An Enhanced Privacy-Preserving Authentication Scheme for Vehicle Sensor Networks

Yousheng Zhou 1,2,3, Xiaofeng Zhao 1,*, Yi Jiang 1, Fengjun Shang 1, Shaojiang Deng 2,† and Xiaojun Wang 4

1 College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China; zhouys@cqupt.edu.cn (Y.Z.); jiangyi@cqupt.edu.cn (Y.J.); shangfj@cqupt.edu.cn (F.S.)
2 College of Computer Science, Chongqing University, Chongqing 400044, China
3 School of Cyber Security and Information Law, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
4 School of Electronic Engineering, Dublin City University, Dublin, Ireland; xiaojun.wang@dcu.ie
* Correspondence: s160201006@stu.cqupt.edu.cn (X.Z.); sj_deng@cqu.edu.cn (S.D.); Tel.: +86-189-8384-7420 (X.Z.); +86-23-6510-2502 (S.D.)

Received: 30 September 2017; Accepted: 2 December 2017; Published: 8 December 2017

Abstract: Vehicle sensor networks (VSNs) are ushering in a promising future by enabling more intelligent transportation systems and providing a more efficient driving experience. However, because of their inherent openness, VSNs are subject to a large number of potential security threats. Although various authentication schemes have been proposed for addressing security problems, they are not suitable for VSN applications because of their high computation and communication costs. Chuang and Lee have developed a trust-extended authentication mechanism (TEAM) for vehicle-to-vehicle communication using a transitive trust relationship, which they claim can resist various attacks. However, it fails to counter internal attacks because of the utilization of a shared secret key. In this paper, to eliminate the vulnerability of TEAM, an enhanced privacy-preserving authentication scheme for VSNs is constructed. The security of our proposed scheme is proven under the random oracle model based on the assumption of the computational Diffie–Hellman problem.

Keywords: vehicle sensor network; authentication; V2V; provable security

1. Introduction

With the rapid development of the intelligent transportation systems (ITSs) [1], vehicular ad hoc networks (VANETs) have become increasingly popular. The vehicles in VANETs can communicate with each other via wireless communication [2]. If vehicles can interact with other vehicles or the roadside infrastructure to exchange collected data for decision-making and safer driving, traffic jams can be avoided and the safety of drivers can be guaranteed to the utmost extent; consequently, VANETs are a promising means of improving traffic safety and management. At present, vehicles are equipped with various sensors that can provide valuable data. Further equipping vehicles with onboard sensing devices can turn VANETs into vehicle sensor networks (VSNs) [3]. Therefore, the authentication protocols used in VANETs can also be used in VSNs. Moreover, dynamic traffic information and many types of physical data associated with traffic distributions can be sensed and collected by such vehicular communication networks. Therefore, VSNs are expected to significantly facilitate future wireless communication.

Two types of communication exist in VANETs, namely vehicle-to-vehicle (V2V) communication and vehicle-to-infrastructure (V2I) communication, which depend on two essential kinds of
components: onboard units (OBUs) and roadside units (RSUs). As shown in Figure 1, OBUs are the wireless communication units equipped on vehicles, whereas RSUs are wireless access units located at significant places on the road. Generally, to assist vehicles and RSUs in performing certain tasks, such as authentication, a backend server should be deployed remotely. The characteristics of VANETs include self-organization, channel-opening behavior, and rapidly changing and multiple-hop topologies. Due to these characteristics, VANETs are more susceptible to malicious attacks. Since safety and privacy are a concern in many applications in VANETs [4,5], communication security issues are worthy of attention. Among the various security mechanisms used in VANETs, authentication is one basic component that is critical for ensuring security. However, a desirable authentication scheme must be efficient and practical for use in fast-moving scenarios, which means that the computation cost for authentication should be as low as possible to enable real-time response. In addition, privacy preservation should be considered, including the identity privacy, location privacy, and interest privacy. Moreover, the location of a vehicle is closely related to who is driving it. When a vehicle communicates with others in a wireless network, it will not be acceptable to the public if the vehicle’s identity and location are disclosed. Thus, privacy preservation must be achieved in the authentication procedure. In addition, it should be possible for the real identities of the malicious vehicles to be revealed by the authorities when necessary [5]. These requirements pose a considerable challenge for the development of an ideal authentication scheme.

Figure 1. Structure of a vehicular ad-hoc network.

The contributions of this paper are as follows: (1) an enhanced privacy-preserving authentication scheme based on the Chuang–Lee’s scheme is proposed that can resist internal attack. In addition, we demonstrate the correctness and security of the improved scheme and analyze its computational costs; (2) to preserve the identity privacy of drivers, anonymity is achieved by randomizing the real identities; and (3) to preserve the location privacy of drivers, unlinkability is achieved in the authentication procedure.

The remainder of this paper is organized as follows. Related work is introduced in Section 2. Preliminaries are presented in Section 3. A review of the Chuang–Lee’s scheme is provided in Section 4. Then, a concrete description of the proposed scheme is offered in Section 5. Section 6 presents the proofs of correctness, security and performance. Finally, the conclusions are provided.
2. Related Work

To cope with the challenges associate with VANETs, many types of authentication schemes have been investigated. Porambage et al. [6] introduced a two-phase authentication protocol for sensor networks that uses certificates and consequently cannot preserve the unlinkability of messages. Raya and Hubaux [7] proposed an authentication scheme for VANETs using anonymous certificates, in which each vehicle can utilize distinct key pairs in each authentication stage to avoid being tracked. However, frequent changing of key pairs is likely to result in burdensome management and storage requirements. Lu et al. [8] proposed an alternative way to avoid the complexity of preloading a large number of anonymous certificates with the support of RSUs. When a vehicle passes an RSU, it will be issued a short-term anonymous certificate; thus, the unlinkability of messages is preserved. However, the efficiency will inevitably be low because each vehicle must frequently interact with RSUs. Subsequently, Lin et al. [9] introduced another secure scheme that does not require interaction with RSUs, in which membership managers, rather than RSUs, are responsible for the issuing of certificates based on group signatures. However, the efficiency of this solutions is low. Zhang et al. [10] presented two additional authentication schemes with privacy preservation; however, the computational costs of their methods are somewhat high because of the utilization of bilinear pairing. Similarly, Zheng et al. [11] introduced an authenticated key agreement scheme based on bilinear pairing. Ou et al. [12] later showed that Zheng et al.’s scheme is susceptible to impersonation attacks, and proposed a more secure authenticated key agreement scheme; however, the computational cost of this scheme is again somewhat high because of the utilization of bilinear pairing. In addition, an authentication scheme with access control for VANETs was investigated by Yeh et al. [13]; however, Horng et al. [14] later showed that Yeh et al.’s scheme [13] is susceptible to privilege escalation attacks.

Recently, Chuang and Lee [15] developed a trust-extended authentication mechanism, called TEAM, for VANETs. In TEAM, vehicles are classified into three types, namely, law executors (LEs), mistrusted vehicles (MVs) and trusted vehicles (TVs), as shown in Figure 1. Moreover, it required each vehicle is equipped with a tamper-proof device from which no attacker can extract any stored data, which is so strong that it is not practical. The performance of this mechanism in response to several types of attacks has been analyzed; however, the linkability of messages in the authentication procedure and the possibility of internal attacks during the secure communication procedure, which can easily be executed by a malicious vehicle, have been ignored. A malicious vehicle can trace a driver by intercepting the message sent during the authentication procedure because the values $D_i$ and $M_4$ are constant. Moreover, a malicious trusted vehicle can compute the real identity of a user and the session key by intercepting a message communicated via the secure communication procedure because it possesses the authorized parameter. Kumari et al. [16] proposed an enhanced trust-extended authentication scheme based on TEAM. However, their scheme fails to protect against internal attacks. Therefore, we have developed an improved authentication procedure and secure communication procedure and have proven their correctness and security. The updating of the constant values used in the authentication procedure is performed by the user himself. Finally, we analyze the computational costs and security features of the improved secure communication procedure.

3. Preliminaries

3.1. Security Model

To accurately capture the capabilities of an attacker, an experiment concerning the interaction between an adversary and a challenger is introduced. The random oracle model, which originates from the work of Bellare et al. [17], is adopted in our security proof. An adversary $A$ can be allowed to communicate with the participants through defined oracle queries; thus, the adversary’s behavior during a real attack can be modeled. In our proposed protocol, each participant is either a common vehicle’s OBU $V_i$ or an LE $E_i$. Let $U$ represent all participants that is the union of common vehicle’s OBUs and LEs.
3.1.1. Protocol Execution

Let $U^i_i$ represent the $i$th instance of a participant $U_i$ and let $b$ denote a randomly chosen bit. All possible oracle queries are described as follows:

- **Execute**$(V^i_i, U^i_i)$: The passive attack capability of the adversary $A$ is tested by this query. Executing this query will output an honest execution transcript of the protocol.
- **Send**$(U^i_i, M)$: The active attack capability of the adversary $A$ is tested by this query. $A$ can send a Send request on a message $M$ to $U^i_i$. Upon receiving this message, $U^i_i$ proceeds with the normal execution of the protocol, and then returns the calculated result to the adversary $A$.
- **Corrupt**$(V^i_i)$: This query models an attack that steals a vehicle’s OBU attack. Upon execution of this query, all the information stored in the OBU of vehicle $U^i_i$ will be extracted by $A$.
- **Reveal**$(U^i_i)$: This query models a known key attack. If a session key has been obtained by $U^i_i$, then the session key of instance $U^i_i$ is returned to $A$. Otherwise, $\bot$ is returned.
- **Test**$(U^i_i)$: This query models the ability of the adversary $A$ to distinguishing a real session key from a random key. If the session key of participant $U^i_i$ has not been defined, $\bot$ will be returned. Otherwise, if $b = 1$, then the session key of instance $U^i_i$ will be returned; if $b = 0$, a random key of the same size will be returned.

3.1.2. Notation

An instance $U^i_i$ is said to have been opened if $A$ has issued a query **Reveal**$(U^i_i)$ to it; otherwise, it is said to be unopened [18]. After receiving the last expected protocol message, $U^i_i$ enters an accept mode and it is said to be accepted.

3.1.3. Partnering

To illustrate the process of partnering, the concept of a session identification code $sid$ is introduced. Given $U_1, U_2 \in OBU$, instances $U^i_1$ and $U^i_2$ are called partners only when the following conditions hold: (1) $U^i_1$ and $U^i_2$ have entered accept mode. (2) The same $sid$ is shared between $U^i_1$ and $U^i_2$. (3) $U^i_1$ and $U^i_2$ are partners of each other.

