Proven Secure Tree-based Authenticated Key Agreement for Securing V2V and V2I Communications in VANETs

Lu Wei, Jie Cui, Hong Zhong, Yan Xu, Lu Liu

***Abstract*—Vehicular ad hoc networks (VANETs) are vulnerable to many kinds of security attacks, so it is necessary to design an authenticated key agreement (AKA) scheme for securing communication channels in VANETs. Existing AKA schemes in VANETs have not provided an efficient and secure method to secure V2V and V2I communications simultaneously while meeting the necessary security and privacy requirements. Further, few key updating mechanisms, which are secure, conditional privacy-preserving, practical, and lightweight, exist in current VANETs AKA schemes. In this paper, we propose a proven secure AKA scheme for securing V2V and V2I communications in VANETs, which can be divided into two parts. The first part is a three-party authentication process in which vehicles, road side unit (RSU), and trust authority (TA) authenticate each other. The second part is the key agreement process, which is used in the key generation and updating processes. For this phase, we design a tree-based key agreement algorithm that considers two scenarios, i.e., the joining of an authenticated vehicle and the leaving of the vehicle. The formal security proof and the security analysis show that our proposed scheme satisfies session key security and the necessary security requirements in VANETs, respectively. The performance analysis demonstrates that our proposed scheme has an advantage over several representative AKA schemes in VANETs.**

***Index Terms*—VANETs, security, authenticated key agreement, vehicle-to-vehicle, vehicle-to-infrastructure, three-party authentication, key updating.**

I. Introduction

WITH the increasing number of vehicles and the continuous progress of information technology, such as automatic driving and 5G, vehicular ad hoc networks (VANETs) have become increasingly popular topics of study in both academia and industry. There are mainly two types of communication technologies for vehicle-to-everything (V2X) that is the base of Intelligent Transportation Systems (ITS), namely, IEEE 802.11p-based and cellular-based solutions. For the IEEE 802.11p-based solutions, the representative standards are IEEE DSRC and ETSI Cooperative-ITS (C-ITS); for cellular-based solutions, the representative standard is 3GPP C-V2X. With the development of cellular technologies, C-V2X solutions that are compatible with current widely used communication technology (4G LTE and 5G) show promising application prospects for VANETs. In the C-V2X Release-14 standard, two modes are supported, namely Modes 3 and 4. Mode 3 encompasses Vehicle-to-Infrastructure (V2I) communication and uses the radio access network (i.e., needs the supports of infrastructures), while Mode 4 encompasses Vehicle-to-Vehicle (V2V) communication and enables user equipments (UEs) to communicate to one another directly (without the assistances of infrastructures) [1] [2]. Besides the two main communication technology, visible light communication (VLC), which uses visible light spectrum to provide energy-efficient wireless transmission and is proved to have extremely low latency and packet error rate through theoretical and experimental analysis, has received extensive attention in ITS research area recently [3] [4] [5].

Communication in VANETs can be mainly divided into two categories, i.e., safety-related communications and non-safety-related communications [6] [7]. In safety-related communications, vehicles share real-time safety-related data, such as current speed, acceleration, and steering angle, with nearby vehicles or infrastructures. Using non-safety-related communications, vehicles can access services from roadside units (RSUs), including payment, location-based, and infotainment services. Both safety-related and non-safety-related communications demand security and privacy. Safety-related communications contain life-critical information which should be protected from being modified or forged, while non-safety-related communications often involve private and confidential data, such as location-based service requests and traffic information releases. Thus, security and privacy are vital for VANETs [8] [9].

To secure the message transmission channel, designing an authenticated key agreement (AKA) protocol is essential. As pointed by Canetti and Krawczyk [10], a key exchange protocol that satisfies session key security, symmetric encryption and authentication algorithms can be combined to obtain secure communication channels. Symmetric encryption and authentication algorithms have been well-studied, so the critical task is to design an efficient AKA protocol that satisfies session key security to generate a secure session key.

Several AKA schemes have been proposed to establish session keys between vehicles or between vehicles and infrastructure. Mejri et al.'s scheme [11] is not lightweight enough because the key is generated from a multiplicative group of integers. Dua et al.'s scheme [12] is constructed using elliptic curve cryptography and hash functions, which involve relative-

L. Wei, J. Cui, H. Zhong, and Y. Xu are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, the School of Computer Science and Technology, Anhui University, Hefei 230601, China, the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China, and the Institute of Physical Science and Information Technology, Anhui University, Hefei 230601, China (e-mail: cuijie@mail.ustc.edu.cn).

L. Liu is with the School of Informatics, University of Leicester, LE1 7RH, UK (email: l.liu@leicester.ac.uk).

ly low computational and communication overhead, but their scheme requires cluster-head vehicles that are assumed to be trusted, and a key updating operation is not considered. Islam et al.'s scheme [13] is constructed on a finite field without using any elliptic curve or bilinear pairing algorithm and considers the vehicle joining and leaving situations. However, their scheme has some weaknesses. First, the key updating process needs trusted authority (TA) to remain online and send an encrypted group key back to every vehicle using unicast, so the method is inefficient and impractical. Second, the updated key is encrypted only by performing an XOR on the updated key and the old key. The encrypted new key is transmitted on public channels; hence, vehicles that have been leaving the region can still compute the newly updated key. Third, their proposed scheme is not proven secure. Ma et al.'s scheme [14] is based on elliptic curve cryptography (ECC) and was proven secure, but it fails to support multiple-vehicle scenario.

The existing VANETs AKA schemes exist several deficiencies. Firstly, these schemes do not well support the establishment of a common session key for multiple vehicles and one RSU, while the scenario is very common in VANETs. Secondly, the key agreement or updating methods of these AKA schemes require the frequent participant of TA or the impractical trustiness on vehicles. Thirdly, the key establishment overhead of these schemes are not as low as possible. Fourthly, the authentication protocols of these schemes do not support multi-TA setting.

To solve the aforementioned weaknesses of existing schemes, we design an efficient AKA protocol that supports key updating situations for V2V and V2I communications. Specifically, our proposed scheme can be divided into two phases, i.e., an authentication phase and a key agreement phase. For the authentication phase, we design an efficient and lightweight authentication algorithm based on secure one-way hash functions and ECC. Through the algorithm, vehicles, RSU, and TA can authenticate each other, so that the identities and public keys of vehicles and RSU can be authenticated. Additionally, the authentication algorithm supports multi-TA setting, so that the proposed scheme will be suitable to be deployed in real environment (because single-point failure problem can be avoided). For the key agreement phase, we design a tree-based key agreement algorithm that considers two types of events, i.e., a new authenticated vehicle joins the communication group, and an old vehicle leaves the communication group, which will trigger RSU to broadcast public key updating message that can help legal vehicles update the session key correctly. The key updating message is secured by elliptic curve signature, and does not contain any identity information, so that the identity privacies of vehicles can always be guaranteed. Additionally, the key agreement algorithm has very low transmission and computational overhead, and does not require any unpractical security assumption. Through the key agreement phase, a common session key, which is the output of the key agreement phase, acts as the input to a secure key derivation function (KDF) for generating a series of temporal session keys to secure V2V and V2I communications together with the off-the-shelf encryption or authentication algorithms.

The proposed scheme has the following potential applications.

1) **Application1: Secure V2I Communication**. If one RSU separately establishes a session key with one vehicle, then the proposed protocol can be used to secure V2I communication. Specifically, a vehicle sends the authentication request and achieves mutual authentication with RSU and TA through the authentication algorithm. An authenticated vehicle can use the material broadcasted by RSU to calculate the common session key through the key agreement algorithm, which can be used to secure V2I communication.
2) **Application2: Secure Vehicle Platooning**. If one RSU establishes a session key with multiple vehicles, then the proposed protocol can be used to secure vehicle platooning communication, which can increase road capacity and decrease fuel consumption [15]. Specifically, a batch of vehicles that are in a platoon send the authentication requests through the authentication phase and achieve mutual authentication with RSU and TA through the authentication algorithm. The authenticated vehicles can use the material broadcasted by RSU to calculate the session key through the key agreement algorithm, which can be used to secure V2V communication among the authenticated platooning vehicles.
3) **Application3: Secure Network Slicing**. If one RSU establishes a session key with multiple vehicles, then the proposed protocol can be used to secure network slicing. Network slicing, which is an important technology in 5G, enables vehicles that are in the same service domain to be in the same virtual slice so that the management will be more convenient and the security attacks will be easier to be found. Through the key agreement algorithm, vehicles that pass the authentication phase can dynamically join or leave a slice, and the negotiated key can be used to secure the internal V2V and V2I communications in a slice.

The contributions of the proposed protocol are summarized as follows.

1) We propose an AKA scheme for VANETs, including a three-party authentication protocol and a tree-based key agreement algorithm, which can be used to secure V2V and V2I communication simultaneously. The authentication protocol enables vehicles, RSU, and TA to achieve mutual authentication with extremely low costs, and supports multi-TA setting. The key agreement algorithm enables secure computation of the common session key by an optimized broadcast message, and supports the vehicle joining and leaving operations. Compared with Kim et al.'s scheme [16] that inspired us, the proposed key agreement algorithm has several advantages or novelties. Firstly, there is no requirement for trust of nodes (vehicles). Secondly, a full authentication mechanism guarantees that public keys used for the session key calculation process cannot be modified. Thirdly, only a part of public keys instead of all public keys of the binary tree are broadcasted as the key updating information,

so that the channel bandwidth can be saved efficiently. Finally, the ECC group is used instead of multiplicative group so that the public keys have lower sizes.

2) We use lightweight cryptography technology that is composed of elliptic curve cryptography and hash functions to design our scheme, so that the computational and communication overhead brought by AKA operations are relatively low. The simulation results showed that our proposed scheme has performance advantages over related schemes.
3) We conduct a formal security proof to show that our proposed scheme satisfies session key security. Hence, our proposed AKA scheme, together with off-the-shelf encryption and authentication algorithms, can be combined to construct secure channels of V2V and V2I communications in VANETs. Besides, our proposed scheme can achieve essential security and privacy requirements in VANETs.

II. Related Work

In order to ensure the safety of vehicular communications, many valuable schemes have been proposed which can be classified into four main categories: public key infrastructure (PKI)-based schemes, identity-based schemes, group signature-based schemes, and AKA-based schemes.

In PKI-based schemes, the digital certificate is used to guarantee the validity of the public key, which is contained in the message sent by a vehicle. Only the certified public key can be used to verify the related messages. Raya et al. [17] considered the security issues in VANETs systematically, devised an appropriate security architecture, and proposed a PKI based protocol to secure VANETs. They first proposed the concept of conditional privacy-preserving, i.e., the identity of a vehicle should not be able to be traced by any entity except for TA. Plossl et al. [18] proposed a privacy-aware and efficient security infrastructure for VANETs. In their scheme, asymmetric cryptography and symmetric cryptography are used to secure accident messages and Internet service contents separately. However, the digital certificate is necessary for every message. Lu et al. [19] proposed a CPPA scheme based on the anonymous certificate. Instead of requiring every vehicle to store a large amount of revoked certificate, the vehicle in their scheme can interact with nearby RSU to get a temporal key certificate that can be verified by other vehicles. However, frequent interactions and computationally expensive operations bring high computational overhead.

Compared with traditional public key cryptography in which the public key is composed of random bits, identity-based cryptography enables public key can be any bit-string specified by the user, so that more convenience and flexibility can be brought. Zhang et al. [20] proposed an efficient batch signature verification algorithm that are based on identity-based signature. In their proposed scheme, the received message signatures can be verified quickly, so that the total computational overhead of the whole system can be reduced dramatically. However, Lee et al. [21] pointed out that Zhang et al. 's scheme can not resist replay attack and man-in-the-middle attack. Chim et al. [22] proposed an RSU-based authentication scheme in which all message authentication tasks are centralized on the RSU side and the authentication results can be broadcasted to nearby vehicles by bloom filter. Therefore, the computational overhead on vehicle side in their scheme are very limited. Since bilinear pairing operations bring high computational overhead, He et al. [23] proposed an efficient conditional privacy-preserving scheme that uses elliptic curve cryptography (ECC) instead of pairing-based cryptography. After He et al. 's scheme [23], many ECC-based conditional privacy preserving schemes has been proposed, such as [24], [25], [26].

