable to impersonation attacks and known session-specific temporary information attacks. The protocols in [9, 32, 34] and our protocol are secure.

5.2. **Performance Comparison.** Performance analysis is conducted from the aspects of computational cost and communication cost. We analyze and compare the computational cost from the login authentication phase of each protocol. If the computational cost of XOR and join operations is negligible. The computational cost comparison is shown in Table 4. It is obvious that the protocols of Ma et al. [10], Wazid et al. [34], Eftekhar et al. [9], and Wu et al. [32] perform point multiplication, Wazid et al. [34] and Wu et al. [32] perform fuzzy extraction, and Wazid et al.’s protocol [34] and Eftekhar et al. [9] also perform ECC point addition. Only our proposed protocol performs the hash operation, so its computational cost is less.

Here, $T_{pm}$ represents the time taken to perform a point multiplication operation, $T_{pa}$ represents the time taken to execute an ECC point addition, $T_f$ represents the time taken to execute a fuzzy extraction function, and $T_h$ represents the time taken to execute a hash operation.

In the comparison of communication cost, we assume that the length of the identity and the random number are 160 bits, the length of the timestamp is 32 bits, the length of the one-way hash function is 256 bits, and the length of ECC point is 320 bits. Therefore, based on our assumption, the communication costs of the protocols of Ma et al. [10], Wazid et al. [34], Eftekhar et al. [9], and Wu et al. [32] are 4512 bits, 3488 bits, 4416 bits, and 4448 bits. Here, we illustrate our

```
(* channel*)
free ch : channel. (* public channel *)
free sch: channel [private]. (* secure channel, used for registering *)
(* shared keys *)
free SKv : bitstring [private].
free SKf : bitstring [private].
free SKc : bitstring [private].
(* constants *)
free Kcs:bitstring [private].
free P:bitstring.
free B:bitstring.
free yj:bitstring.
(* functions & reductions & equations *)
fun h(bitstring) : bitstring. (* hash function *)
fun mult(bitstring,bitstring) : bitstring. (* scalar multiplication operation *)
fun add(bitstring,bitstring):bitstring. (* Addition operation *)
fun sub(bitstring,bitstring):bitstring. (* Subtraction operation *)
fun mod(bitstring,bitstring):bitstring. (* modulus operation *)
fun con(bitstring,bitstring):bitstring. (* concatenation operation *)
red for all m:bitstring, n:bitstring; getmess(con(m,n))=m.
fun xor(bitstring,bitstring):bitstring. (* XOR operation *)
equation for all m:bitstring, n:bitstring; xor(xor(m,n),n)=m.
fun Gen(bitstring):bitstring. (* Generator operation *)
fun Rep(bitstring,bitstring):bitstring.
```

**Figure 5:** The definition in the ProVerif tool.
protocol as an example to show the specific analysis. In our protocol, the messages transmitted in the login authentication phase are $M_1 = \{A_1, V_1, IDCS, PID\}$, $M_2 = \{A_2, V_1, V_2, PFID_1, PID_1\}$, $M_3 = \{N_{X'}, N_{Y'}, V_3, V_4\}$, and $M_4 = \{N'_{X'}, N'_{Y'}\}$, where $\{A_1, A_2, N'_{X'}, N'_{Y'}\}$ are random strings, IDCS is an identity, and $\{V_1, V_2, V_3, V_4, PFID_1, PID_1\}$ are hash values. Therefore, the total communication cost of our proposed protocol is 2336 bits. The comparison of communication cost is shown in Table 5. Obviously, the communication cost of our proposed protocol is less.

In the security comparison, we found that Ma et al.’s protocol [10] cannot provide user anonymity and untraceability and is vulnerable to impersonation attacks and known session-specific temporary information attacks.
Although the protocols of [9, 32, 34] can resist known security attacks, the overhead in the aspect of computational cost and communication cost is much more than that of our proposed protocol. Therefore, our protocol is better in terms of security and performance.

6. Conclusions

In this paper, we first review the AKE protocol in IoV and SIoV, and then, we propose a lightweight and authenticated key agreement protocol using fog nodes. The security analysis of the protocol is conducted by using BAN, ROR, and ProVerif. The comparison of security and performance shows that the protocol achieves higher performance in terms of computing power and communication cost compared with other protocols. In future research, we will focus on improving the security and performance of the protocol in SIoV.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This article was supported by the Guangxi Key Laboratory of Trusted Software (no. KX202033).

References