3.1.4. Freshness

To avoid cases in which the security of the scheme is trivially broken by the adversary, the concept of freshness is introduced. The objective is to only permit the adversary to issue **Test** queries to fresh oracle instances. Specifically, an instance $U^i_i$ is called fresh when it enters accept mode and both $U^i_i$ and its partner are unopened.

3.1.5. Semantic Security

Suppose that an adversary $A$ executes a protocol $P$. $A$ can ask a **Test** query to a fresh instance after being given access to **Execute**, **Send**, **Reveal**, **Corrupt** and **Test** queries, and outputs a guess bit $b'$. If $b' = b$ where $b$ is chosen in the **Test** query, $A$ is said to win this experiment defining semantic security. Let $Succ$ represent the event in which $A$ is successful. The advantage of $A$ in breaking the semantic security of $P$ is defined as follows

$$Adv_{P,D}(A) = 2Pr[Succ] - 1,$$

where the password is selected from a dictionary $D$.

3.2. Elliptic Curve Discrete Logarithm Problem

Let $G$ be an elliptic curve group defined by a generator $P$ and a prime number $p$. Then, the two central mathematical problems in elliptic curve cryptography (ECC), namely, the discrete logarithm problem and the computational Diffie–Hellman assumption, can be defined as follows [19].
Definition 1. Elliptic curve discrete logarithm (ECDL) problem. Let $Q = aP$, where $Q, P \in G$ and $a \in \mathbb{Z}_p^*$. The objective of the elliptic curve discrete logarithm problem is to find $a$ when given two points $Q, P \in G$.

Definition 2. Elliptic curve computational Diffie–Hellman (ECCDH) assumption. Let $G$ denote a representative group of order $p$ and $A$ denote an adversary. Consider the following experiment:

Experiment $\text{Exp}_{\text{ECCDH}}^G(G, P, p)$,

$Q_1 = r_1 P, Q_2 = r_2 P, r_1, r_2 \in \mathbb{Z}_p^*$,

$Q = A_{\text{ECCDH}}(Q_1, Q_2)$,

if $Q = r_1 \cdot r_2 \cdot P, b = 1$, else $b = 0$,

return $b$.

The advantage of $A$ in solving the ECCDH problem is defined as follows:

$$\text{Adv}_{G}^{\text{ECCDH}}(A) = \Pr[\text{Exp}_{G}^{\text{ECCDH}}(G, P, p) = 1],$$

$$\text{Adv}_{G}^{\text{ECCDH}}(t) = \max \left\{ \text{Adv}_{G}^{\text{ECCDH}}(A) \right\},$$

where the maximum is taken over all $A$ with time-complexity at most $t$.

4. Review of the Chuang–Lee’s Scheme

In this section, we review Chuang and Lee’s trust-extended authentication scheme (TEAM) [15]. In their scheme, the vehicles are classified into three types, namely, law executors (LEs), mistrusted vehicles (MVs) and trusted vehicles (TVs), as shown in Figure 1. An LE, such as a police vehicle, is treated as permanently trusted and plays a role similar to that of a mobile authentication server (AS). When a normal vehicle is authenticated successfully, it is deemed to be trusted, otherwise, it is treated as mistrusted. A TV will turn into an MV once the lifetime of its key has expired. To ensure the security of communication, an OBU can obtain service from providers only if it has been authenticated successfully.

TEAM consists of eight procedures: registration, login, password change, general authentication, trusted-extended authentication, secure communication, key update and key revocation. Before each vehicle joins the network, its OBU performs the registration procedure to register itself with the AS. The login procedure is performed when a vehicle intends to access service from the vehicular ad hoc network. After successfully completing the login procedure, the OBU checks its authentication state. If the vehicle is an MV, it needs to perform either the general authentication procedure or the trust-extended authentication procedure; it will then turn into a TV once it has been authenticated successfully and has obtained an authenticated key. Then, it can play the role of an LE to authenticate other mistrusted OBUS via the trust-extended authentication procedure. Two trusted vehicles can perform the secure communication procedure to interact with each other. A trusted vehicle can choose to perform the key update procedure with an LE when its key is approaching expiration. Otherwise, the state of the TV changes to mistrusted when the lifetime of the key has expired.

The OBU of each vehicle is equipped with secure hardware, including a tamper-proof device (TPD) and an event data recorder (EDR). The TPD hinders an attacker from obtaining information from the OBU. Recording important data, such as public parameters, preloaded secret keys, times, and locations, is the responsibility of the EDR. In addition, each vehicle is synchronized via a GPS device. Finally, each vehicle periodically broadcasts a hello message with its authentication state (mistrusted or trusted). The related notations are briefly defined in Table 1. The details of the TEAM protocol follow.
Table 1. The notations.

| Notation | Definitions |
|----------|-------------|
| $x$      | A private key for the AS |
| $x_i$    | A private key for user $i$ |
| $PSK$    | A pre-shared secure key set among the LEs and the AS |
| $ID_i$   | The identification code for entity $i$ |
| $PW_i$   | The password for user $i$ |
| $AID_i$  | The alias for entity $i$ |
| $h$      | A secure hash function |
| $⊕$      | The XOR operator |
| $∥$      | The combination of strings |
| $P_{pub_i}$ | A public key for user $i$ and $P_{pub_i} = x_iP$ |
| $p$      | A secure large prime |
| $E(F_p)$ | A secure elliptic curve |
| $P$      | The primitive generator for $G$ |
| $G$      | The subgroup of $E(F_p)$ with order $p$ |
| $Z_p^*$  | The set consisting of all primes in $\{0, 1, ..., p - 1\}$ |
| $r, y \in R Z_p^*$ | An element selected randomly from $Z_p^*$ |
| $SK_{ij}$ | A session key between entity $i$ and entity $j$ |
| $MSG_{KU}$ | A key update message |

4.1. Registration

**LE Registration:** In this procedure, an LE registers itself with the AS via the manufacturer or a secure channel. The secure key set $\{PSK_i, i = 1, ..., n\}$ is sent to the LE by the AS. Only this secure key set is required to be stored in the secure hardware of the LE. No other user information needs to be stored. Furthermore, the lifetime of each $PSK_i$ is set to be short for robust security. When the lifetime of each trusted vehicle’s key expires, this vehicle is required to perform the key update procedure with the LE. The procedure for the key set generation is depicted in Figure 2. It can be seen that the old $PSK_i$ (e.g., $PSK_1$) cannot be used to derive the new $PSK_i$ (e.g., $PSK_2$) because a one-way hash function is introduced in the key generation procedure.

$$PSK_n \quad PSK_{n-1} \quad \cdots \quad PSK_2 \quad PSK_1$$

$$h(\text{nonce}) \rightarrow h^2(\text{nonce}) \rightarrow \cdots \rightarrow h^{n-1}(\text{nonce}) \rightarrow h^n(\text{nonce})$$

**Figure 2.** Key set generation scheme based on the hash–chain method.

**Normal Vehicle Registration:** All vehicles except LEs need to perform this procedure when they are delivered to market. This registration procedure is performed only once by each vehicle.

**Step1.** $U_i \rightarrow AS$: A user $U_i$ chooses his password $PW_i$ and sends its public identity $ID_i$ and $PW_i$ to the AS via the manufacturer or a secure channel.

**Step2.** The AS evaluates the following parameters for $U_i$ after it receives $ID_i$ and $PW_i$: $A_i = h(ID_i||x), B_i = h^2(ID_i||x) = h(A_i), C_i = h(PW_i) ⊕ B_i$, and $D_i = PSK ⊕ A_i$.

**Step3.** $AS \rightarrow U_i$: The parameters (i.e., $ID_i, B_i, C_i, D_i, h()$) are stored in the OBU’s secure hardware by the AS via a secure channel.
4.2. Login

The login procedure is performed when a user $U_i$ intends to access the service from vehicle sensor networks. The login procedure is described as follows:

Step1. $U_i \rightarrow OBU_i$: $ID_i$ and $PW_i$ are input to $OBU_i$ by $U_i$.

Step2. First, $OBU_i$ verifies $ID_i$. Then, it checks whether $B_i = h(PW_i) \oplus C_i$ holds. If so, $OBU_i$ launches the general authentication procedure or the trust-extended authentication procedure. Otherwise, the login request will be rejected.

4.3. Password Change

When a user $U_i$ wants to update his password, he invokes the optional password change procedure. The steps of this procedure are described below:

Step1. $ID_i$ and $PW_i$ are input to its $OBU_i$ by $U_i$.

Step2. First, $OBU_i$ verifies $ID_i$ checks whether $B_i = h(PW_i) \oplus C_i$. If so, $U_i$ will be requested to input his new password $PW^*$. $OBU_i$ computes $C_i^* = C_i \oplus h(PW_i) \oplus h(PW^*_i)$ and replaces $C_i$ with $C_i^*$. Otherwise, the request will be rejected.

4.4. General Authentication

The general authentication procedure is performed between $OBU_i$ and $LE_j$ after $U_i$ has completed the login procedure. The steps of this procedure are described below:

Step1. $OBU_i$ chooses a random number $r_i$ and computes its alias $AID_i = h(r_i) \oplus ID_i$. Then, it produces the request messages $M_1 = h(B_i) \oplus r_i$ and $M_2 = h(r_i||AID_i||ID_i)$.

Step2. $OBU_i \rightarrow LE_j$: The authentication messages (i.e., $AID_i,M_1,M_2$ and $D_i$) are sent from $OBU_i$ to $LE_j$.

Step3. Upon receiving the authentication request message (i.e., $AID_i,M_1,M_2,D_i$), $LE_j$ uses $PSK$ to retrieve $A_i = D_i \oplus PSK$ and $r_i = M_1 \oplus h^2(A_i)$ and then checks whether $M_2 = h(r_i||AID_i||D_i)$ holds. The authentication request will be rejected if this equation does not hold. Otherwise, $LE_j$ computes $ID_i = AID_i \oplus h(r_i)$ and produces a random number $r_j$ with which to calculate $AID_j = r_j \oplus ID_j$ and $SK_{ij} = h(r_i||r_j)$. Finally, $LE_j$ calculates the response messages $M_3 = r_j \oplus h^2(r_i)$, $M_4 = A_i \oplus h(ID_i)$ and $M_5 = h(M_4||r_i||AID_j)$.

Step4. $LE_j \rightarrow OBU_i$: $LE_j$ returns its response messages (i.e., $AID_j,M_3,M_4,M_5$) to $OBU_i$.

Step5. $OBU_i$ computes $h^2(r_i)$ to retrieve $r_j = M_3 \oplus h^2(r_i)$ and checks whether $M_5 = h(M_4||r_j||AID_j)$ holds. $OBU_i$ terminates the process if this equation does not hold. Otherwise, $OBU_i$ computes $A_i = M_4 \oplus h(ID_i)$, calculates $SK_{ij} = h(r_i||r_j)$, and stores $A_i$ in its secure hardware.