Group signature scheme was firstly introduced by Chaum et al. [27], which can provide anonymity for message signers. In group signature-based schemes, every member of the group can sign messages anonymously but there is a trust authority can trace the signature using a trapdoor function [28]. Sun et al. [29] first adopt short group signatures and identity-based signature to secure vehicular communication, so that CPPA can be achieved. However, Sun et al. 's scheme is based on centralized key management, which limits the performance. To solve this issue, Hao et al. [30] proposed a distributed key management scheme that can identify compromised RSUs and colluded malicious vehicles. Wasef et al. [31] proposed an efficient group signature scheme that can support batch signature verification operation, so that the system performance can be improved efficiently. Shao et al. [32] proposed a decentralized group model based on a modified group signature scheme for V2X communication, which supports threshold and anonymous authentication.

Another method to secure vehicular communication is by designing an AKA protocol. As pointed by Canetti and Krawczyk [10], key exchange or agreement protocol that satisfies session key security, symmetric encryption, and authentication algorithms can be combined to obtain secure communication channels, which can guarantee the communication security. Huang et al. [33] proposed a batch AKA protocols, which can establish different session keys for different vehicles simultaneously. In their proposed scheme, an ECC-based signature algorithm is used to avoid the high computational overhead of bilinear pairing. Taha et al. [34] proposed a link-layer AKA scheme based on certificateless public key cryptography, which can guarantee the security of public hotspots in VANETs. Wang et al. [35] proposed a self-certified public key protocol to support value-added vehicular services. Their scheme supports batch authentication, but it is based on the computationally expensive bilinear pair operation so that the AKA computational overhead may fail to satisfy the performance requirement. Mejri et al. [11] first considered the group key generation problem in VANETs, which was constructed on a variant of traditional Diffie-Hellman key exchange algorithm. However, the key in their protocol is selected from the multiplicative group of integers modulo a large prime, so that the transmission overhead will be very large.

Dua et al. [12] proposed a two-level AKA protocol where elected cluster-head (CH) vehicles are responsible for establishing session keys for nearby vehicles respectively. Liu et al. [36] proposed a dual authentication scheme for securing V2V communications by using trusted computing and pairing based cryptography. However, pairing based cryptography brings too much computational overhead. Jiang et al. [37] proposed an integrated AKA frame for vehicular cloud computing (VCC). In their scheme, a single-server three-factor authentication protocol and non-interactive key establishment protocol were proposed to secure VCC. Islam et al. [13] presented a password-based CPPA and group key generation protocol for VANETs. However, their schemes exist some weakness. Firstly, the key updating process needs TA to keep online and send the encrypted group key back to every vehicle over unicast, which is very inefficient. Secondly, the updated key is encrypted by simply performing bitwise XOR operation on the updated key and old key, and the encrypted new key is transmitted on the public channel, which means that vehicles that have left the region can still compute the new updated key. Third, their proposed scheme is not proven secure. Dang et al. [38] presented an efficient pairing-free one-round AKA protocol, which can be proven secure in the extended CK model and resist all essential attacks. However, their scheme was constructed on bilinear pairing, so that the computational overhead of their scheme will be high. Liu et al. [39] proposed a secure and efficient AKA protocol that uses RSU to negotiate a shared secret key among vehicles and RSU. However, the design of their scheme relies on bilinear pairing and the key updating mechanism requires the broadcast of public keys of all vehicles, so that the efficiency of their scheme is not high enough. Ma et al. [14] designed an AKA protocol without using bilinear pairing for fog-based VANETs, which is proven to be session-key security and can be used to secure V2I communications. Li et al. [40] proposed a proven secure and lightweight two-party AKA protocol that can be used to secure V2V communications. Eftekhari et al. [41] proposed a three-party pairwise AKA protocol for fog-based VANETs, which supports the the password updating at the user side and provides relatively better performance advantages than traditional key agreement schemes. However, the session key established by their protocol is a pairwise key for two parties instead of three parties.

To construct secure and reliable group communication, it is necessary to construct a group key establishing mechanism. Kim et al. [16] proposed a tree-based group Diffie-Hellman key agreement protocol that has significant efficiency advantage over traditional group key agreement schemes. Their proposed scheme supports node joining and leaving, and group merging and dividing. Although their proposed protocol is proven secure and satisfies forward secrecy, backward secrecy and key independence, one requirement of their proposed protocol makes it unsuitable to be acted as the key management scheme for V2V scenario. In their proposed protocol, the member joining or leaving event will change the structure of old binary tree (BT), so a sponsor is required to compute and broadcast the updated blinded (or public) keys to other members, whereas the sponsor node is not at a constant position of BT. Since the node (vehicle) in VANETs is untrustworthy, it is not reasonable to assume a vehicle can be the sponsor node (a vehicle may forge the updated blinded keys to be broadcasted). After Kim et al. [16]'s scheme, multiple improved schemes such as [42], [43], [44]. However, these schemes still have the same requirement as that in Kim et al. [16]'s scheme and are not lightweight enough.

III. Background

A. System Model

The system model of our proposed scheme for VANETs mainly contained three roles, i.e., vehicle, RSU, and TA, as shown in Fig. 1.

- Vehicle: Vehicle is considered an untrustworthy entity and the on-board-unit (OBU) installed of which is responsible for V2V and V2I communications in VANETs. Before joining VANETs, all vehicles are required to register to TA through a secure and exclusive channel. A vehicle can communicate with nearby vehicles through V2V communication and communicate with RSU through V2I communication.
- RSU: RSU is denoted as road side unit in VANETs, which is a kind of wireless infrastructure, deployed on roadside and mainly responsible for providing network access service for vehicles in the proximity. For improving traffic safety and conveniences, RSU also provide safety message broadcasting services and entertainment services to nearby vehicles.
- TA: TA is responsible for providing registration and authentication services for vehicles and RSU. TA is trustworthy by all entities in VANETs and controlled by official traffic management. To avoid the single-point failure, we assume that redundant TAs are set up.

B. Complexity Assumptions

Let $\mathbb{G}$ denote a cyclic additive group of elliptic curve over finite field, $P \in \mathbb{G}$ denote a generator of group $\mathbb{G}$, and q is the prime order of group $\mathbb{G}$. Then we have the following computational or decisional hard problems.

1) Elliptic Curve Discrete Logarithm Problem (ECDLP): Given one tuple $\{P, Q\}$ where $Q = xP$, the advantage for any probabilistic polynomial time (PPT) adversary to calculate $x \in^R \mathbb{Z}_q^*$ is negligible.
2) Elliptic Curve Computational Diffie Hellman Problem (ECCDHP): Given one tuple $\{P, xP, yP \in \mathbb{G}\}$ where $x, y \in^R \mathbb{Z}_q^*$, the advantage for any probabilistic polynomial time (PPT) adversary to calculate $xyP \in \mathbb{G}$ is negligible.
3) Elliptic Curve Decisional Diffie Hellman Problem (ECDDHP): Given two tuples $\{P, xP, yP, xyP \in \mathbb{G}\}$ and $\{P, xP, yP, zP \in \mathbb{G}\}$ where $x, y, z \in^R \mathbb{Z}_q^*$, any probabilistic polynomial time (PPT) adversary is decisional hard to distinguish the two tuples.

IV. Our proposed scheme

In this section, we describe the main phases in our proposed protocol that consists of six phases. In the first three phases, TA setups up all necessary parameters and broadcasts public parameters, and vehicles and RSU register to TA to hold long term session keys with TA respectively. In the fourth

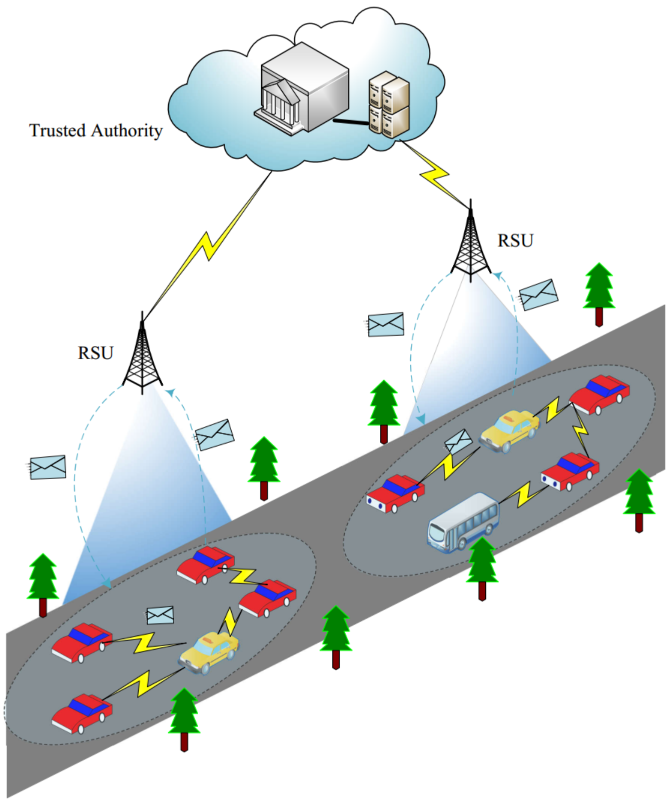


Fig. 1. System Model

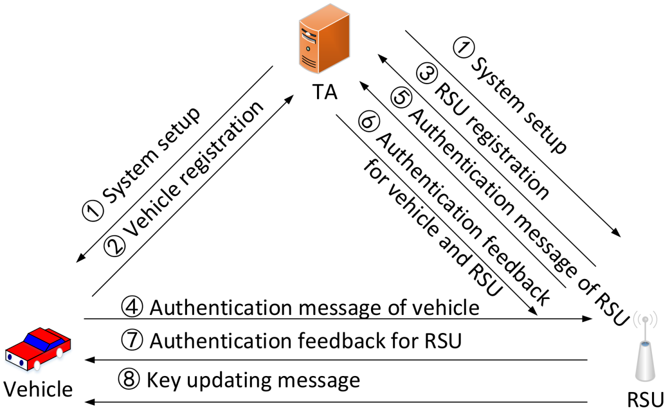

TA
① System setup
② Vehicle registration
① System setup
③ RSU registration
⑤ Authentication message of RSU
⑥ Authentication feedback for vehicle and RSU
④ Authentication message of vehicle
⑦ Authentication feedback for RSU
⑧ Key updating message
Vehicle
RSU

Fig. 2. The Main Process of The Proposed Scheme

phase, a three-party authentication phase is entered to make the vehicle, RSU and TA get authenticated by each other, which can ensure that vehicle's identity and the public key for key agreement get authenticated. In the last two phases, the tree-based key agreement algorithm, which considers the the events of joining or leaving of authenticated vehicles, makes authenticated vehicles and RSU establish a common session key. The main process are shown in Fig. 2, and the detailed steps of each phase are as follows.

A. Setup Phase

In this phase, TA initializes all necessary system parameters. Specifically, given a security parameter λ that is equal to the group order q in group-based cryptography as the input, TA executes the following steps.

TABLE I. Notations Table

Acronym	Full name
VANETs	Vehicular ad hoc networks
RSU	Roadside unit
TA	Trusted authority
AKA	Authenticated key agreement
V2V	Vehicle-to-vehicle
V2I	Vehicle-to-infrastructure
C-V2X	Cellular-based vehicle-to-everything
3GPP	The 3rd generation partnership project
ECC	Elliptic curve cryptography
D2D	Device-to-device
ECDLP	Elliptic curve discrete logarithm problem
ECCDHP	Elliptic curve computational diffie hellman problem
ECDDHP	Elliptic curve decisional diffie hellman problem
TPM	Trusted platform module
s, P_{pub}	System secret key and public key, respectively
RID_i	Real identity of vehicle
TPK_i, TSK_i	Tree public key and secret key, respectively

1) TA chooses a cyclic additive group $\mathbb{G}$ of elliptic curve over finite field where the group order is a prime q and the group generator is P.
2) TA chooses a random number $s \in \mathbb{Z}_q^*$ as the system private key and calculates the corresponding public key $P_{pub} = sP$ as the system public key.
3) TA chooses several cryptographic hash functions $h_i(1 \leq i \leq 5)$ where $h_1 : \{0,1\}^* \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_2 : \mathbb{G} \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_3 : \{0,1\}^* \times \{0,1\}^* \times \mathbb{G} \times \mathbb{G} \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_4 : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_5 : \{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \times \mathbb{G} \times \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_6 : \mathbb{G} \rightarrow \mathbb{Z}_q^*$.
4) TA publishes the public system parameters $\{\mathbb{G}, q, P, P_{pub}, h_i(1 \leq i \leq 6)\}$.

B. Vehicle Registration Phase

In this phase, every vehicle V_i is required to register with TA to make its unique identity information get recorded by TA, where the registration process should be performed in a secure private channel. Specifically, the phase contain several steps.

1) V_i chooses its real identity $RID_i \in \mathbb{Z}_q^*$, and sends RID_i to TA.
2) Upon receiving RID_i from V_i, TA checks whether RID_i is existed. If it exists, TA sends a fail signal to V_i and V_i needs to re-choose RID_i. Otherwise, TA calculates $l_i = h_1(s, RID_i)$, sends l_i to V_i, and finishes the registration phase with V_i.

C. RSU Registration Phase

In this phase, every RSU_j is required to register with TA to get its identity information recorded and establish a long-term session key, where the process is performed in a secure private channel. Specifically, the phase contains several steps.

1) RSU_j chooses its identity $ID_j \in \mathbb{Z}_q^*$ and one random number $d \in \mathbb{Z}_q^*$ that is used to calculate the elliptic curve digital signature (ECDSA [45]) in the later phase, calculates $Q_j = dP$, and sends $\{ID_j, Q_j\}$ to TA.
2) Upon receiving $\{ID_j, Q_j\}$, TA checks ID_j to make sure its uniqueness. If ID_j is unique, TA calculates the long-term session key $l_j = h_1(s, ID_j)$ between RSU_j and itself, and sends l_j to RSU_j. Otherwise, RSU_j needs to re-choose ID_j.

Remark 1.1. In the above two registration phases, l_i and l_j act as the long-term session keys that are used to secure the following authentication phase. Note that l_i and l_j are generated by a cryptographic secure hash function with two inputs, i.e., identity and TA's secret key. The one-way characteristic of the hash function guarantees that TA's secret key will not be leaked. The generation method can make that TA can recover l_i or l_j quickly upon revealing the identity RID_i (recovered from PID_i as shown in the next subsection) or receiving ID_j. An alternative long-term key generation method is by setting l_i and l_j to true random numbers. However, the method requires TA to maintain a table for the correlation of the identity and corresponding long-term session key.

Remark 1.2. In the above settings, TA is a single entity that is responsible for providing registration and authenticating services simultaneously. As shown in the next subsection, the core authentication operations of TA include revealing the real identity, computing the long-term session key, and checking the validity of the real identity. The first two operations heavily rely on the system secret key s, and the last operation requires to know the list of the real identities of legal vehicles or unrevoked vehicles. To avoid the single-point failure and improve the authenticating performance, one single TA can be replaced by a root-TA with multiple sub-TAs, in which root-TA provides registration and management services and sub-TAs provide authentication service. Specifically, the operations involving the system secret key s can be executed in the trusted platform module (TPM) and every sub-TA equips the TPM to protect the security of s. As for the list of the real identities, root-TA can distribute the Cuckoo filter [46] or Bloom filter [47] (provides $O(1)$-level lookup operation) that stores the fingerprints of the legal vehicles' real identities, so that the legality of a vehicle can be quickly determined and the list of the real identities can be securely protected (sub-TA only needs to lookup the received filter and believes that an identity is legal if it can be found). Note that the following phases can also be suitable for the multi-TA environment and the only difference is that sub-TAs instead of TA execute the operations.

D. Three-Party Authentication Phase

Since vehicles are not trustworthy, they are required to get authenticated by TA before entering key agreement phase. Identically, the credibility of RSU should also be affirmed by TA. So three parties containing V_i, RSU_j and TA are required to perform mutual authentication. The vehicle that wants to negotiate a common session key with other vehicles is required to get authenticated by RSU and TA in this phase. The interactive figure of the phase is as shown in Fig. 3. It is noteworthy that all the following steps are all performed in public insecure channel.

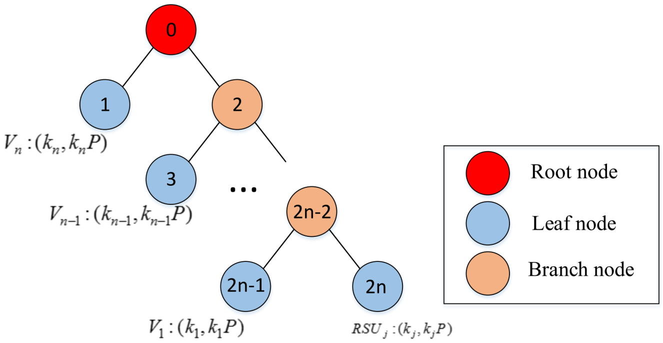


Fig. 4. The Structure of Tree BT_n

1) V_i chooses a random number $r_i \in \mathbb{Z}_q^*$, gets the current timestamp T_i, calculates $R_i = r_i P$ and its pseudo identity $PID_i = RID_i \oplus h_2(r_i P_{pub}, T_i)$. To participate in the group key generation process, V_i chooses a random number $k_i \in \mathbb{Z}_q^*$ as the partial secret key that would be used to generate group key, and calculates the corresponding partial public key $K_i = k_i P$. In the end, V_i calculates the hash value $\alpha_i = h_3(RID_i, T_i, R_i, K_i, l_i)$, and sends $M_1 = \{PID_i, T_i, R_i, K_i, \alpha_i\}$ to RSU_j.
2) After receiving M_1 from V_i, RSU_j checks the validity of T_i by determining whether the equation $|T_i - T_{cur}| \leq \Delta T$ holds where T_{cur} denotes the current timestamp and ΔT denotes the max legal time interval (note that the following steps all use the timestamp checking method).If not, RSU_j rejects the message and aborts the request. Otherwise, RSU_j stores M_1, gets the current timestamp T_j, calculates the hash value $\gamma_j = h_4(ID_j, T_j, \alpha_i, l_j)$, and sends $M_2 = \{ID_j, T_j, \gamma_j, PID_i, T_i, \alpha_i, K_i, R_i\}$ to TA.
3) After receiving M_2 from RSU_j, TA first checks the validity of T_j. If valid, TA recovers l_j by calculating $l_j = h_1(s, ID_j)$, calculates $\gamma_j' = h_4(ID_j, T_j, \alpha_i, l_j)$, and checks if $\gamma_j' = \gamma_j$ holds. If not, TA aborts the request. Otherwise, TA checks the validity of T_i. If valid, TA calculates $RID_i' = PID_i \oplus h_2(sR_i, T_i)$, and determines whether RID_i' is a legal identity. If not, TA aborts the request. Otherwise, TA recovers l_i by calculating $l_i = h_1(s, RID_i')$, calculates $\alpha_i' = h_3(RID_i', T_i, R_i, K_i, l_i)$, and checks whether the equation $\alpha_i' = \alpha_i$ holds. If not, TA aborts the request. Otherwise, TA authenticates the legality of V_i and RSU_j, and calculates $\delta_{TA} = h_4(T_{TA}, ID_j, Q_j, l_i)$ and $\eta_{TA} = h_5(T_{TA}, PID_i, \alpha_i, K_i, l_j)$, where T_{TA} denotes the current timestamp, Q_j denotes the public keys generated by RSU_j in the registration phase. At the end, TA sends $M_3 = \{T_{TA}, \delta, \eta\}$ to RSU_j.
4) After receiving M_3 from TA, RSU_j checks the validity of T_{TA}. If valid, RSU_j computes $\eta_{TA}' = h_5(T_{TA}, PID_i, \alpha_i, l_i)$, and checks if the equation $\eta_{TA}' = \eta$ holds. If not, RSU_j rejects the request. Otherwise, RSU_j authenticates the identity PID_i and partial public