Step6. $OBU_i \rightarrow LE_j$: The message $SK_{ij} \oplus h(r_j)$ is sent to $LE_j$ by $OBU_i$.

Step7. $LE_j$ uses $SK_{ij}$ to retrieve $h(r_j)$. Then, it checks whether the retrieved hash value is equal to the pre-computed hash value using the chosen $r_j$. In this way, a replay attack from an illegal OBU is avoided.

As this time, the state of $OBU_i$ changes to trusted since $OBU_i$ has been authenticated successfully and has obtained the parameter $PSK = A_i \oplus D_i$. Now, not only $LE$ but also $OBU_i$ can authenticate other mistrusted OBUS.

4.5. Trust-Extended Authentication

A mistrusted OBU becomes trusted once it has been authenticated successfully and has obtained $PSK$. Then, it can play the role of an $LE$ to authenticate other mistrusted OBUS. The corresponding trust-extended authentication procedure is the same as the general authentication procedure.
4.6. Secure Communication

The secure communication procedure is performed between two trusted vehicles OBU\(_i\) and OBU\(_j\) when they intend to interact with each other.

**Step1.** After completing the login procedure, OBU\(_i\) generates a random number \(r_i\) and computes the messages \(AID_i = ID_i \oplus r_i\), \(M_1 = PSK \oplus r_i\), and \(M_2 = PSK \oplus h(AID_i||r_i)\), where PSK was obtained in a previous authentication procedure.

**Step2.** OBU\(_i\) → OBU\(_j\): A secure communication request (i.e., \(AID_i,M_1,M_2\)) is sent to OBU\(_j\) by OBU\(_i\).

**Step3.** Upon receiving (i.e., \(AID_j,M_1,M_2\)), OBU\(_j\) uses PSK to retrieve \(r_i\) from \(M_1\) and then computes \(PSK \oplus h(AID_j||r_i)\) and checks whether it is equal to \(M_2\). The request will be rejected if this equality does not hold. Otherwise, OBU\(_j\) randomly chooses \(r_j\) and computes \(AID_j = ID_j \oplus r_j\), \(M_3 = PSK \oplus r_j\), \(M_4 = PSK \oplus h(AID_j||r_j||h(r_j))\) and a session key \(SK_{ij} = h(r_i||r_j||PSK)\).

**Step4.** OBU\(_j\) → OBU\(_i\): OBU\(_j\) returns the response messages (i.e., \(AID_j,M_3,M_4\)) to OBU\(_i\).

**Step5.** After receiving the messages \(\{AID_j,M_3,M_4\}\), OBU\(_i\) verifies whether OBU\(_i\) is trusted: OBU\(_i\) uses PSK to retrieve \(r_j\) from \(M_3\) and checks whether \(M_4 = h(AID_j||r_j||h(r_j))\) holds. If so, OBU\(_i\) computes a session key \(SK_{ij} = h(r_i||r_j||PSK)\) and a reply message \(M_5 = SK_{ij} \oplus h(r_j)\). Otherwise, the process is terminated.

**Step6.** OBU\(_i\) → OBU\(_j\): OBU\(_i\) sends \((M_5)\) to OBU\(_j\).

**Step7.** After receiving the message \(M_5\), OBU\(_j\) computes \(SK_{ij} \oplus h(r_j)\) and then checks whether it is equal to \(M_5\). If this quality holds, then the two trusted vehicles can communicate securely using \(SK_{ij}\). Otherwise, OBU\(_j\) terminates the process.

4.7. Key Revocation

Key revocation will be triggered when the lifetime of a key expires. The state of a mistrusted vehicle changes to trusted when the mistrusted vehicle is authenticated successfully and obtains PSK via performing either the general authentication procedure or the trust-extended authentication procedure. Then, a timer is instantiated by the secure hardware and begins to count down. The state of the vehicle becomes mistrusted when the lifetime of the key expires. When key expiration is approaching, the system requests that the trusted vehicle performs the key update procedure.

4.8. Key Update

The key update procedure will be invoked by OBU\(_i\) when the key lifetime of the TV is approaching expiration. The steps of this procedure are described as follows.

**Step1.** OBU\(_i\) randomly chooses \(r_i\) to compute the messages \(M_1 = PSK_{old} \oplus r_i\), \(M_2 = PSK_{old} \oplus MSG_{KLI}\), and \(M_3 = h(r_i||MSG_{KLI})\).

**Step2.** OBU\(_i\) → LE\(_j\): A key update request (i.e., \(M_1,M_2,M_3\)) is sent to LE\(_j\) by OBU\(_i\).

**Step3.** LE\(_j\) retrieves \(r_i\) and \(MSG_{KLI}\) using the current PSK (i.e., \(PSK_{old}\)). The key update request will be rejected if \(h(r_i||MSG_{KLI})\) does not match \(M_3\). Otherwise, LE\(_j\) chooses a random number \(r_j\) and computes \(M_4 = r_j \oplus h(r_j)\), \(M_5 = PSK_{new} \oplus r_j\), and \(M_6 = h(r_j||PSK_{new})\), where PSK is produced via the hash-chain method. Therefore, the new PSK cannot be inferred by other OBUs using the current PSK. Finally, LE\(_j\) computes \(SK_{ij} = h(r_i||r_j||PSK_{new})\).

**Step4.** LE\(_j\) → OBU\(_i\): LE\(_j\) returns the reply messages (i.e., \(M_4,M_5,M_6\)) to OBU\(_i\).

**Step5.** Upon receiving the reply messages, OBU\(_i\) computes \(h(r_j)\) to retrieve \(r_j = M_4 \oplus h(r_j)\), and obtains \(PSK_{new} = M_5 \oplus r_j\). Next, OBU\(_i\) checks whether \(M_6 = h(r_j||PSK_{new})\) and \(PSK_{old} = h(PSK_{new})\). If this condition holds, OBU\(_i\) renews the PSK and computes \(SK_{ij} = h(r_i||r_j||PSK_{new})\). Otherwise, OBU\(_i\) terminates the process.

**Step6.** OBU\(_i\) → LE\(_j\): OBU\(_i\) sends the message \(SK_{ij} \oplus h(r_j)\) to LE\(_j\).
Step 7. LE retrieves $h(r_j)$ using $SK_{ij}$. Then, it checks whether the retrieved hash value is equal to the pre-computed hash value using the chosen $r_j$. In this way, a replay attack from an illegal OBU is avoided. Now, this session key can be used to communicate securely between two trusted vehicles.

5. Improved Scheme

A concrete description of our enhanced privacy-preserving authentication scheme is presented in this section. In our scheme, the vehicles are also classified into three types: law executors (LEs), mistrusted vehicles (MVs) and trusted vehicles (TVs) as displayed in Figure 1. The LEs are equipped with TPD, but the normal vehicles such as TV and MV are not equipped with TPD. Our improved scheme consists of nine procedures: initialization, registration, login, password change, general authentication, trust-extended authentication, secure communication, key update and revocation. The notations used in this section are also briefly defined in Table 1.

5.1. Initialization

The initialization procedure is performed by the AS when it sets up the system parameters:

Step 1. Let $G$ be an elliptic curve group defined by a generator $P$ and a prime number $p$. The AS randomly selects $\{x \in Z_p^*\}$ as its secret key.

Step 2. The AS computes the secure key set $\{PSK_i, i = 1, ..., n\}$ using the hash–chain method as shown in Figure 2, e.g., $h^2(x) = h(h(x))$.

5.2. Registration

**LE Registration:** In this procedure, an LE registers itself with the AS via the manufacturer or a secure channel. The secure key set $\{PSK_i, i = 1, ..., n\}$ and the public parameters $\{G, p, P\}$ are sent to the LE by the AS. Only the secure key set and the public parameters are required to be stored in the secure hardware of the LE. No other user information needs to be stored. Similarly, the lifetime of each $PSK_i$ is set to be short for robust security. When the lifetime of each trusted vehicle’s key expires, this vehicle is required to perform the key update procedure with an LE.

**Normal Vehicle Registration:** All vehicles except LEs need to perform this procedure when they are delivered to market. This registration procedure is performed only once by each vehicle. The steps of the normal vehicle registration procedure are described in Figure 3.

Step 1. $U_i \rightarrow AS$: A user $U_i$ chooses his password $PW_i$ and sends its public identity $ID_i$ and $PW_i$ to the AS via the manufacturer or a secure channel.

Step 2. The AS chooses a random number $y_i$ with which to evaluate the following parameters for $U_i$ after it receives $ID_i$ and $PW_i$: $A_i = h(ID_i||x), B_i = h(PW_i) \oplus A_i, C_i = h(PSK||y_i) \oplus A_i,$ and $D_i = h(ID_i||PW_i||A_i)$.

Step 3. $AS \rightarrow U_i$: The parameters (i.e., $B_i, C_i, D_i, y_i, h(), G, p, P$) are stored in the OBU’s secure hardware by the AS via a secure channel.

Step 4. $U_i$ chooses a number $x_i$ as his private key and computes $P_{pub_i} = x_iP$ as his public key, and then computes $Z_i = x_i \oplus h(PW_i)$ and stores $(P_{pub_i}, Z_i)$ in its OBU secure hardware.
5.3. Login

The login procedure is performed when a user \( U_i \) intends to access service from the vehicle sensor network. The login procedure is described as follows:

Step1. \( U_i \rightarrow OBU_i \); ID, and PW are input to OBU, by \( U_i \).

Step2. First, OBU, retrieves \( A_i = h(PW_i) \oplus B_i \). Then, it checks whether \( D_i = h(ID_i||PW_i||A_i) \) holds. If so, OBU, launches the general authentication procedure or the trust-extended authentication procedure. Otherwise, the login request will be rejected.

5.4. Password Change

When a user \( U_i \) wants to update his password, the optional password change procedure will be invoked. The steps of this procedure are described as follows:

Step1. \( ID_i \) and \( PW_i \) are input to OBU, by \( U_i \).

Step2. First, OBU, retrieves \( A_i = h(PW_i) \oplus B_i \). Then, it checks whether \( D_i = h(ID_i||PW_i||A_i) \) holds. If so, \( U_i \) will be requested to input his new password \( PW_i^* \). OBU, computes \( B_i^* = B_i \oplus h(PW_i) \oplus h(PW_i^*) \) and \( D_i^* = h(ID_i||PW_i^*||A_i) \) and replaces \( B_i \) and \( D_i \) with \( B_i^* \) and \( D_i^* \). Otherwise, the request will be rejected.