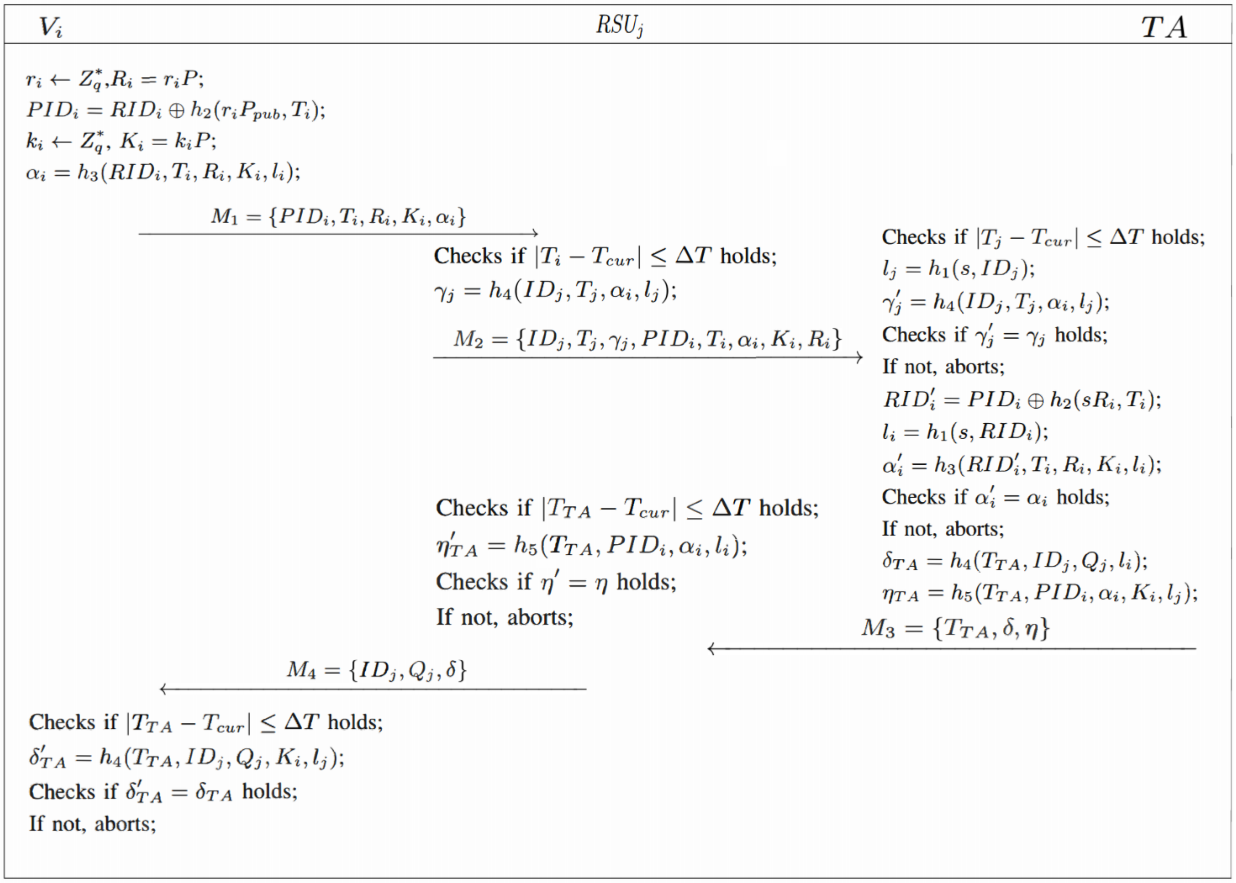

V_i
RSU_j
TA
$r_i \leftarrow Z_q^*, R_i = r_i P$;
$PID_i = RID_i \oplus h_2(r_i P_{pub}, T_i)$;
$k_i \leftarrow Z_q^*$, $K_i = k_i P$;
$\alpha_i = h_3(RID_i, T_i, R_i, K_i, l_i)$;
$M_1 = \{PID_i, T_i, R_i, K_i, \alpha_i\}$
Checks if $|T_i - T_{cur}| \leq \Delta T$ holds;
$\gamma_j = h_4(ID_j, T_j, \alpha_i, l_j)$;
$M_2 = \{ID_j, T_j, \gamma_j, PID_i, T_i, \alpha_i, K_i, R_i\}$
Checks if $|T_j - T_{cur}| \leq \Delta T$ holds;
$l_j = h_1(s, ID_j)$;
$\gamma_j' = h_4(ID_j, T_j, \alpha_i, l_j)$;
Checks if $\gamma_j' = \gamma_j$ holds;
If not, aborts;
$RID_i' = PID_i \oplus h_2(sR_i, T_i)$;
$l_i = h_1(s, RID_i)$;
$\alpha_i' = h_3(RID_i', T_i, R_i, K_i, l_i)$;
Checks if $\alpha_i' = \alpha_i$ holds;
If not, aborts;
$\delta_{TA} = h_4(T_{TA}, ID_j, Q_j, l_i)$;
$\eta_{TA} = h_5(T_{TA}, PID_i, \alpha_i, K_i, l_j)$;
$M_3 = \{T_{TA}, \delta, \eta\}$
Checks if $|T_{TA} - T_{cur}| \leq \Delta T$ holds;
$\eta_{TA}' = h_5(T_{TA}, PID_i, \alpha_i, l_i)$;
Checks if $\eta' = \eta$ holds;
If not, aborts;
$M_4 = \{ID_j, Q_j, \delta\}$
Checks if $|T_{TA} - T_{cur}| \leq \Delta T$ holds;
$\delta_{TA}' = h_4(T_{TA}, ID_j, Q_j, K_i, l_j)$;
Checks if $\delta_{TA}' = \delta_{TA}$ holds;
If not, aborts;

Fig. 3. The Three-Party Authentication Phase

key K_i of V_i. At last, RSU_j sends $M_4 = \{ID_j, Q_j, \delta\}$ to V_i.

5) After receiving M_4 from RSU_j, V_i checks the validity of T_{TA}. If valid, V_i computes $\delta_{TA}' = h_4(T_{TA}, ID_j, Q_j, l_i)$ and checks whether the equation $\delta_{TA}' = \delta_{TA}$ holds. If not, V_i aborts the process. Otherwise, V_i authenticates the legality of TA and RSU_j.

Remark 2. In step 1), V_i sends its pseudo identity $PID_i = RID_i \oplus h_2(r_i P_{pub}, T_i)$ instead of RID_i, because the use of RID_i will leak the identity privacy of V_i and make V_i easy to be tracked. To achieve conditional privacy-preserving, i.e., only TA can extract RID_i from the message, PID_i is a XOR-encryption (similar to stream cipher) of RID_i based on ECDH. The method is similar to the method that IMSI is encrypted using ECIES in 5G system [48]. Without knowing s, the adversary has to solve ECCDHP that is computationally hard to get RID_i. The timestamp T_i makes the secret key $h_2(r_i P_{pub}, T_i)$ keeps changing, so that RID_i will be more secure. In steps 2)-4), a timestamp checking mechanism is existed for resisting replay attack as much as possible. All steps in this phase try to use secret long-term session keys and secure hash functions to construct keyed-hash authentication functions (similar to HMAC [49]), so that the vehicle, RSU, and TA can authenticate each other. From the view of RSU_j, RSU_j can get two conclusions if step 4) passes, i.e., the identity of V_i is legal, and K_i that will participate the following key agreement phase is sent from V_i and not modified by any adversary during the transmission process. From the view of V_i, ID_j and Q_j that is used to verify the ECDSA signature in the following phases are verified if step 5) passes.

E. Key Agreement Phase for New Vehicle Joining

After a vehicle get authenticated by RSU, its identity and K_i get trusted, then it would have the ability to establish common session key with RSU and other authenticated vehicles. In this paper, we design a new tree-based key agreement algorithm to compute the common session key, which only requires the behaviour of the RSU is honest. It is noteworthy that the original tree-based key agreement algorithm [16] that inspired us is not suitable for VANETs because of the reasons described in Section I&II.

To describe the key agreement algorithm for new vehicle joining, we denote the vehicles that have joined key agreement process as ordered sequence $\{V_1, V_2, ..., V_{n-1}\}$ where V_1 is the first joined vehicle and V_{n-1} is the last joined vehicle, and the new vehicle is V_n. It is noteworthy that the indexes $i = 1, 2, ..., n$ of V_i are just used to mark the order of vehicle joining and will be changed when one of vehicles leaves. The method used to calculate the common session key is as shown follows.

1) A binary tree BT_n is constructed to compute the common group key and the structure of BT_n satisfies two features.

One is that the depth of BT_n is equal to n, which is the number of current vehicles. The other is that BT_n is generated by inserting BT_{n-1} to the right child of a perfect binary tree with three nodes.

2) Every node N_i of BT_n is labeled with a number $i (0 \leq i \leq 2n)$, so a node labeled with a odd number is a leaf node and a node labeled with a even number except for $2n$ is a branch node. And every N_i is associate with a tree secret key TSK_i and a tree public key TPK_i that is computed by $TPK_i = TSK_i \cdot P$. The secret key of a branch node N_i is computed by equation (1). All the secret key of leaf nodes N_i except for the rightmost and bottommost leaf node N_{2n} is set to the partial secret key $k_{n-(i-1)/2}$ of vehicle $V_{n-(i-1)/2}$. The secret key of the rightmost and bottommost leaf node N_{2n} is set to the partial secret key k_j of RSU_j. The structure of BT_n is as shown in Fig. 4.

$$\begin{aligned} TSK_i =& h_6(TSK_{i+1} \cdot TPK_{i+2}) \\ =& h_6(TSK_{i+2} \cdot TPK_{i+1}) \end{aligned} \tag{1}$$

3) The secret key TSK_0 of the root node N_0 is the common group key for RSU_j and vehicles $V_1, V_2, ..., V_n$.

Before establishing session with authenticated vehicles, RSU_j chooses a random number k_j as the partial secret key and calculates corresponding partial public key $K_j = k_j P$ just as the ways vehicles do. If the n-th vehicle V_n that has passed the authentication in the last phase sends a joining request to RSU_j, RSU_j updates the binary tree by adding the new leaf node, computes the secret key TSK_{2i} and public key TPK_{2i} of every branch node by using equation (1) recursively, where $0 \leq i \leq n-1$. Next, RSU_j chooses a random number $b_j \in \mathbb{Z}_q^*$, calculates $B_j = b_j P$, gets x-axis value B_j^x of point B_j, calculates the signature $s_j = b_j^{-1} \cdot (h(n, T_j, TPK_1, TPK_2) + B_j^x \cdot d)$ to get the ECDSA [45] signature $\sigma_j = (B_j^x, s_j)$. At the end, RSU_j broadcasts the tuple $\{TPK_1, TPK_2, n, T_j, \sigma_j\}$ to nearby vehicles $V_i (1 \leq i \leq n)$ where the index n is used for assisting vehicles to maintain the structure of BT_n.

After receiving $\{TPK_1, TPK_2, n, T_j, \sigma_j\}$ from RSU_j, V_i needs to check the validity. Specifically, V_i checks the freshness of T_j and rejects the message if T_j is expired. Next, V_i calculates $X_j = h(TPK_1, TPK_2, T_j) \cdot s_j^{-1} \cdot P + B_j^x \cdot s_j^{-1} \cdot Q_j$ and compares the x-axis value of X_j with B_j^x. If the two values are equal, $V_i (1 \leq i \leq n)$ receives the message from RSU_j, and uses $\{TPK_1, TPK_2, T_j\}$ to calculate the group key TSK_0 by equation (1). Otherwise, V_i rejects the message.

To describe the above process more vividly, we give a example where three authenticated vehicles $\{V_1, V_2, V_3\}$ and RSU_j establish a common session key. The process is as shown in Fig. 5.

1) At first, V_1 sends the joining request to RSU_j. RSU_j computes $TSK_0 = h_6(k_j \cdot TPK_1) = h_6(k_j \cdot K_1) = h_6(k_j k_1 P)$ and $TPK_0 = TSK_0 \cdot P$ and broadcasts message $\{TPK_1, TPK_2, 1, T_j, \theta_j\}$ to V_1. It is noteworthy that the partial public keys of every authenticated vehicles have been sent to RSU_j in authentication phase. After receiving the message, V_1 computes $TSK_0 = h_6(k_1 \cdot TPK_2) = h_6(k_1 k_j P)$. For simplicity, we ignore the signature generation and verification process.

2) Next, V_2 sends the joining request to RSU_j. RSU_j computes $TSK_0 = h_6(TSK_2 \cdot TPK_1) = h_6(TSK_2 \cdot K_2) = h_6(h_6(k_1 k_j P) \cdot k_2 P)$ where TSK_2 is equal to TSK_0 in step 1) and $TPK_2 = TSK_2 \cdot P = h_6(k_1 k_j P) \cdot P$, and broadcasts $\{TPK_1, TPK_2, 2, T_j, \theta_j\}$ to V_1 and V_2. For V_1, V_1 computes $TSK_0 = TSK_2 \cdot TPK_1 = h_6(h_6(TSK_2 \cdot TPK_1)) = h_6(h_6(k_1 k_j P) \cdot k_2 P)$ where TSK_2 is equal to TSK_0 in step 1). For V_2, V_2 computes $TSK_0 = h_6(TSK_1 \cdot TPK_2) = h_6(k_2 \cdot h_6(k_1 k_j P) \cdot P)$.

3) At last, V_3 sends the joining request to RSU_j. RSU_j computes $TSK_0 = h_6(TSK_2 \cdot TPK_1) = h_6(TSK_2 \cdot K_3)$ where TSK_2 is equal to TSK_0 in step 2) and $TPK_2 = TSK_2 \cdot P$, and broadcasts $\{TPK_1, TPK_2, 3, T_j, \theta_j\}$ to V_1, V_2 and V_3. For V_1 and V_2, they can calculate $TSK_0 = h_6(TSK_2 \cdot TPK_1) = h_6(TSK_2 \cdot K_3)$ where TSK_2 is equal to TSK_0 in step 2). For V_3, V_3 computes $TSK_0 = TSK_1 \cdot TPK_2 = h_6(k_3 \cdot TPK_2)$ where TPK_2 is contained in the broadcast message of RSU_j.