5.5. General Authentication

The general authentication procedure is performed between OBU, and LE, after \( U_i \) has completed the login procedure. The general authentication procedure is shown in Figure 4 and the steps are described as follows.

Step1. OBU, chooses a random number \( r_i \) and computes its alias \( AID_i = h(r_i) \oplus ID_i \). Then, it produces the request messages \( M_1 = h(A_i) \oplus r_i \) and \( M_2 = h(r_i||AID_i||C_i||y_i) \), where \( A_i \) is obtained from the login procedure.

Step2. OBU, \( \rightarrow \) LE,: The authentication messages (i.e., \( AID_i, M_1, M_2, C_i, \) and \( y_i \)) are sent from OBU, to LE,.

Step3. Upon receiving the authentication request messages (i.e., \( AID_i, M_1, M_2, C_i, \) and \( y_i \)), LE, uses PSK to retrieve \( A_i = C_i \oplus h(PSK||y_i) \) and \( r_i = M_1 \oplus h(A_i) \) and then checks whether \( M_2 = h(r_i||AID_i||C_i||y_i) \) holds. The authentication request will be rejected if it does not. Otherwise, LE, produces a random number \( r_j \) to calculate \( AID_j = ID_j \oplus h(r_j) \) and \( SK_{ij} = h(r_i||r_j) \). Finally, LE, calculates the response messages \( M_3 = r_j \oplus h^2(r_j), M_4 = PSK \oplus r_j, \) and \( M_5 = h(AID_j||SK_{ij}||r_j||PSK) \).

Step4. LE, \( \rightarrow \) OBU, : LE, return response messages (i.e., \( AID_j, M_3, M_4, \) and \( M_5 \)) to OBU,.

Step5. OBU, computes \( h^2(r_j) \) to retrieve \( r_j = M_3 \oplus h^2(r_j), PSK = M_4 \oplus r_j, \) and \( SK_{ij} = h(r_i||r_j) \) and checks whether \( M_5 = h(AID_j||SK_{ij}||r_j||PSK) \) holds. OBU, terminates the process if
it does not. Otherwise, OBU\textsubscript{i} calculates the reply message \( M_6 = SK_{ij} \oplus h(r_j); \) computes \( C_{\text{new}} = h(PSK||r_i) \oplus A_i \) and \( E_i = h(PW_i) \oplus PSK; \) replaces \( C_i \) and \( y_i \) with \( C_{\text{new}} \) and \( r_i \), respectively, and stores \( E_i \) in its secure hardware.

**Step6.** \( OBU_i \rightarrow LE_j; \) The message \( M_6 \) is sent to to \( LE_j \) by \( OBU_i. \)

**Step7.** \( LE_j \) uses \( SK_{ij} \) to retrieve \( h(r_j). \) Then, it checks whether the retrieved hash value is equal to the pre-computed hash value using the chosen \( r_j. \) In this way, a replay attack from an illegal OBU is avoided.

\[
\begin{align*}
\text{OBU}_i & \quad \text{LE}_j \\
AID_i = h(r) \oplus ID_i \\
M_1 = h(A) \oplus r_i \\
M_2 = h(r_i \parallel AID_i \parallel C_i \parallel y_i) & \quad AID_j, M_1, M_2, C_i, y_i & \quad A = h(PSK \parallel y_j) \oplus C_j \\
r_i = M_1 \oplus h(A_i) \\
h(r_i \parallel AID_i \parallel C_i \parallel y_i) = ? M_2 \\
AID_j = h(r_j) \oplus ID_j \\
SK_j = h(r_j) \\
M_j = r_j \oplus h^2(r_j) \\
M_j = PSK \oplus r_j \\
M_j = h(AID_j \parallel SK_j \parallel r_j \parallel PSK) \quad \text{compute } h^2(r) \\
M_i = PSK \oplus h(r_i) \quad \text{keep PSK, update } C_i \\
M_6 = ? SK_j \oplus h(r_j) \\
\end{align*}
\]

**Figure 4.** General authentication procedure.

At this time, the state of \( OBU_i \) changes to trusted since \( OBU_i \) has been authenticated successfully and has obtained the parameter \( PSK. \) Now, not only \( LE \) but also \( OBU_i \) can authenticate other mistrusted OBUs.

5.6. Trust-Extended Authentication

This procedure is the same as in the Chuang–Lee scheme.

5.7. Secure Communication

The secure communication procedure is performed between two trusted vehicles \( OBU_i \) and \( OBU_j \) when they intend to interact with each other. The secure communication procedure is shown in Figure 5 and the steps are described as follows.

**Step1.** After completing the login procedure, \( OBU_i \) retrieves \( PSK = E_i \oplus h(PW_i), x_i = Z_i \oplus h(PW_i), \) then it generates a random number \( r_i \) and computes the messages \( AID_i = ID_i \oplus h(r_i P_{pub}), T = r_i P, u_i = T + PSK \cdot P, \) and \( M_1 = h(T||ID_i||AID_i) \), where \( E_i \) was obtained from a previous authentication procedure.

**Step2.** \( OBU_i \rightarrow OBU_j: \) A secure communication request (i.e., \( AID_i, u_i, M_1 \)) is sent to \( OBU_j \) by \( OBU_i. \)

**Step3.** Upon receiving (i.e., \( AID_i, u_i, M_1 \)), \( OBU_j \) uses \( PSK \) to retrieve \( T \) from \( u_i \) and then computes \( ID_i = AID_i \oplus h(x_j T), \) and checks whether \( M_1 \) is equal to \( h(T||ID_i||AID_i). \) The request
will be rejected if this equality does not holds. Otherwise, OBU\textsubscript{j} randomly chooses \( r_j \) and computes
\[
\text{AID}_j = ID_j \oplus h(r_jP_{pub}),
\]
\[
R = r_jP,
\]
\[
u_j = R + PSK \cdot P,
\]
\[
s = r_jP_{pub} + x_jT,
\]
\[
k = h(T \| R \| P_{pub} \| P_{pub} \| s),
\]
\[
M_2 = h(ID_j \| k).
\]

Step 4. OBU\textsubscript{j} → OBU\textsubscript{i}: OBU\textsubscript{j} returns the response messages (i.e., AID\textsubscript{j}, u\textsubscript{j}, M\textsubscript{2}) to OBU\textsubscript{i}.

Step 5. After receiving the messages \{AID\textsubscript{j}, u\textsubscript{j}, M\textsubscript{2}\}, OBU\textsubscript{i} verifies whether OBU\textsubscript{j} is trusted: OBU\textsubscript{i} computes
\[
R = u_j - PSK \cdot P,
\]
\[
ID_j = AID_j \oplus h(x_jT),
\]
\[
s = r_jP_{pub} + x_jT,
\]
\[
k = h(T \| R \| P_{pub} \| P_{pub} \| s),
\]
\[
M_2 = h(ID_j \| k).
\]

Step 6. OBU\textsubscript{i} → OBU\textsubscript{j}: OBU\textsubscript{i} sends \( M_3 \) to OBU\textsubscript{j}.

Step 7. After receiving the message \( \{M_3\} \), OBU\textsubscript{j} checks whether \( M_3 = h(u_j \| k) \) holds. If so, the two trusted vehicles can communicate securely using \( k \). Otherwise, OBU\textsubscript{j} terminates the process.

![Secure communication procedure](image-url)
improved scheme. Finally, we evaluate the performance of our scheme against that of the existing related schemes.

6.1. Correctness

The BAN logic is a useful way to validate the correctness of security protocols, especially for the authentication protocols [20]. Some relevant notations are listed in Table 2. The verification procedure consists of the following steps.

Table 2. Symbol and description of BAN logic.

| Symbol | Description |
|--------|-------------|
| $P \equiv X$ | Entity $P$ trusts opinion $X$ |
| $P \triangleleft X$ | Entity $P$ sees opinion $X$, or $P$ holds $X$ |
| $P \triangleright X$ | Entity $P$ has said opinion $X$ |
| $\tilde{P}(X)$ | $X$ is fresh |
| Rule2 | Rule2 comes from Rule1 |
| $\mapsto P$ | $k$ is the public key of entity $P$ |
| $\{X\}_{PSK}$ | $X$ is encrypted by key $K$ |

6.1.1. The Correctness of the General Authentication Procedure

Idealization

First, we use formal logical language to idealize the general authentication procedure in our improved scheme in accordance with the rules of the BAN logic as follows:

1. $OBU_i \rightarrow LE_j : \{M_1 = h(r_i || AID_i || C_i || y_i), AID_i, \{r_i\}_{A_i}, \{A_i\}_{PSK}\}$.
2. $LE_j \rightarrow OBU_i : \{M_2 = h(AID_j || SK_{ij} || r_j || PSK), AID_j, \{r_j\}_{r_j}, \{PSK\}_{r_j}\}$.
3. $OBU_i \rightarrow LE_j : \{M_3 = SK_{ij} \oplus h(r_j)\}$.

Goal

There are two roles in the general authentication procedure: $OBU_i$ and $LE_j$. Since $OBU_i$ needs to obtain the authorized parameter $PSK$ from the $LE_j$, it must believe $PSK$. Moreover, $OBU_i$ and $LE_j$ must believe each other and each other’s aliases, and they must believe the session key computed in the general authentication procedure. Thus, there are five goals of the general authentication procedure in our improved scheme as follows:

G1. $OBU_i \equiv LE_j \equiv PSK$: $OBU_i$ believes $PSK$.

G2. $OBU_i \equiv LE_j \equiv AID_j$: $OBU_i$ believes $LE_j$ and his alias $AID_j$.

G3. $LE_j \equiv OBU_i \equiv AID_i$: $LE_j$ believes $OBU_i$ and his alias $AID_i$.

G4. $OBU_i \equiv OBU_i^{SK_{ij}} \leftrightarrow LE_j$: $OBU_i$ believes the share key between himself and $LE_j$.

G5. $LE_j \equiv LE_j^{SK_{ij}} \leftrightarrow OBU_i$: $LE_j$ believes the share key between himself and $OBU_i$.

Assumptions

With the goals set, the assumptions also need to be stated as follows:
A1. OBU \_i \triangleleft \text{AID}_i; \ OBU_i \text{ possesses an alias AID}_i.

A2. LE_j \triangleleft \text{AID}_j; \ OBU_j \text{ possesses an alias AID}_j.

A3. OBU \_i \equiv z(r_i, r_j): \ OBU_i \text{ believes the freshness of } r_i \text{ and } r_j.

A4. LE_j \equiv z(r_i, r_j, y_i): \ LE_j \text{ believes the freshness of } r_i, r_j \text{ and } y_i.