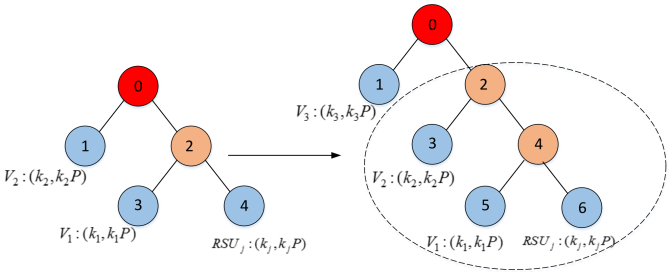


Fig. 5. The Key Agreement for New Vehicle Joining

F. Key Agreement Phase for Old Vehicle Leaving

If V_x leaves the communication range of RSU_j or quits the current group by sending quitting request, RSU will determine that V_x is a leaving vehicle. Assuming that n vehicles have joined the group process, if a vehicle $V_x (1 \leq x \leq n)$ becomes a leaving vehicle, the old tree BT_n changes to BT_{n-1} as follows. The leaf node $N_{2n-2x+1}$ and branch node N_{2n-2x} of BT_n will be deleted, the bottom subtree which consists of $N_{2n-2x+2}, N_{2n-2x+3}, ..., N_{2n}$ will be moved up one level.

After receiving group leaving request from V_x, RSU_j updates the tree structure (from BT_n to BT_{n-1}), and calculates private and public keys of all updated nodes for BT_{n-1}, i.e., $N_0, N_2, ..., N_{2n-2x-2}$ of BT_{n-1}. Next, RSU_j needs to sign $\{x, n-1, T_j, 2||TPK_2, ..., (2n-2x)||TPK_{2n-2x}\}$ to get the ECDSA [45] signature σ_j and broadcasts $\{x, n-1, T_j, \sigma_j, 2||TPK_2, ..., (2n-2x)||TPK_{2n-2x}\}$. It is noteworthy that the index together with TPK_i is used to avoid the adversary to change the relative order of TPK_i. After receiving the broadcast message, any joined V_i is able to update the tree structure of BT_n according to the number x and n contained in the latest updating message of the joining phase (we will show its correctness in the next section), and uses required tree public keys to compute TSK_0. At the end, every vehicle should update the indexes of BT_{n-1} by subtracting i by one for every known $x+1 \leq i \leq n$. Since the signing and verifying method of messages sent from RSU_j is exactly as that of vehicle joining situation except for using a different hash function, we do not elaborate the signature sighing and verifying steps here.

To describe the above process more vividly, we give a example in which V_2 of four joined vehicles $\{V_1, V_2, V_3, V_4\}$ becomes a leaving vehicle, and show how $\{V_1, V_3, V_4\}$ and RSU_j compute the new group key. After concluding V_x as a leaving vehicle, RSU_j updates the structure of old tree BT_4 to that of the new tree BT_3, sets TSK_6 of BT_4 to TSK_4 of BT_3, calculates $TSK_2 = h_6(TSK_4 \cdot K_3)$, $TPK_2 = TSK_2 \cdot P$, $TSK_0 = h_6(TSK_2 \cdot K_1)$, $TPK_0 = TSK_0 \cdot P$, and broadcasts $\{x = 2, TPK_2, TPK_4\}$ to $\{V_1, V_3, V_4\}$. For V_1, it updates the structure of tree, and calculates $TSK_4 = h_6(k_1 \cdot K_j) = h_6(k_1 k_j P)$, $TSK_2 = h_6(TSK_4 \cdot K_3)$, and $TSK_0 = h_6(TSK_2 \cdot K_4)$. It is noteworthy that V_1 has known K_j, K_3, K_4 in the new vehicle joining process. For V_3, it updates the structure of tree, and calculates $TSK_2 = h_6(k_3 \cdot TPK_4)$, $TSK_0 = h_6(TSK_2 \cdot K_4)$. For V_4, it updates the structure of tree, and calculates $TSK_0 = h_6(k_4 \cdot TPK_2)$. The process is as shown in Fig. 6.

Remark 3.1. In the key agreement mechanism, the binary tree (BT) structure is used to manage $K_i (1 \leq i \leq n)$ orderly, so that the common session key is calculated using a deterministic method. The core of the BT structure is that K_j generated by RSU keeps locating at the rightmost and bottommost location. The special location make RSU be able to compute the public and private keys of all branch nodes and the root node of BT, so that RSU can broadcast necessary public keys of some nodes to joined (or un-leaved) vehicles for updating common session key when a new vehicle joins the group or a joined vehicle leaves the group. As shown in the next section, the broadcast content sent by RSU is enough for vehicles to calculate the new common session key.

Remark 3.2. The proposed protocol can be extended to support cross-RSU communication. Suppose that V_i exits the session $S1$ organized by RSU_1 and enters the session $S2$ organized by RSU_2. If V_i still wants to join the secure communication with original members in $S1$, RSU_1 and RSU_2 can collaborate to implement the function. Specifically, the exit of V_i will trigger RSU_1 to execute the key updating algorithm for the old vehicle leaving (see Section IV-F) so that TSK_0 of $S1$ will be updated, whereas the entrance of V_i (after the authentication process is passed) will trigger RSU_2 to execute the key updating algorithm for the new vehicle joining (see Section IV-E) so that $TSK_1^{'}$ of $S2$ will be updated. To support cross-RSU communication, RSU_1 and RSU_2 can share the public key of the root node, i.e., TPK_1 and $TPK_1^{'}$, to each other. So all members in $S1$ and $S2$ can get the common session key for cross-RSU communication by calculating $h_6(TSK_1 \cdot TPK_1^{'})$ and $h_6(TSK_1^{'} \cdot TPK_1)$, respectively.

V. Security Analysis

In this section, the content can be divided into four parts. Firstly, we show the correctness of the tree-based key agreement mechanism separatively. Secondly, we describe the security model to prove the security of the whole AKA scheme. Thirdly, we give a security proof to show that our proposed protocol is proven secure under the given security model. Finally, we demonstrate that our proposed protocol satisfies several security requirements in VANETs.

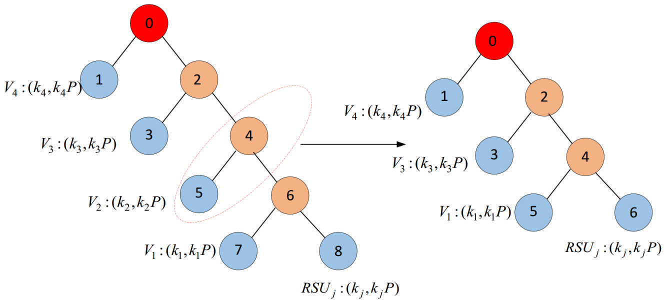


Fig. 6. The Key Agreement for Old Vehicle Leaving

A. Correctness Analysis

To show the correctness of the key agreement mechanism, we assume the current number of vehicles is $n (n \geq 1)$. According to the key computational method, the common session key is calculated by recursively executing $TSK_i = h_6(TSK_{i+1} \cdot TSK_{i+2} \cdot P)$, which is a deterministic value because the function and the inputs of the recursive process are deterministic.

First, we show that all vehicles including the old vehicles and the new joining vehicle can establish a common session key correctly. According to the key agreement process for new vehicle joining, n vehicles send the joining requests to RSU_j successively, so that RSU_j gets a deterministic tree BT_n. With authenticated public keys K_i sent from $V_i (1 \leq i \leq n)$ and its own secret key k_j, RSU_j can calculate unique public and private keys for the branch nodes and root node, and broadcasts $\{n, TPK_1, TPK_2\}$ to the n vehicles. For the new vehicle V_n, TPK_2 is the public key of the right child of root node N_0. With its own secret key k_n, the new vehicle V_n calculates the common session key $TSK_0 = h_6(k_n \cdot TPK_2)$. With TSK_2 that is the common session key for $\{V_1, ..., V_{n-1}\}$, the old vehicles $V_i (1 \leq i \leq n-1)$ calculate $TSK_0 = h_6(TSK_2 \cdot TPK_1)$. Because $k_n \cdot TPK_2 = TSK_2 \cdot TPK_1 = k_n \cdot TSK_2 \cdot P$, we can say that the key agreement mechanism for new vehicle joining make all vehicles establish a common session key.

Next, we show that all vehicles can update the common session key correctly when one of the n vehicles leaves the original group. From the key agreement process for new vehicle joining, we can find that the process has one feature, i.e., all of the old vehicles $V_i (1 \leq i \leq n-1)$ know the public key K_n of the new vehicle V_n but the new vehicle V_n cannot know the public keys K_i of the old vehicles $V_i (1 \leq i \leq n-1)$. Because of the feature, the vehicles $V_i (1 \leq i \leq x-1)$ that are located on the low locations of tree BT_n can calculate the updated common session key TSK_0 of BT_{n-1} independently, but the vehicles $V_i (x+1 \leq i \leq n)$ that are located on the high locations of tree BT_n can't. So upon determining that V_x becomes a leaving vehicle, RSU_j will update BT_n to BT_{n-1} (by deleting $N_{2n-2x}, N_{2n-2x+1}$ and updating $i = i-1$ of V_i for $x+1 \leq i \leq n$) and broadcast

$\{x, 2||TPK_2, ..., (2n-2x)||TPK_{2n-2x}\}$. Upon receiving the broadcast message, V_i updates BT_n to BT_{n-1} using the same way as RSU_j does, puts the received TPKs to the right position of BT_{n-1}, and calculates the new TSK_0.

From the above analysis, we can get a conclusion that the designed key agreement mechanism makes every authenticated vehicle establish a common session key correctly.

B. Security Model

Definition 1 (Participants). There are three participants in the protocol, i.e., the vehicle V_i, RSU_j and the trust authority TA. And we suppose Π^t_Λ denote the instance t of participant $\Lambda \in \{V_i, RSU_j, TA\}$.

Definition 2 (Adversary Model). To analyze the security of our proposed protocol, we need to set up a series of interactive games between a challenger $\mathcal{C}$ which honestly executes the protocol and a adversary $\mathcal{A}$ which can eavesdrop, modify, and fabricate the messages transmitted in public channels. For the adversary model in our proposed scheme, we follow Bellare et al.'s model [14] [50] mostly except that add two oracles called $Join(\Pi^u_{V_i})$ and $Leave(\Pi^u_{V_i})$ to simulate the vehicle joining and leaving request respectively. Specifically, $\mathcal{A}$ can send oracle queries to $\mathcal{C}$ and $\mathcal{C}$ response the queries as follow.

1) *Execute(*$\Pi^u_{V_i}$,$\Pi^v_{RSU_j}$,Π^w_{TA}*)*: This query simulates the eavesdropping attack which means that $\mathcal{A}$ can obtain the messages transmitted among participants $\Pi^u_{V_i}$,$\Pi^v_{RSU_j}$,Π^w_{TA}. It is noteworthy that $\mathcal{A}$ can not obtain the messages in the registration phase of $\Pi^u_{V_i}$ or $\Pi^v_{RSU_j}$ since the communication of the registration phase is in a private and secure channel.
2) *Reveal(*Π^t_Λ*)*: Upon receiving this query from $\mathcal{A}$, the session key TSK_0 that is agreed by $\Pi^u_{V_i}$ and $\Pi^v_{RSU_j}$ is revealed to $\mathcal{A}$.
3) *Send(*Π^t_Λ*,m)*: If $\mathcal{A}$ sends the query with the message m, $\mathcal{C}$ sends the response message to $\mathcal{A}$ according to the corresponding protocol.
4) *Join*$(\Pi^u_{V_i})$: If $\mathcal{A}$ sends the query to $\mathcal{C}$ for updating the common session key, $\mathcal{C}$ executes *Send(*Π^t_Λ*,joining)*.
5) *Leave*$(\Pi^u_{V_i})$: If $\mathcal{A}$ sends the query to $\mathcal{C}$ for updating the common session key, $\mathcal{C}$ executes *Send(*Π^t_Λ*,leaving)*.
6) *Test*$(\Pi^u_{V_i})$: Upon receiving the query from $\mathcal{A}$, $\mathcal{C}$ flips a coin to get a value $b \in \{0, 1\}$ that is confidential to $\mathcal{A}$. If $b = 1$, $\mathcal{C}$ sends the session key to $\mathcal{A}$. If $b = 0$, $\mathcal{C}$ sends a random number that is in the same domain space with session key to $\mathcal{A}$.