A5. LE_j \equiv LE_j \overset{PSK}{\leftrightarrow} \text{OBU}_i; \ LE_j \text{ believes the share key PSK between himself and OBU}_i.

A6. OBU_i \equiv OBU_j \overset{r_j}{\leftrightarrow} \text{LE}_j; \ OBU_i \text{ believes the share key } r_j \text{ between himself and LE}_j.

A7. LE_j \equiv LE_j \overset{r_j}{\leftrightarrow} \text{OBU}_i; \ LE_j \text{ believes the share key } r_j \text{ between himself and OBU}_i.

Verification

In this subsection, we will verify the correctness of our proposed general authentication procedure using the BAN logic. The detailed steps of the proof are as follows:

OBU \_i \text{ computes AID}_i \text{ and } \{r_j\}_A:

V1. \[
\begin{align*}
LE_j \overset{}{\triangleleft} \text{AID}_j(A_i, \text{PSK}, M_2, y_i, LE_j) &\equiv LE_j \overset{PSK}{\leftrightarrow} \text{OBU}_i, \\
&\text{OBU}_i \equiv \text{OBU}_j \overset{M}{\sim} \text{LE}_j.
\end{align*}
\]

V2. \[
\begin{align*}
LE_j \equiv (r_i, y_i, LE_j) &\equiv \text{OBU}_i \overset{M}{\sim} \text{LE}_j, \\
&\text{LE}_j = \text{OBU}_i \overset{M}{\sim} \text{AID}_i.
\end{align*}
\]

V3. \[
\begin{align*}
LE_j \equiv \text{OBU}_i \overset{M}{\sim} \text{LE}_j. \\
&\text{LE}_j = \text{OBU}_i \overset{M}{\sim} \text{AID}_i.
\end{align*}
\]

LE \_j \text{ computes } LE_j \overset{SK_{ij}}{\leftrightarrow} \text{OBU}_i, \{r_j\}_r, \{/PSK\}_r:

V4. \[
\begin{align*}
\text{OBU}_i \equiv \text{AID}_i \{r_j\}_r, \{PSK\}_r, M_5, \text{OBU}_i \equiv \text{OBU}_j \overset{r_j}{\leftrightarrow} \text{LE}_j, \\
&\text{OBU}_i \equiv \text{LE}_j \overset{M}{\sim} M_5.
\end{align*}
\]

V5. \[
\begin{align*}
\text{OBU}_i \equiv \{r_j, r_i\}, \text{OBU}_i \equiv \text{LE}_j \overset{r_j}{\sim} M_5, \\
&\text{OBU}_i \equiv \text{LE}_j \overset{M}{\sim} M_5.
\end{align*}
\]

V6. \[
\begin{align*}
\text{OBU}_i \equiv \text{LE}_j \overset{M}{\sim} M_5, \\
&\text{OBU}_i \equiv \text{LE}_j \overset{r_j}{\sim} M_5.
\end{align*}
\]

V7. \[
\begin{align*}
\text{OBU}_i \equiv \{r_j, r_i\}, \\
&\text{OBU}_i \equiv \{SK_{ij}\}.
\end{align*}
\]

V8. \[
\begin{align*}
\text{OBU}_i \equiv \{SK_{ij}\}, \text{OBU}_i \equiv \text{LE}_j \overset{r_j}{\sim} M_5, \\
&\text{OBU}_i \equiv \text{OBU}_j \overset{r_j}{\leftrightarrow} \text{LE}_j.
\end{align*}
\]

V9. \[
\begin{align*}
\text{OBU}_i \equiv \text{LE}_j \overset{M}{\sim} M_5, \\
&\text{OBU}_i \equiv \text{LE}_j \overset{PSK}{\leftrightarrow} \text{OBU}_i.
\end{align*}
\]

OBU \_i \text{ computes } M_6:

V10. \[
\begin{align*}
\text{OBU}_i \equiv \text{LE}_j \overset{M_6}{\sim} M_5, \\
&\text{OBU}_i \equiv \text{LE}_j \overset{PSK}{\leftrightarrow} \text{OBU}_i.
\end{align*}
\]

V11. \[
\begin{align*}
\text{LE}_j \overset{SK_{ij}}{\leftrightarrow} M_6, \text{LE}_j \equiv \text{LE}_j \overset{M_6}{\leftrightarrow} \text{OBU}_i, \\
&\text{OBU}_i \equiv \text{OBU}_j \overset{M_6}{\sim} M_6.
\end{align*}
\]

V12. \[
\begin{align*}
\text{LE}_j \equiv \{r_j\}, \text{LE}_j \equiv \text{OBU}_j \overset{M_6}{\sim} M_6, \\
&\text{LE}_j \equiv \text{OBU}_j \overset{M_6}{\sim} M_6.
\end{align*}
\]

V13. \[
\begin{align*}
\text{LE}_j \equiv \text{OBU}_j \overset{M_6}{\sim} M_6, \\
&\text{LE}_j \equiv \text{OBU}_j \overset{M_6}{\sim} M_6.
\end{align*}
\]
In formula V3 and formulas V9 and V10, $LE_j$ believes that $OBU_i$ has sent $M_2$ and $OBU_i$ believes that $LE_j$ has sent $M_5$. Because $LE_j$ has verified the correctness of message $M_2$ and $OBU_i$ has verified the correctness of message $M_5$, $OBU_i$ and $LE_j$ each believe the other party and its alias, and $OBU_i$ believes the $PSK$ obtained from $LE_j$. In formula V8, because $OBU_i$ is able to calculate $r_j$ and believes this value which is necessary to compute $SK_{ij}$, $OBU_i$ believes the freshness of $SK_{ij}$, and $OBU_i$ believes the session key $SK_{ij}$ that it computes. Similarly, in formula V15, $LE_j$ believes the value $r_j$ and the freshness of $SK_{ij}$, thus $OBU_i$ believes the session key $SK_{ij}$ that it computes. According to formulas V3, V8, V9, V10 and V15, we can infer that our improved general authentication procedure achieves our goals.

6.1.2. The Correctness of the Secure Communication Procedure

Idealization

First, we use formal logical language to idealize the secure communication procedure in our improved scheme in accordance with the rules of the BAN logic as follows:

1. $OBU_i \rightarrow OBU_j : \{ M_1 = h(r_i, P||ID_i, \{ ID_i \}_{pub_i}, \{ ID_j \}_{pub_j}, \{ r_i P \}_{PSK} ) \}$
2. $OBU_i \rightarrow OBU_j : \{ M_2 = h(ID_j || k), \{ ID_j \}_{pub_j}, \{ r_j P \}_{PSK} ) \}$
3. $OBU_i \rightarrow OBU_j : \{ M_3 = h(\{ r_j P \}_{PSK} || k ) \}$

Goal

There are two roles in the secure communication procedure: $OBU_i$ and $OBU_j$, which are the on-board units of the two communication vehicles. Since $OBU_i$ and $OBU_j$ need to generate a common session key for their communication, they must believe each other and each other’s identities, and they must believe the session key computed in the secure communication procedure. Thus, there are four goals of the secure communication procedure in our improved scheme as follows:

G1. $OBU_i \models OBU_j \models ID_j; OBU_i$ believes $OBU_j$ and its identity $ID_j$.

G2. $OBU_j \models OBU_i \models ID_i; OBU_j$ believes $OBU_i$ and its identity $ID_i$.

G3. $OBU_i \models OBU_i \xrightarrow{k} OBU_j; OBU_i$ believes the shared key between itself and $OBU_j$.

G4. $OBU_j \models OBU_j \xleftarrow{k} OBU_i; OBU_j$ believes the shared key between itself and $OBU_i$.

Assumptions

With the goals set, the assumptions also need to be stated as follows:

A1. $OBU_i \triangleleft ID_i; OBU_i$ owns its identity $ID_i$.

A2. $OBU_j \triangleleft ID_j; OBU_j$ owns its identity $ID_j$.

A3. $OBU_i \triangleleft x_i; OBU_i$ holds own private key $x_i$.

A4. $OBU_j \triangleleft x_j; OBU_j$ holds own private key $x_j$. 
A5. \( \text{OBU}_i \models P_{\text{pub}} \rightarrow \text{OBU}_j; \text{OBU}_i \) believes own public key \( P_{\text{pub}} \).

A6. \( \text{OBU}_j \models P_{\text{pub}} \rightarrow \text{OBU}_j; \text{OBU}_j \) believes own public key \( P_{\text{pub}} \).

A7. \( \text{OBU}_i \triangleleft (P_{\text{pub}}, P_{\text{pub}}); \text{OBU}_i \) holds own public key \( P_{\text{pub}} \) and \( \text{OBU}_j \)'s public key \( P_{\text{pub}} \).

A8. \( \text{OBU}_j \triangleleft (P_{\text{pub}}, P_{\text{pub}}); \text{OBU}_j \) holds own public key \( P_{\text{pub}} \) and \( \text{OBU}_i \)'s public key \( P_{\text{pub}} \).

A9. \( \text{OBU}_i \models \exists (r_i, r_j); \text{OBU}_i \) believes the freshness of \( r_i \) and \( r_j \).

A10. \( \text{OBU}_j \models \exists (r_i, r_j); \text{OBU}_j \) believes the freshness of \( r_i \) and \( r_j \).

A11. \( \text{OBU}_i \models \text{OBU}_i \xrightarrow{\text{PSK}} \text{OBU}_j; \text{OBU}_i \) believes the share key \( \text{PSK} \) between himself and \( \text{OBU}_j \).

A12. \( \text{OBU}_j \models \text{OBU}_j \xrightarrow{\text{PSK}} \text{OBU}_i; \text{OBU}_j \) believes the share key \( \text{PSK} \) between himself and \( \text{OBU}_i \).