Definition 3 (Partnership). Two instances Π^u_Λ and Π^v_Λ have partnership if one of the two conditions is held, i.e., Π^u_Λ can exchange messages with Π^v_Λ directly, or Π^u_Λ and Π^v_Λ both participate to compute the common session key.

Definition 4 (Freshness). The instance Π^t_Λ is considered to be fresh if neither Π^t_Λ nor its partner has asked *Reveal(*Π^t_Λ*)* queries.

Definition 5 (Semantic Security of AKA Protocol). An adversary $\mathcal{A}$ can send *Test(*$\Pi^u_{V_i}$*)* query to $\Pi^u_{V_i}$ or $\Pi^v_{RSU_j}$. At the end of the interactive game between $\mathcal{A}$ and $\mathcal{C}$, $\mathcal{A}$ is required to send a guess bit $\hat{b}$ to $\mathcal{C}$. Let W denote the event that $\mathcal{A}$ win the game, which means that the guess of $\mathcal{A}$ is equal to the random choice ($\bar{b} = b$). The advantage that $\mathcal{A}$ breaks the session key security of the AKA protocol P is calculated as $Adv^{AKA}_P(\mathcal{A}) = 2\left|Pr[W] - \frac{1}{2}\right|$. The AKA protocol P is semantic secure if $Adv^{AKA}_P(\mathcal{A})$ is negligible for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$.

C. Security Proof

Theorem 1. If there is a PPT adversary $\mathcal{A}$ that aims to break the proposed protocol with advantage $Adv^{AKA}_P(\mathcal{A})$ under the below security model, then a PPT adversary $\mathcal{B}$ can be constructed to solve ECDDHP with advantage $Adv^{ECCDHP}(\mathcal{B})$. And the relationship between $Adv^{AKA}_P(\mathcal{A})$ and $Adv^{ECDDHP}(\mathcal{B})$ satisfies

$$\begin{aligned} Adv^{AKA}_P(\mathcal{A}) = & \frac{\sum_{i=1}^{5} q^2_{h_i} + (q_s + 2N)^2}{q} + \\ & \frac{6q_{sign}(q_{sign} + q_{bf})}{(q-1)/2 - q_{sign} - q_{bf}} + \\ & \frac{2q_{h_6}}{q} + 4N \cdot Adv^{ECCDHP}(\mathcal{B}). \end{aligned} \tag{2}$$

where q_{h_i}, q_s, q_{sign}, q_{bf} and N denote the max number of hash, send, sign, bijective random oracle queries, and the max number of vehicles that establish a common session key, respectively.

Proof. To prove the theorem, we use a security proof technology called game hopping which contains a series of games between the adversary and the challenger. We define W_j to be the event that $\mathcal{A}$ guesses the bit correctly, i.e., $\bar{b} = b$. And these games are defined as follows.

Game 0. To break the protocol, $\mathcal{A}$ can send various oracle queries to $\mathcal{C}$ and $\mathcal{C}$ responses as follow methods. It is noteworthy that $\mathcal{C}$ has finished the work in the setup phase, owned the registration information of V_i and RSU_j, sent the public parameters to $\mathcal{A}$.

- *Send(*Π^t_Λ*,m)*. According to the differences of participants and messages, *Send(*Π^t_Λ*,m)* queries can be divided into the following types.
  1) *Send(*$\Pi^t_{V_i}$*,authentication request)*: This query simulates the scenario that a vehicle sends the authentication request to RSU_j, and $\mathcal{A}$ can get the message M_1 sent from V_i in the protocol through the query. Upon receiving the query, $\mathcal{C}$ chooses random $r_i, RID_i \in \mathbb{Z}^*_q$, calculates $R_i = r_iP$ and $l_i = h(s, RID_i)$, $PID_i = RID_i \oplus h_2(r_iP_{pub}, T_i)$, chooses $k_i \in \mathbb{Z}^*_q$,calculates $K_i = k_iP$, $\alpha_i = h_3(RID_i, T_i, R_i, K_i, l_i)$, and sends $M_1 = \{PID_i, T_i, R_i, K_i, \alpha_i\}$ to $\mathcal{A}$.
  2) *Send(*$\Pi^t_{RSU_j}$*,*M_1*)*: This query simulates the scenario that RSU_j sends the authentication request to TA, and $\mathcal{A}$ can get the message M_2 sent from V_i in the protocol through the query. Upon receiving the query, $\mathcal{C}$ uses the content contained in M_1, chooses random $ID_j, d \in \mathbb{Z}^*_q$, calculates $l_j = h(s, ID_j)$, $Q_j = dP$, and $\gamma_j = h_4(ID_j, T_j, \alpha_i, l_j)$, and sends $M_2 = \{ID_j, T_j, PID_i, T_i, \alpha_i, \gamma_j, R_i\}$ to $\mathcal{A}$.
  3) *Send(*Π^t_{TA}*,*M_2*)*: This query simulates the scenario that TA sends the authentication response to RSU_j, and $\mathcal{A}$ can get the message M_3 sent from TA in the protocol

through the query. Upon receiving the query, $\mathcal{C}$ uses the message in M_2, checks the validity of T_j, rejects $\mathcal{A}$'s query and returns $\perp$ if T_j is not legal. Otherwise, $\mathcal{C}$ calculates $l_j = h(s, ID_j)$ and $\gamma'_j = h_4(ID_j, T_j, \alpha_i, l_j)$, and checks if $\gamma'_j = \gamma_j$ holds. If not, rejects the query from $\mathcal{A}$. Otherwise, TA checks the validity of T_i , rejects $\mathcal{A}$'s query and returns $\perp$ if T_j is not legal. $\mathcal{C}$ calculates $RID'_i = PID_i \oplus h_2(sR_i, T_i), l_i = h(s, RID'_i)$, and $\alpha'_i = h_3(RID'_i, T_i, R_i, K_i, l_i)$, and checks whether the equation $\alpha'_i = \alpha_i$ holds. If not, $\mathcal{C}$ rejects the query from $\mathcal{A}$ and returns $\perp$. Otherwise, $\mathcal{C}$ calculates $\delta = h_4(T_{TA}, ID_j, Q_j, l_i)$ and $\eta = h_5(T_{TA}, PID_i, \alpha_i, l_j)$, and sends $M_3 = \{T_{TA}, \delta, \eta\}$ to $\mathcal{A}$.

4) *Send*($\Pi^t_{RSU_j}$,M_3): This query simulates the scenario that RSU_j authenticates the message M_3 sent from TA. Upon receiving the query, $\mathcal{C}$ computes $\eta' = h_5(T_{TA}, PID_i, \alpha_i, l_i)$, and checks if the equation $\eta' = \eta$ holds. If not, $\mathcal{C}$ rejects the query and returns $\perp$. Otherwise, $\mathcal{C}$ sends $M_4 = \{ID_j, Q_j, \delta\}$ to $\mathcal{A}$.
5) *Send*($\Pi^t_{V_i}$,M_4): This query simulates the scenario that V_i authenticates the message M_4 sent from RSU_j. Upon receiving the query, $\mathcal{C}$ computes $\delta' = h_4(T_{TA}, ID_j, Q_j, l_i)$ and checks whether $\delta' = \delta$ holds. If not, $\mathcal{C}$ rejects the query and returns $\perp$. Otherwise, $\mathcal{C}$ outputs a signal to $\mathcal{A}$ in order to confirm the legality of M_4.

- *Join*($\Pi^u_{V_i}$). Upon receiving the query, $\mathcal{C}$ computes the secret key TSK_{2i} and public key TPK_{2i} of current binary tree BT_n where $0 \leq i \leq n-1$, chooses a random number $b_j \in \mathbb{Z}^*_q$, calculates $B_j = b_j P$ and gets x-axis value B^x_j of point B_j, calculates $s_j = b^{-1}_j \cdot (h(T_j, TPK_1, TPK_2) + B^x_j \cdot d)$ to get the signature $\sigma_j = (B^x_j, s_j)$, and sends $M_5 = \{TPK_1, TPK_2, n, T_j, \sigma_j\}$ to $\mathcal{A}$.
- *Leave*($\Pi^u_{V_i}$). Upon receiving the query, $\mathcal{C}$ calculates private and public keys of all branch nodes $N_0, N_2, ..., N_{2n-2x+2}$ of the current binary tree BT_n, signs these keys using the same method in *Join*($\Pi^u_{V_i}$), and sends $M_6 = \{x, T_j, \sigma_j, TPK_2, TPK_4, ..., TPK_{2n-2x+2}\}$ to $\mathcal{A}$.
- *Execute*($\Pi^u_{V_i}$,$\Pi^v_{RSU_j}$,Π^w_{TA}). Upon receiving the query, $\mathcal{C}$ sends the newest $\{M_1, M_2, M_3, M_4, M_5, M_6\}$ to $\mathcal{A}$.
- *Test*($\Pi^u_{V_i}$): Upon receiving the query from $\mathcal{A}$, $\mathcal{C}$ flips a coin to get a value $b \in \{0, 1\}$ that is confidential to $\mathcal{A}$. If $b = 1$, $\mathcal{C}$ sends the current session key TSK_0 to $\mathcal{A}$. If $b = 0$, $\mathcal{C}$ sends a random number that is in the same domain space with TSK_0 to $\mathcal{A}$.

It is obvious that Game 0 simulates the original attack and the advantage for $\mathcal{A}$ to break Game 0 is the same as that of Definition 5. Thus we have

$$Adv^{AKA}_P(\mathcal{A}) = 2\left|Pr[W_0] - \frac{1}{2}\right|. \tag{3}$$

Game 1. Game 1 is the same as Game 0 except that hash functions $h_i (1 \leq i \leq 5)$ are simulated as random oracles. And the hash lists $L_{h_i} (1 \leq i \leq 5)$ are maintained by $\mathcal{C}$. When $\mathcal{C}$ receives hash query m from $\mathcal{A}$, $\mathcal{C}$ looks up the tuple $(m, h_i(m))$ from L_{h_i}. If found, $\mathcal{C}$ sends $(m, h_i(m))$ to $\mathcal{A}$. Otherwise, $\mathcal{C}$ chooses a random number $h_i(m) \in \mathbb{Z}^*_q$, saves $(m, h_i(m))$ in L_{h_i}, and sends $h_i(m)$ to $\mathcal{A}$. It is obvious that the setting does not affect the advantage of $\mathcal{A}$. Thus we have

$$Pr[W_1] = Pr[W_0]. \tag{4}$$

Game 2. Game 2 is the same as Game 1 except that one additional condition is added, i.e., no collision exists. It is easy to see that the collision probability of one $h_i (1 \leq i \leq 6)$ is at most $\frac{q^2_{h_i}}{2q}$ according to birthday paradox. Besides, messages $M_i (1 \leq i \leq 6)$ sent from $\mathcal{C}$ may collide and the collision probability is at most $\frac{(q_s+q_j+q_l)^2}{2q}$, where q_j and q_l denote the number of joining query and the number of leaving query respectively. Since the number of non-repeating join queries and leaving queries is bounded on the max number of vehicles N, we get $\frac{(q_s+q_j+q_l)^2}{2q} \leq \frac{(q_s+2N)^2}{2q}$. Hence we have

$$|Pr[W_2] - Pr[W_1]| \leq \frac{\sum^5_{i=1} q^2_{h_i} + (q_s + 2N)^2}{2q}. \tag{5}$$

Game 3. Game 3 is the same as Game 2 except that one additional condition is added, i.e., EU-CMA (Existential Unforgeability under Chosen Message Attack) is satisfied for the digital signature scheme used in our scheme. In our proposed scheme, we use ECDSA as the signature algorithm. As proved by [51], under the random oracle model, the advantage for $\mathcal{A}$ to break the ECDSA is at most $\frac{3q_{sign}(q_{sign}+q_{bf})}{(q-1)/2-q_{sign}-q_{bf})}$ where q_s denotes the maximum number of signing queries, q_{bf} denotes the maximum number of bijective random oracle queries. Hence we can get

$$|Pr[W_3] - Pr[W_2]| \leq \frac{3q_{sign}(q_{sign} + q_{bf})}{(q-1)/2 - q_{sign} - q_{bf}}. \tag{6}$$

In Game 3, by sending *Join*($\Pi^u_{V_i}$) query or *Leave*($\Pi^u_{V_i}$) query together with Send(Π^t_{Λ},m) query, $\mathcal{A}$ can get the public keys of all branch nodes and leaf nodes of BT_n which can be used to compute the group key TSK_0. For $\mathcal{A}$, there are two possible ways to get the common session key $TSK_0 = h_6(TSK_1 TSK_2 P)$ so that $\mathcal{A}$ can distinguish between the session key TSK_0 and a random number.