Verification

In this subsection, we will verify the correctness of our proposed secure communication procedure using the BAN logic. The detailed steps of the proof are as follows:

\( \text{OBU}_i \) computes \( \{1D_i\}P_{\text{pub}} \) and \( \{r_iP\}_{\text{PSK}} \)

\[
\begin{align*}
\text{V1. } & \text{OBU}_i \models \rightarrow \text{OBU}_j, \text{OBU}_j \triangleleft \{1D_i\}P_{\text{pub}}, \\
& \text{OBU}_i \triangleleft \text{ID}_j \\
\text{V2. } & \text{OBU}_i \triangleleft \{r_iP\}_{\text{PSK}}, \text{ID}_n, M_1, \text{OBU}_j \equiv \text{OBU}_i \xrightarrow{\text{PSK}} \text{OBU}_j, \\
& \text{OBU}_i \models \text{OBU}_i \sim M_1 \\
\text{V3. } & \text{OBU}_i \models \{r_i\}, \text{OBU}_j \equiv \text{OBU}_i \sim M_1, \\
& \text{OBU}_i \equiv \text{OBU}_i \equiv M_1 \\
\text{V4. } & \text{OBU}_j \models \text{OBU}_i \equiv M_1, \\
& \text{OBU}_j \equiv \text{OBU}_i \equiv \text{ID}_j.
\end{align*}
\]

\( \text{OBU}_j \) computes \( \text{OBU}_j \xleftarrow{k} \text{OBU}_i, \{r_iP\}_{\text{PSK}} \)

\[
\begin{align*}
\text{V5. } & \text{OBU}_i \models \rightarrow \text{OBU}_j, \text{OBU}_j \triangleleft \{1D_i\}P_{\text{pub}}, \\
& \text{OBU}_j \triangleleft \text{ID}_j \\
\text{V6. } & \text{OBU}_i \triangleleft \{r_iP\}_{\text{PSK}}, r_i, r_j, P_{\text{pub}}, P_{\text{pub}}, M_2, \text{OBU}_i \equiv \text{OBU}_i \equiv \text{OBU}_i, \\
& \text{OBU}_j \models \text{OBU}_j \sim M_2, \\
\text{V7. } & \text{OBU}_i \models \exists (r_i, r_j), \text{OBU}_i \equiv \text{OBU}_i \sim M_2, \\
& \text{OBU}_i \equiv \text{OBU}_i \equiv M_2 \\
\text{V8. } & \text{OBU}_i \equiv \text{OBU}_i \equiv M_2, \\
& \text{OBU}_i \equiv \text{OBU}_i \equiv \text{ID}_j, \\
\text{V9. } & \text{OBU}_i \equiv \text{OBU}_i \equiv M_2, \\
& \text{OBU}_i \equiv \text{OBU}_i \equiv s \\
\text{V10. } & \text{OBU}_i \models \exists (r_i, r_j), \\
& \text{OBU}_i \equiv \text{OBU}_i \equiv \text{ID}_j \\
\text{V11. } & \text{OBU}_i \models \exists (k), \text{OBU}_i \equiv \text{OBU}_i \equiv s, \\
& \text{OBU}_i \equiv \text{OBU}_i \leftarrow \text{OBU}_j.
\end{align*}
\]

\( \text{OBU}_i \) computes \( M_3 \).
Theorem 1. \[ V \text{party is a trusted vehicle.} \]

To complete the proof, four experiments are constructed, where the first one transmitted in the network are all associated with a participant’s identity, a temporary secret random number, and the other party is a trusted vehicle. In formula 8, \[ V \text{party is a trusted vehicle.} \]

Simultaneously, it can query the \( h \) oracle. However, all executions are halted where a collision occurs when simulating the \( Send \) and the \( h \) oracle. \( A \) issues \( Send \) to try to deceive the other participants into accepting a modified message. Simultaneously, it can query the \( h \) oracle to verify whether a hash collision exists. Since the messages transmitted in the network are all associated with a participant’s identity, a temporary secret random number, and the other party is a trusted vehicle.

In formula 4 and formula 8, \( V_j \) believes that \( OBU_j \) has sent \( M_1 \) and \( OBU_j \) believes that \( OBU_j \) has sent \( M_2 \). Because \( OBU_j \) has verified the correctness of message \( M_1 \) and \( OBU_j \) has verified the correctness of message \( M_2 \), \( OBU_i \) and \( OBU_j \) each believe the other’s identity and that the other party is a trusted vehicle. In formula 11, because \( OBU_i \) can use its private key to obtain \( ID_j \) and calculate \( k \), \( OBU_i \) can verify \( M_2 \) by means of \( ID_j \) and \( k \); thus, \( OBU_i \) believes the session key \( k \) that it computes. Similarly, in formula 16, \( OBU_i \) can compute the session key \( k \) to verify \( M_3 \), so \( OBU_j \) believes the session key \( k \) that it computes. According to formulas 4, 8, 14 and 16, we can infer that our improved secure communication procedure achieves our goals.

6.2. Security Analysis

In this section, the security proof of the critical secure communication procedure and general authentication procedure is presented. We show that the proposed improved protocol is secure through a formal security analysis in the random oracle model as well as an informal security analysis.

6.2.1. The Formal Security Analysis

Theorem 1. Let \( GAP \) denote the general authentication procedure presented in Figure 4. Let \( |\text{Hash}| \) and \( |D| \) denote the range space of the hash function and the size of the password dictionary \( D \), respectively. Finally, let \( A \) represent an adversary within a polynomial time \( t \) against the semantic security of \( GAP \) by issuing \( q_{\text{send}} \) \( Send \) queries, \( q_{\text{exe}} \) Execute queries and \( q_h \) hash queries. Then, we have

\[
\text{Adv}_{GAP,D}(A) \leq q_h^2 |\text{Hash}| + 2q_{\text{send}} |D|.
\]

Proof of Theorem 1. To complete the proof, four experiments are constructed, where the first one simulates a real attack. For every experiment \( Exp_n \), we use an event \( \text{Succ}_n \) to denote the event in which the adversary successfully guesses the bit \( b \) from the \( Test \) query.

Experiment \( Exp_0 \). This experiment simulates an actual attack. According to definition, we have

\[
\text{Adv}_{GAP,D}(A) = 2 \text{Pr}[\text{Succ}_0] - 1. \tag{1}
\]

Experiment \( Exp_1 \). In this experiment, the oracles \( \text{Execute}, \text{Send}, \text{Corrupt}, \text{Reveal}, \text{Test} \) as in an actual attack are simulated. It can be seen that one cannot distinguish this experiment from the actual experiment. Thus,

\[
\text{Pr}[\text{Succ}_1] = \text{Pr}[\text{Succ}_0]. \tag{2}
\]

Experiment \( Exp_2 \). All oracles considered in experiment \( Exp_1 \) are also simulated in this experiment; however, all executions are halted where a collision occurs when simulating the \( Send \) and the \( h \) oracle. \( A \) issues \( Send \) to try to deceive the other participants into accepting a modified message. Simultaneously, it can query the \( h \) oracle to verify whether a hash collision exists. Since the messages transmitted in the network are all associated with a participant’s identity, a temporary secret random number, and the other party is a trusted vehicle.

\[
\begin{align*}
V12. & \quad OBU_{j} \leftarrow \{r,p\}^\text{PSK}, k, M_3, OBU_{i} \equiv OBU_{i}^\text{PSK}, OBU_{j} \not\equiv OBU_{j}^\text{PSK}, OBU_{j} \not\equiv OBU_{j}^\text{PSK}, OBU_{i} \not\equiv OBU_{i}^\text{PSK}, OBU_{j} \not\equiv OBU_{j}^\text{PSK} \rightarrow OBU_{i}, \\
V13. & \quad OBU_{j} \equiv \{r_0, r_1\}, OBU_{i} \equiv OBU_{i}^\text{PSK}, OBU_{j} \equiv OBU_{j}^\text{PSK}, OBU_{i} \not\equiv OBU_{i}^\text{PSK}, OBU_{j} \not\equiv OBU_{j}^\text{PSK} \rightarrow OBU_{i}, \\
V14. & \quad OBU_{j} \equiv \{r_0, r_1\}, OBU_{i} \equiv OBU_{i}^\text{PSK}, OBU_{j} \equiv OBU_{j}^\text{PSK}, OBU_{i} \not\equiv OBU_{i}^\text{PSK}, OBU_{j} \not\equiv OBU_{j}^\text{PSK} \rightarrow OBU_{i}, \\
V15. & \quad OBU_{j} \equiv \{r_0, r_1\}, OBU_{i} \equiv OBU_{i}^\text{PSK}, OBU_{j} \equiv OBU_{j}^\text{PSK}, OBU_{i} \not\equiv OBU_{i}^\text{PSK}, OBU_{j} \not\equiv OBU_{j}^\text{PSK} \rightarrow OBU_{i}, \\
V16. & \quad OBU_{j} \equiv \{r_0, r_1, k\}, OBU_{i} \equiv OBU_{i}^\text{PSK}, OBU_{j} \equiv OBU_{j}^\text{PSK}, OBU_{i} \not\equiv OBU_{i}^\text{PSK}, OBU_{j} \not\equiv OBU_{j}^\text{PSK} \rightarrow OBU_{i}.
\end{align*}
\]
number and a long-lived key, and the authentication procedure only uses an XOR operation and a hash function, there is no other collision except hash collision. The probability of collision in the $h$ oracle is at most $\frac{q_h^2}{2|\text{Hash}|}$ by the birthday paradox. Hence,

$$|Pr[\text{Succ}_1] - Pr[\text{Succ}_2]| \leq \frac{q_h^2}{2|\text{Hash}|}.$$  

(3)

**Experiment Exp$_3$.** All oracles considered in experiment Exp$_2$ are simulated in this experiment, in addition to stopping the stimulation of a $\text{Corrupt}$ query to an OBU. Note that the information $B_i, C_i, D_i, y_i, Z_i$ and $P_{pub_i}$ stored in the OBU can be extracted by $A$ when the $\text{Corrupt}(U_i)$ query is issued. However, this information is useless to $A$ for calculating the session key since it would also need the secret $A_i$, and it is difficult to derive $A_i$ from $B_i$ without also obtaining the user’s correct password $PW_i$ via the password attack. Hence, we obtain

$$|Pr[\text{Succ}_2] - Pr[\text{Succ}_3]| \leq \frac{q_{\text{send}}}{|D|}.$$  

(4)

In addition, we know that the adversary $A$ can only win the game by guessing the bit $b$ when querying the $\text{Test}$ oracle because the adversary has no advantage. Therefore,

$$Pr[\text{Succ}_3] = \frac{1}{2}.$$  

(5)

From Equations (2) to (5), we have

$$|Pr[\text{Succ}_0] - \frac{1}{2}| = |Pr[\text{Succ}_0] - Pr[\text{Succ}_3]|$$

$$\leq |Pr[\text{Succ}_0] - Pr[\text{Succ}_1]| + |Pr[\text{Succ}_1] - Pr[\text{Succ}_2]|$$

$$+ |Pr[\text{Succ}_2] - Pr[\text{Succ}_3]|$$

$$\leq \frac{q_h^2}{2|\text{Hash}|} + \frac{q_{\text{send}}}{|D|}.$$  

Therefore, from Equation (1), we get

$$Adv_{\text{GAP,D}}(A) \leq \frac{q_h^2}{|\text{Hash}|} + \frac{2q_{\text{send}}}{|D|}.$$  

Table 3. Simulation of random oracles $h$ and $h'$.

| A hash query $h(m)$ (resp.$h'$) that matches a record $(m,r')$ in the list $\Lambda_h$ (resp.$\Lambda_{h'}$), returns $r'$. Otherwise, it chooses a random number $r$, adds the record $(m,r)$ to the list $\Lambda_h$ (resp.$\Lambda_{h'}$), and returns $r$. |

**Theorem 2.** Let $G$ represent a group with a prime order $p$, and SCP denote the secure communication procedure presented in Figure 5. Let $\ell$ be the size of the identity space, $|\text{Hash}|$ and $|D|$ represent the range space of the hash function and the size of the password dictionary $D$. Finally, let $A$ represent an adversary attacking the semantic security of the secure communication protocol with time-complexity at most $t$ by issuing $q_{\text{send}}$ Send queries, $q_{\text{exe}}$ Execute queries and $q_h$ Hash queries. Then, we have:

$$Adv_{\text{SCP,D}}(A) \leq \frac{2(q_{\text{send}} + q_{\text{exe}})}{\ell} + \frac{q_h^2}{|\text{Hash}|} + \frac{(q_{\text{send}} + q_{\text{exe}})^2}{p} + \frac{2q_{\text{send}}}{|D|} + 2q_hAdv_{G}^{\text{ECCDH}}(t + (q_{\text{send}} + q_{\text{exe}}) t_p),$$
where \( t_p \) denotes the time required to produce a point.