1) $\mathcal{A}$ sends h_6 query to $\mathcal{C}$ to guess the value $h_6(TSK_1 TSK_2 P)$ directly. Since h_6 has been simulated as random oracle with collision resistance, the advantage for $\mathcal{A}$ to win the game using this method is at most $\frac{q_{h_6}}{q}$.
2) $\mathcal{A}$ gets the public key of every node of BT_n, tries to compute $TSK_1 TSK_2 P$ and gets $h_6(TSK_1 TSK_2 P)$. The most efficient method for $\mathcal{A}$ to compute $TSK_1 TSK_2 P$ directly is by using public keys of two child nodes of the root, i.e., $TSK_1 \cdot P$ and $TSK_2 \cdot P$. It is easy to see that the method is actually to solve ECCDHP. So the advantage for $\mathcal{A}$ to win the game using this method is at most $(q_j + q_l) \cdot Adv^{ECCDHP}(\mathcal{B}) = 2N \cdot Adv^{ECCDHP}(\mathcal{B})$.

Hence we have

$$Pr[W_3] \leq \frac{1}{2} + \frac{q_{h_6}}{q} + 2N \cdot Adv^{ECCDHP}(\mathcal{B}). \tag{7}$$

By combining equations (3)-(6), we have

$$\begin{aligned}
Adv_P^{AKA}(\mathcal{A}) &= 2|Pr[W_0] - \frac{1}{2}| \\
&= 2|Pr[W_1] - \frac{1}{2}| \\
&= 2|Pr[W_1] - Pr[W_2] + Pr[W_2] - Pr[W_3] + \\
&\quad Pr[W_3] - \frac{1}{2}| \\
&\leq 2(|Pr[W_2] - Pr[W_1]| + |Pr[W_3] - Pr[W_2]| + \\
&\quad |Pr[W_3] - \frac{1}{2}|) \\
&\leq \frac{\sum_{i=1}^{5} q_{h_i}^2}{q} + \frac{6q_{sign}(q_{sign} + q_{bf})}{(q-1)/2 - q_{sign} - q_{bf}} + \\
&\quad \frac{2q_{h_6} + (q_s + 2N)^2}{q} + 4N \cdot Adv^{ECCDHP}(\mathcal{B}).
\end{aligned} \tag{8}$$

D. Security Analysis

In this subsection, we show that our proposed protocol satisfies the following security and privacy requirements.

1) **Conditional Anonymity or Privacy-Perserving:** Conditional anonymity or privacy-perserving means that only TA can reveal the real identity according the received pseudo identity. Every message sent from V_i contains the pseudo identity $PID_i = RID_i \oplus h_2(r_i P_{pub}, T_i) = RID_i \oplus h_2(sR_i, T_i)$ instead of using the real identity RID_i. Without knowing s which is generated by TA and keeps confidential to all entities, RID_i is an one-time-pad ciphertext for malicious adversaries. The security of collision-resistant hash function and the computational hardness of solving ECCDHP make any adversary has negligible advantage to reveal RID_i from PID_i. However, TA can get RID_i by calculating $RID_i = PID_i \oplus h_2(sR_i, T_i)$. Hence, conditional anonymity of vehicle is satisfied.
2) **Mutual Authentication:** Mutual authentication means that the message receiver can verify the integrity of the received message and the identity validity of the message sender. In the three-party authentication phase, every message is authenticated by received entity by checking the hash value. It is worth noting that pure hash function can not guarantee any message authentication. In our proposed protocol, long-term secret keys together with secure hash functions guarantee that any content contained in the message cannot be modified. The pseudo identity and message of a vehicle or RSU are checked by TA by checking the hash value, and the hash values (secured by long-term secret keys) sent by TA ensure that vehicle and RSU can achieve mutual authentication.
3) **Resistance of Replay Attack:** Resistance of replay attack means that an adversary cannot intercept a valid message and resend it later to attempt to make the message be valid again. Every message M sent from V_i, RSU_j or TA is attached with a timestamp T_i. A receiver can check the timeliness of M by checking whether the equation $|T_j - T_{cur}| \leq \Delta T$ holds, where T_{cur} denotes the current timestamp and ΔT denotes the max legal time interval. So the malicious adversary can not replay an expired message to perform any meaningful attack.
4) **Resistance of Man-in-the-middle Attack:** Resistance of man-in-the-middle attack means that the active adversary cannot forge a valid message after intercepting valid messages from valid senders. Messages sent from V_i and RSU_j are identified by PID_i and ID_j respectively. We claim that the malicious adversary can not use PID_i or ID_j to forge a message. To forge a valid message identified by PID_i, the adversary has to know the real identity RID_i and l_i. However, the only ways for the adversary to get RID_i are by guessing or computing based on PID_i. As for l_i, it is confidential for the adversary. So the adversary is computationally efficient to forge a message sent from a vehicle. To forge the authentication message from RSU_j, the adversary has to get l_j to forge a valid hash value γ_j. However, l_j is confidential for the adversary. To forge the key agreement message from RSU_j, the adversary has to make vehicles trust its public key which is used to validate the signature. However, the public key Q_j is guaranteed by δ which is sent from TA. Thus, our protocol is secure against man-in-the-middle attack.
5) **Backward Secrecy:** Backward secrecy is a property such that a new addition to a group is unable to decrypt information created prior to their introduction. In the key agreement phase of our protocol, the common group key can only be computed successfully by authenticated vehicles because the common group key is computed using public keys and private keys of authenticated vehicles. So any un-authenticated vehicle or the adversary can not compute the common group key, so that backward secrecy is guaranteed.
6) **Forward Secrecy:** Forward secrecy is a property that someone who already had a key doesn't have access to future keys for the group. Once an authenticated vehicle sends the leaving request, the common session key would be updated. According to the computation method of the vehicle leaving process in our protocol, the key of an authenticated vehicle that participates the common key agreement process is out of the computation process of new common group key. So any leaving vehicle can not compute the new common session key with material broadcasted by RSU_j.
7) **Perfect Forward Secrecy:** Perfect forward secrecy means that the leakage of the long-term session key does not affect the old keys that were established in the key agreement processes. In our proposed protocol, the long-term session key l_i or l_j is just used for the authentication process, which means its leakage will not leak the common group key. So perfect forward secrecy is achieved in our proposed protocol.
8) **Un-traceability:** Un-traceability means that the passive adversary cannot monitor the communication channel to trace a vehicle by analyzing message sender. In the three-party authentication phase, the timestamp contained in the pseudo identity $PID_i = RID_i \oplus h_2(r_i P_{pub}, T_i)$ keeps

changing, so that different messages sent from the same vehicle are identified by different pseudo identities. As for the public keys K_i that are used for participating the key agreement process, they are generated by a random way and only sent to TA once, which means that it is hard for the adversary to trace a vehicle by analyzing K_i. In the key agreement phase, the broadcast message sent from TA do not contain any identity information of the vehicle. So un-traceability is achieved in our proposed protocol.

9) **Resistance of Stolen-verifier Attack:** Resistance of stolen-verifier attack means that the leakage of the real identity of a vehicle or the identity of a RSU will not help the adversary to break the protocol. The long-term session key l_i or l_j is the basic requirement to protect the interactions among vehicle, RSU, and TA. The leakage of the real identity RID_i of a vehicle or the identity ID_j of a RSU will not leak $l_i = h_1(s, RID_i)$ or $l_j = h_1(s, ID_j)$, because the system secret key s that is essential for computing l_i and l_j is only known by the TA. So resistance of stolen-verifier attack is guaranteed.

10) **Resistance of Modification Attack:** Resistance of modification attack means that the active adversary can not modify a message to cheat the message receiver. In the three-party authentication phase, the interacted message mainly involves (pseudo) identities, timestamps, public keys, and hash values. The active adversary can not modify these content successfully because the pre-shared long-term session key and secure one-way hash function guarantees that the modification of these content will break failure of the check of the corresponding hash value. In the key agreement phase for new vehicle joining or old vehicle leaving, the active adversary can not modify the content broadcasted by RSU successfully because the content is secured by ECDSA signature.

In the 3GPP's C-V2X R14 document [52], security and privacy requirements for V2X communication were defined. Specifically, four requirements should be satisfied for security, i.e., the identity of the message sender should be verified, message integrity should be ensured, replay attack should be detected, and data confidentiality (for some application) should be protected. Three requirements should be satisfied for privacy, i.e., the leakage risk of user's permanent identity should be low, pseudonymity of user should be used, and user's identity should be protected from eavesdropping. So we can find that the security analysis part can satisfy not only the security and privacy requirements defined by 3GPP, but also other advanced requirements.

VI. Performance Analysis

In this section, we give the performance analysis of our proposed protocol. To show the performance comprehensively, we give the performance analysis in three scenarios, i.e., scenario 1, scenario 2, and scenario 3. In scenario 1, we only consider the key agreement process with only three parties, i.e., one vehicle, one RSU, and one TA. In scenario 2, we consider the situation that multiple vehicles agree on a common session key with one RSU. In scenario 3, we consider the situation where some vehicles leave the group or exceed the communication range. It is obvious that the three scenarios cover all situations of our proposed protocol. Before giving the analysis in different scenarios, we firstly give the experimental environment and related parameters.

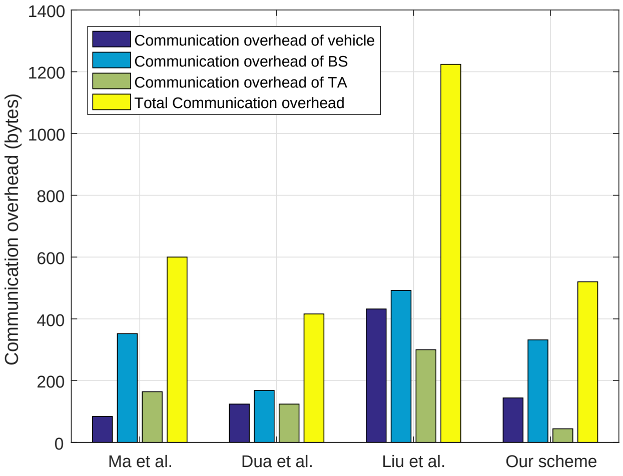


Fig. 7. The Comparison of Communication Overhead

For measuring the computational and communication overhead of cryptography operation, we have the following settings for cryptographic operations related to bilinear pairing and elliptic curve respectively. To achieve 80-bit security level, we use a symmetric bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ where $\mathbb{G}$ is a additive group generated by a point $\overline{P}$ with order $\overline{q}$ over the super singular elliptic curve $y^2 = x^3 + x \ mod \ \overline{p}$ with embedding degree 2, $\overline{q}$ is a 160-bit prime number and $\overline{p}$ is a 512-bit prime number. At the same time, we use an additive group $\mathbb{G}$ which is generated by a point P with order q over the non-singular elliptic curve $y^2 = x^3 + ax + b \ mod \ p$, where p and q are two 160-bit primes, and $a, b \in \mathbb{Z}_p^*$. We measure all involved cryptography operations of our scheme and comparison schemes using two C/C++ cryptography libraries called MIRACL [53] and Crypto++ [54]. Under the Ubuntu 14.04 environment with Intel i7-6700 CPU and 8GB DDR4 RAM, we have the results that are as shown in Table II.