**Proof of Theorem 2.** To complete the proof, six experiments are constructed, where the first one simulates a real attack. For every experiment \( \text{Exp}_n \), we use \( \text{Succ}_n \) to denote the event in which the adversary successfully guesses the bit \( b \) from the \( \text{Test} \) query.

**Experiment** \( \text{Exp}_0 \). This experiment simulates an actual attack, which begins with the random selection of a secure key \( \text{PSK} \). According to definition, we have

\[
\text{Adv}_{SCP,D}(A) = 2\text{Pr}[\text{Succ}_0] - 1. \tag{6}
\]

**Experiment** \( \text{Exp}_1 \). In this experiment, the oracles \( \text{Execute, Send, Corrupt, Reveal, and Test} \), as in the actual attack with a chosen random secure key \( \text{PSK} \) are simulated. It can be seen that one cannot distinguish this experiment from the actual experiment. Thus,

\[
\text{Pr}[\text{Succ}_1] = \text{Pr}[\text{Succ}_0]. \tag{7}
\]

**Experiment** \( \text{Exp}_2 \). All oracles considered in experiment \( \text{Exp}_1 \) are also simulated in this experiment. In addition, we stop simulating the adversary to execute guessing attacks on the real identity of a participant. In this case, we have

\[
|\text{Pr}[\text{Succ}_1] - \text{Pr}[\text{Succ}_2]| \leq \frac{q_{\text{send}} + q_{\text{exe}}}{\ell}. \tag{8}
\]

**Proof.** Each participant’s real identity is always converted into an alias using a random number (i.e., \( \text{AID}_i = \text{ID}_i \oplus H(r_i P_{\text{pub}_i}) \)). Therefore, the adversary cannot determine the participant’s real identity because every alias is different and there is nothing that can be used to verify the real identity.

**Experiment** \( \text{Exp}_3 \). All oracles considered in experiment \( \text{Exp}_2 \) are also simulated in this experiment; however, all executions are halted where a collision occurs among \( (\text{AID}_j, u_j, M_1), (\text{AID}_j, u_j, M_2) \), and \( (M_3) \). The probability of colliding in the \( h \) oracle is at most \( q_h^2 / 2 |\text{Hash}| \) by the birthday paradox. Similarly, the probability of colliding in the transcript is at most \( (q_{\text{send}} + q_{\text{exe}})^2 / 2p \). Consequently,

\[
|\text{Pr}[\text{Succ}_2] - \text{Pr}[\text{Succ}_3]| \leq \frac{q_h^2}{2 |\text{Hash}|} + \frac{(q_{\text{send}} + q_{\text{exe}})^2}{2p}. \tag{9}
\]

**Experiment** \( \text{Exp}_4 \). All oracles considered as experiment \( \text{Exp}_3 \) are simulated in this experiment, in addition to stopping the stimulation of a \( \text{Corrupt} \) query to an OBU. Note that the information \( B_i, C_i, D_i, y_i, Z_i, P_{\text{pub}_i} \), and \( E_i \) stored in the OBU can be extracted by \( A \) when the \( \text{Corrupt}(U_i) \) query is issued. However, this information is useless to \( A \) for calculating the session key since it would require the secure key \( \text{PSK} \), a private key \( x_i \) and a temporary secret random number, and it is difficult to derive \( \text{PSK} \) and \( x_i \) from \( E_i \) and \( Z_i \) without obtaining the user’s correct password \( \text{PW}_i \) via the password attack. Hence, we obtain

\[
|\text{Pr}[\text{Succ}_3] - \text{Pr}[\text{Succ}_4]| \leq \frac{q_{\text{send}}}{|D|}. \tag{10}
\]

**Experiment** \( \text{Exp}_5 \). In this experiment, we use the private oracle \( h' \) in place of the oracle \( h \) for computing \( k \) as shown in Table 3, such that the session key is totally independent of \( h \). More precisely, one obtains \( k = h'(T || R || P_{\text{pub}_i} || P_{\text{pub}}) \) in \( \text{Execute} \) queries. Therefore, the experiments \( \text{Exp}_4 \) and \( \text{Exp}_5 \) are indistinguishable except for the occurrence of the following event \( \text{Ask}_{h'} \): \( A \) issues queries to \( h \) on \( T || R || P_{\text{pub}_i} || P_{\text{pub}} || s \), i.e., the value \( T || R || P_{\text{pub}_i} || P_{\text{pub}} || \text{ECCDH}(T, P_{\text{pub}_i}) + \text{ECCDH}(R, P_{\text{pub}}) \).
We randomly select $\alpha$ and $Y$ where $Y = \alpha A - PSK \cdot P$, $R = \beta A - PSK \cdot P$, $P_{pub_i} = \gamma B$, and $P_{pub_j} = \varphi B$. Note that $AskH_6$ means that a query $h$ on $T||R||Y$ has been issued by $A$, where $Y = ECCDH(T, P_{pub_i}) + ECCDH(R, P_{pub_j})$. Indeed, $Pr[AskH_6] = Pr[Succ_6]$. We have:

$$ECCDH(T, P_{pub_i}) = \alpha \varphi \cdot ECCDH(A, B) - \varphi PSK \cdot B,$$

$$ECCDH(R, P_{pub_j}) = \beta \gamma \cdot ECCDH(A, B) - \gamma PSK \cdot B.$$

Therefore,

$$ECCDH(T, P_{pub_i}) + ECCDH(R, P_{pub_j}) = (\alpha \varphi + \beta \gamma) \cdot ECCDH(A, B) - (\varphi + \gamma) PSK \cdot B.$$

If $A$ knows the session key $k$ constructed by $(\alpha A, \beta A, PSK \cdot R, \gamma B, \varphi B)$, it must have issued queries to $h$ on $T||R||P_{pub_i}||P_{pub_j}$ that was recorded in the list $\Lambda_h$. Therefore, we can conclude that

$$Pr[Succ_6] \leq q_h Adv^{ECCDH}_G (t + (q_{send} + q_{exe})t_p).$$

(12)

From Equations (7) to (12), we have

$$Pr[Succ_0] - \frac{1}{2} = Pr[Succ_0] - Pr[Succ_3]$$

$$\leq Pr[Succ_0] - Pr[Succ_1] + Pr[Succ_1] - Pr[Succ_2] + Pr[Succ_2] - Pr[Succ_3]$$

$$+ Pr[Succ_3] - Pr[Succ_4] + Pr[Succ_4] - Pr[Succ_5]$$

$$\leq q_{send} + q_{exe} + \frac{q_{h}^2}{\ell} + \frac{(q_{send} + q_{exe})^2}{2p} + \frac{q_{send}}{|D|}$$

$$+ q_{h} Adv^{ECCDH}_G (t + (q_{send} + q_{exe})t_p).$$

Therefore, from Equation (6), we get

$$Adv_{SCP,D}(A) \leq \frac{2(q_{send} + q_{exe})}{\ell} + \frac{q_{h}^2}{|Hash|} + \frac{(q_{send} + q_{exe})^2}{p} + \frac{2q_{send}}{|D|}$$

$$+ 2q_{h} Adv^{ECCDH}_G (t + (q_{send} + q_{exe})t_p).$$

6.2.2. Informal Security Analysis

Confidentiality of Session Key

In our proposed scheme, when an authentication, secure communication or key update procedure is performed, a session key is generated using two random numbers chosen by the participants. Then, the generated key is used to ensure a secure communication. Moreover, the random numbers used to generate each session key are different. Therefore, it is difficult for an adversary $A$ to successfully guess the session key or derived it from the communicated messages.
Anonymity

In our proposed scheme, to preserve users’ privacy, the original identity of every participant is converted into an alias via an XOR operation with a hash that takes a random number \( r_i \) as an input (i.e., \( \text{AID}_i = ID_i \oplus h(r_i) \), \( \text{AID}_j = ID_j \oplus h(r_j P_{pub}) \)). Therefore, an adversary \( A \) cannot determine a user’s original identity without the random number \( r_i \) or the private key \( x_j \) even if \( T \) has been obtained because of the hardness of the ECCDH problem in \( G \).

Unlinkability

In our proposed scheme, the original identities of the participants are not transmitted over the unsecure network; instead, every participant’s identity is converted into an alias. Moreover, the authentication, secure communication and key update phases are independent of each other. In addition, after every authentication procedure performed by \( \text{OBU}_i \), the value \( C_i \) updates itself. Therefore, for two or more authentication messages that are sent by the same user, the adversary \( A \) cannot determine whether they have the same origin. Thus, \( A \) cannot trace the location of a user by intercepting messages.

Resistance to Impersonation Attack

In the authentication procedure of our improved scheme, if an adversary wishes to impersonate \( \text{OBU}_i \), it must obtain both the \( A_i \) and \( ID_i \) of \( \text{OBU}_i \). Otherwise, it cannot compute a valid authentication request, since the original identity of \( \text{OBU}_i \) is converted into an alias via an XOR operation with a random number \( r_i \) chosen by itself and this random number \( r_i \) is hidden by its \( A_i \). Moreover, the adversary can successfully impersonates \( \text{OBU}_i \) only by correctly guessing the random number, which is difficult because the random number is reselected with each authentication. Furthermore, in the secure communication procedure, the original identity of \( \text{OBU}_i \) is also converted into an alias with a random number \( r_i(AID_i = ID_i \oplus h(r_i P_{pub})) \). The adversary cannot successfully impersonate the OBU since the random number cannot be guessed.

Resistance to Internal Attack

In our proposed scheme, an internal attack refers to the case in which the owner of a vehicle, who possesses the common secure key \( \text{PSK} \), attempts to reveal the session key for a communication channel. Under our improved scheme, in the secure communication procedure, even if the adversary can intercept all exchanged messages, \( (\text{AID}_i, u_i) \) of \( \text{OBU}_i \) and \( (\text{AID}_j, u_j) \) of \( \text{OBU}_j \) and compute \( T \) and \( R \) using the secure key \( \text{PSK} \), it cannot determine the user’s original identity or compute the session key \( k \) under the assumption of the hardness of \( \text{ECCDH} \) problem in \( G \).

6.3. Performance Analysis

In our proposed scheme, the general authentication procedure is based only on an XOR operation and a hash function; thus, the computation cost is low. To demonstrate the performance of the proposed scheme, we compare the the critical secure communication procedure with the existing two-party secure communication schemes with session key agreement [6,11,12,15,16]. Next, we implement our scheme based on cryptographic libraries and present a concrete comparison of execution times. Then, we compare the security features of these schemes. Some notations are defined as follows for convenience:

\( T_h \) : The execution time of a hash function operation.
\( T_{bp} \) : The execution time of a bilinear pairing operation.
\( T_{mul} \) : The execution time of an ECC-based scalar point multiplication operation.
\( T_{add} \) : The execution time of an ECC-based scalar point addition operation.
The detailed comparison is presented in Table 4, where the middle and right columns list the complexity and total execution time, respectively, of each scheme. The transmission time is not considered in the comparison since it depends on the actual characteristics of the network, not the scheme. All operations listed in Table 4 were implemented using the OpenSSL library and the JPBC library, and the experiments were conducted on a Windows 7 PC (Samsung Electronics, Hwaseong, Korea) equipped with an Intel(R) Core(TM) i7-6500U CPU (Santa Clara, CA, USA).

Table 4. The execution time of basic operation.

| Operation | $T_h$ | $T_{mul}$ | $T_{bp}$ | $T_{add}$ | Execution time (ms) |
|-----------|-------|-----------|----------|-----------|---------------------|
|           | 0.004 | 0.326     | 6.28     | 0.038     | 0.004 0.326 6.28 0.038 |

As seen in Table 5 and Figure 6, the execution time of our scheme is less than those of some other schemes [11,12]. Although the execution times of Chuang–Lee’s scheme and Kumari’s scheme are less than that of our scheme, their schemes fail to resist internal attack because the participants’ aliases depend only on a random number that is hidden by $PSK$ as shown in Table 6. Therefore, a trusted vehicle can reveal a participant’s real identity because it holds $PSK$. Meanwhile, because Porambage’s scheme uses certificates for authentication, the unlinkability of messages cannot be preserved, and a user’s anonymity can be violated. Therefore, our proposed scheme is a preferable solution for secure communication in vehicle sensor networks compared with the existing similar schemes presented in [6,11,12,15,16].

Table 5. Comparison of efficiency.

| Scheme       | Computation Cost               | Computation Time (ms) |
|--------------|--------------------------------|-----------------------|
| Reference [6] | $4T_h + 4T_{mul} + 2T_{add}$ | $\approx 1.838$       |
| Reference [11]| $10T_h + 6T_{mul} + 2T_{bp} + 4T_{add}$ | $\approx 14.7$       |
| Reference [12]| $10T_h + 6T_{mul} + 2T_{bp} + 4T_{add}$ | $\approx 14.7$       |
| Reference [15]| $8T_h$                          | $\approx 0.032$       |
| Reference [16]| $10T_h$                        | $\approx 0.04$        |
| Proposed     | $12T_h + 10T_{mul} + 6T_{add}$ | $\approx 3.54$        |

Figure 6. Execution time(ms) of different authentication protocols.
Table 6. Comparison of security features.

| Security Threats and Scheme                  | Ref. [6] | Ref. [11] | Ref. [12] | Ref. [15] | Ref. [16] | Proposed |
|--------------------------------------------|----------|-----------|-----------|-----------|-----------|----------|
| Provides user anonymity                     | ×        | ×         | ×         | ×         | ×         | √        |
| Resistance to user traceability attack     | ×        | ×         | ×         | ×         | ×         | √        |
| Resistance to impersonation attack          | √        | ×         | √         | √         | √         | √        |
| Resist inside attack                        | √        | √         | √         | ×         | ×         | √        |
| Unlinkability of message                   | ×        | √         | √         | ×         | √         | √        |

7. Conclusions

With the emergence of intelligent transportation, the security of vehicle sensor networks is attracting attention from individuals and vehicle manufacturers, and privacy preservation in communication over vehicle sensor networks has become a critical issue. In this paper, we have demonstrated that Chuang and Lee’s TEAM scheme exists the linkability of messages in the authentication protocol; thus, a malicious vehicle can track a driver by intercepting transmitted message. Simultaneously, TEAM scheme can suffer the internal attack in the secure communication protocol; thus, a malicious trusted vehicle can compute the real identity of a user and the session key. To address this shortcoming, an improved authentication scheme based on elliptic curves for better performance and security has been constructed, in which the difficulty of deriving real identities arises from the need to solve an elliptic curve discrete logarithm problem. In this way, privacy preservation is achieved since the real identities of users are protected. The correctness of our proposed scheme has been proven using BAN logic, and a rigorous security proof has been provided based on the random oracle model. In future work, elliptic curves based authentication schemes involving three parities will be investigated.

Acknowledgments: Our work was jointly supported by the National Social Science Foundation of China (No. 14CTQ026), the National Natural Science Foundation of China (No. 61702067, No. 61472464, No. 61672004, No. 61672119), the Chongqing Research Program of Application Foundation and Advanced Technology (No. cstc2017jcyjAX0201), and the Natural Science Foundation of Shandong Province, China (No. ZR2015FL024).

Author Contributions: Yousheng Zhou and Xiaofeng Zhao conceived and designed of the study and wrote the paper; Yi Jiang and Fengjun Shang contributed to perform the experiments and prove, analyze the data; Shaojiang Deng and Xiaojun Wang contributed to analysis tools and helped perform the analysis with constructive discussions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Wang, F.Y.; Zeng, D.; Yang, L. Smart Cars on Smart Roads: An IEEE Intelligent Transportation Systems Society Update. IEEE Pervasive Comput. 2006, 5, 68–69.
2. Bedi, P.; Jindal, V. Use of Big Data technology in Vehicular Ad-hoc Networks. In Proceedings of the 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), New Delhi, India, 24–27 September 2014; pp. 1677–1683.
3. Lee, U.; Zhou B.; Gerla M.; Magistretti, E.; Bellavista, P.; Corradi, A. Mobeyes: Smart mobs for urban monitoring with a vehicular sensor network. IEEE Wirel. Commun. 2006, 13, 52–57.
4. Hu, H.; Lu, R.; Huang, C.; Zhang, Z. TripSense: A Trust-Based Vehicular Platoon Crowdsensing Scheme with Privacy Preservation in VANETs. Sensors 2016, 16, 803, doi:10.3390/s16060803.
5. Wang, L.; Liu, G.; Sun, L. A Secure and Privacy-Preserving Navigation Scheme Using Spatial Crowdsourcing in Fog-Based VANETs. Sensors 2017, 17, 668, doi:10.3390/s17040668.
6. Porambage, P.; Schmitt, C.; Kumar, P.; Gurtov, A.; Ylianttila, M. Two-phase authentication protocol for wireless sensor networks in distributed IoT applications. In Proceedings of the 2014 IEEE Wireless Communications and Networking Conference (WCNC), Istanbul, Turkey, 6–9 April 2014; pp. 2728–2733.
7. Raya, M.; Hubaux, J.P. Securing vehicular ad hoc networks. J. Comput. Secur. 2007, 15, 39–68.
8. Lu, R.; Lin, X.; Zhu, H.; Ho, P.-H.; Shen, X. ECPP: Efficient conditional privacy preservation protocol. In Proceedings of the IEEE International Conference on Computer Communications, Phoenix, AZ, USA, 13–18 April 2008; pp. 1229–1237.

9. Lin, X.; Sun, X.; Ho, P.-H.; Shen, X. CSIS: A secure and privacy-preserving protocol for vehicular communications. *IEEE Trans. Veh. Technol.* 2007, 56, 3442–3456.

10. Zhang, J.; Xu, Y. Privacy-preserving authentication protocols with efficient verification in VANETs. *Int. J. Commun. Syst.* 2014, 27, 3676–3692.

11. Zheng, M.; Zhou, H.; Chen, J. An efficient protocol for two-party explicit authenticated key agreement. *Concurr. Comput.* 2013, 27, 2954–2963.

12. Ruan, O.; Kumar, N.; He, D.; Lee, J.-H. Efficient provably secure password-based explicit authenticated key agreement. *Pervasive Mob. Comput.* 2015, 24, 50–60.

13. Yeh, L.Y.; Chen, Y.C.; Huang J.L. PAACP: A portable privacy-preserving authentication and access control protocol in vehicular ad hoc networks. *Comput. Commun.* 2011, 34, 447–456.

14. Horng, S.J.; Tzeng, S.-F.; Wang, X.; Qiao, S.; Gong, X.; Khan, M.K. Cryptanalysis on a Portable Privacy-Preserving Authentication and Access Control Protocol in VANETs. *Wireless Pers. Commun.* 2014, 79, 1445–1454.

15. Chuang, M.C.; Lee, J.F. TEAM: Trust-Extended authentication mechanism for vehicular ad hoc networks. *IEEE Syst. J.* 2014, 8, 749–758.

16. Kumari, S.; Karuppiah, M.; Li, X.; Wu, F.; Das, A.K. An enhanced and secure trust-extended authentication mechanism for vehicular ad-hoc networks. *Secur. Commun. Netw.* 2016, 9, 4255–4271.

17. Bellare, M.; Pointcheval, D.; Rogaway, P. Authenticated key exchange secure against dictionary attacks. *Tecnol. Electron. Inform.* 2000, 1807, 139–155.

18. Abdalla, M.; Pointcheval, D. Simple Password-Based Encrypted Key Exchange Protocols. In *Topics in Cryptology—CT-RSA*; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3376, pp. 191–208.

19. Lu, R.; Cao, Z.; Chai, Z.; Liang, X. A simple user authentication scheme for grid computing. *Int. J. Netw. Secur.* 2008, 7, 202–206.

20. Lee, C.-C.; Li, C.-T.; Chiu, S.-T.; Lai, Y.-M. A new three-party-authenticated key agreement scheme based on chaotic maps without password table. *Nonlinear Dyn.* 2014, 79, 2485–2495.

© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).