For measuring the network performance of our proposed protocol, we use three tools, i.e., Omnet++ [55], Sumo [56], and Veins [57]. OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators which supports the simulation for wired network and wireless ad-hoc network. Sumo is an open source, highly portable, microscopic and continuous road traffic simulation package designed to handle large road networks. Veins acts as a middleware between Omnet++ and Sumo. And the parameters of simulation are as shown in Table III.

A. Analysis for Scenario 1

In this subsection, we compare our proposed scheme with Ma et al. 's scheme [14], Dua et al. 's scheme [12], and Liu et al. 's scheme [36] in scenario 1. In scenario 1, one vehicle, one RSU (in our and Liu et al. 's [36] model) or one fog node (in Ma et al. 's model [14]) or one cluster head (in Dua et al. 's scheme [12]), and one TA participate to agree a common session key for V2I communication. It is noteworthy that, in our protocol, scenario 1 is equal to the situation that only one authenticated vehicle sends the joining request.

First, we analyze the computation overhead of generating a common session key in the above four schemes. In the authentication phase and key agreement phase of Ma et al. 's scheme [14], the vehicle needs to perform three scalar multiplication operations, four general one-way hash operations, which means that the total computational overhead for a vehicle is $3T_{sm-ecc} + 4T_h \approx 1.061$ms. As for fog node, it needs to perform four scalar multiplication operations, four general one-way hash operations, which means that the total computational overhead for a fog node is $4T_{sm-ecc} + 4T_h \approx 1.412$ms. Besides, TA needs to perform eight scalar multiplication operations, nine general one-way hash operations, which means that the total computational overhead for TA is $8T_{sm-ecc} + 9T_h \approx 2.826$ms. Thus the total computation overhead in Ma et al. 's scheme [14] for authenticated key agreement is about 5.299ms. In the authentication phase and key agreement phase of Dua et al. 's scheme [12], the vehicle needs to perform four scalar multiplication operations, six general one-way hash operations, which means that the total computational overhead for a vehicle is $4T_{sm-ecc} + 6T_h \approx 1.416$ms. As for cluster head, it needs to perform five scalar multiplication operations, ten general one-way hash operations, which means that the total computational overhead for a cluster head is $5T_{sm-ecc} + 10T_h \approx 1.775$ms. Besides, TA needs to perform three scalar multiplication operations, two general one-way hash operations, which means that the total computational overhead for TA is $8T_{sm-ecc} + 9T_h \approx 2.826$ms. Thus the total computation overhead in Dua et al. 's scheme [12] for authenticated key agreement is about 4.248ms. In the authentication phase and key agreement phase of Liu et al. 's scheme [36], the vehicle needs to perform two map-to-point operations, one bilinear pairing operation, one encryption/decryption operation, and one scalar multiplication operation related to bilinear pairing, which means that the total computational overhead for a vehicle is $2T_{mtp} + T_{bp} + T_{enc} + T_{sm-bp} \approx 6.199$ms. As for RSU, it needs to perform two map-to-point operations, one bilinear pairing operation, one encryption/decryption operation, which means that the total computational overhead for a RSU is $2T_{mtp} + 2T_{bp} + T_{enc} \approx 5.573$ms. Besides, TA needs to perform three map-to-point operations, two scalar multiplication operations related to bilinear pairing, one encryption/decryption operation, which means that the total computational overhead for TA is $3T_{mtp} + T_{sm-bp} + T_{enc} \approx 1.181$ms. Thus the total computation overhead in Liu et al. 's scheme [36] for authenticated key agreement is about 12.953ms. In the authentication phase and key agreement phase of our proposed scheme, the vehicle needs to perform six scalar multiplication operations, five general one-way hash operations, which means that the total computational overhead for a vehicle is $6T_{sm-ecc} + 5T_h \approx 2.116$ms. As for RSU, it needs to perform two scalar multiplication operations, four general one-way hash operations, which means that the total computational overhead for a fog node is $2T_{sm-ecc} + 4T_h \approx 0.71$ms. Besides, TA needs to perform one scalar multiplication operation, three general one-way hash operations, which means that the total computational overhead for TA is $T_{sm-ecc} + 3T_h \approx 0.357$ms. Thus the total computation overhead in our scheme for authenticated key agreement is about 3.183ms.

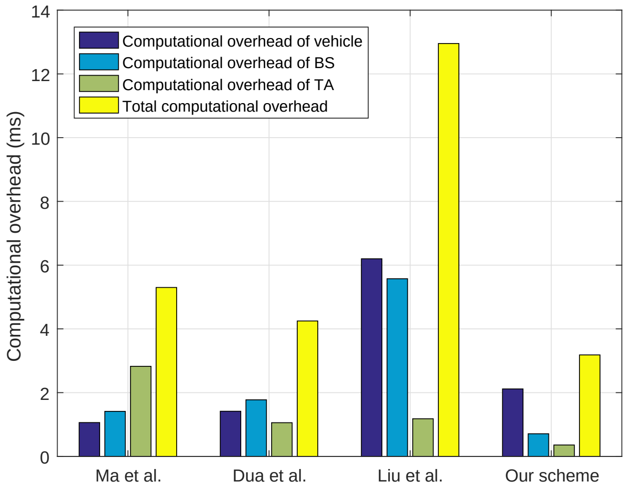


Fig. 8. The Comparison of Computational Overhead

The above results give us the performance comparison of the four schemes when they are all performed in scenario 1, as shown in Fig. 8. We can see that our proposed scheme has a computational advantage compared with schemes [12], [36], [14] in terms of RSU's, TA's and total computational overhead.

Next, we analyze the communication overhead of generating a common session key in the above four schemes. Since the length of $\overline{p}$ and p are 64 bytes and 20 bytes respectively, the size of group $\mathbb{G}_1$ and $\mathbb{G}$ are 128 bytes and 40 bytes respectively. (The elements in $\mathbb{G}_1$ and $\mathbb{G}$ are points on elliptic curve.) We set the length of timestamp to be 4 bytes, the size of hash value to be 20 bytes. According to the standard of block cipher, the length of one block is 128 bits (16 bytes).

In the authentication phase and key agreement phase of Ma et al. 's scheme [14], the communication overhead of the vehicle, fog node, TA are $|\mathbb{G}| + 2 * |\mathbb{Z}_q^*| + |T_i| = 84$ bytes, $6|\mathbb{G}| + 5 * |\mathbb{Z}_q^*| + 3 * |T_j| = 352$ bytes, and $3|\mathbb{G}| + 2 * |\mathbb{Z}_q^*| + |T_{TA}| = 164$ bytes. Thus the total communication overhead of Ma et al. 's scheme [14] is 600 bytes. As for Dua et al. 's scheme [12], the communication overhead of the vehicle, cluster head, TA are $2 * |\mathbb{G}| + 2 * |\mathbb{Z}_q^*| + |T_i| = 124$ bytes, $2 * |\mathbb{G}| + 4 * |\mathbb{Z}_q^*| + 2 * |T_i| = 168$ bytes and $|\mathbb{G}| + 4 * |\mathbb{Z}_q^*| + |T_i| = 124$ bytes. Thus the total communication overhead of Dua et al. 's scheme [12] is 416 bytes. For Liu et al. 's scheme [36], the communication overhead of the vehicle, RSU, TA are $3 * |\mathbb{G}_1| + 2 * |\mathbb{Z}_q^*| + 2 * |T_i| = 432$ bytes, $3 * |\mathbb{G}_1| + 5 * |\mathbb{Z}_q^*| + 2 * |T_j| = 492$ bytes and $2 * |\mathbb{G}_1| + 2 * |\mathbb{Z}_q^*| + |T_i| = 300$ bytes. Thus the total communication overhead of Liu et al. 's scheme [36] is 1224 bytes. In our scheme, the communication overhead of the vehicle, RSU, TA are $2 * |\mathbb{G}| + 3 * |\mathbb{Z}_q^*| + |T_i| = 144$ bytes, $4|\mathbb{G}| + 8 * |\mathbb{Z}_q^*| + 3 * |T_j| = 332$ bytes, and $2 * |\mathbb{Z}_q^*| + |T_{TA}| = 44$

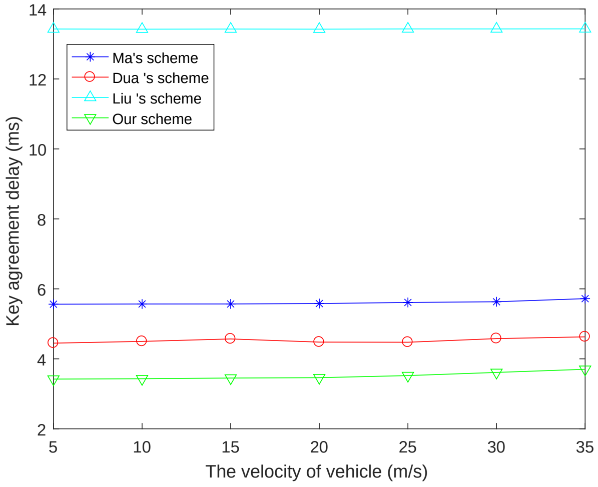


Fig. 9. The Comparison of Key Agreement Delay in Scenario 1

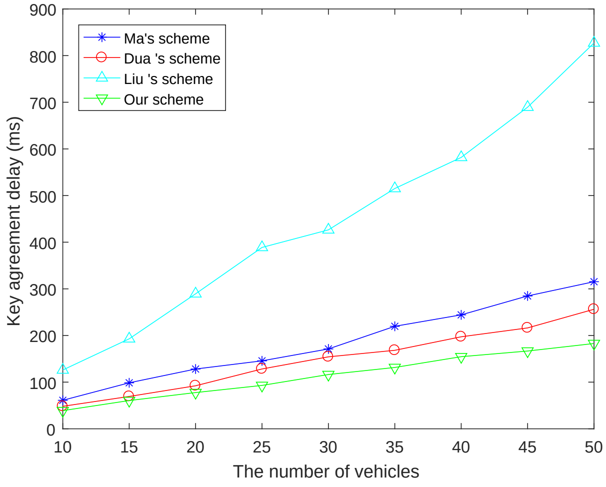


Fig. 10. The Comparison of Key Agreement Delay in Scenario 2

bytes. Thus the total communication overhead of our proposed scheme is 520 bytes.

The above results give us the performance comparison of the four schemes when they are all performed in scenario 1, as shown in Fig. 7. We can see that our scheme has the lowest communication overhead of TA, and not a bad advantage in terms of total communication overhead compared with schemes [12], [36], [14].

Last, we compare our scheme with schemes [12], [36], [14] in simulation environment. We define key agreement delay as the sum of message transmission delay and computation delay for computing the common session key. From Fig. 9, we can get two conclusions. One is that the key agreement delay of our scheme is the lowest among the four schemes. The other is that the velocity of the vehicle (in legal range) has an impact on key agreement delay to some extent. Specifically, in Liu et al. 's scheme [36], the velocity of the vehicle has almost no affection on the key agreement delay because the computation delay is much higher than the message transmission delay such that the fluctuation of transmission delay is negligible. In Dua et al. 's scheme [12], the velocity of the vehicle affects the key agreement delay to a relatively obvious extent because the key agreement process involves both V2V and V2I communications (i.e., more transmission delay will be produced). In Ma et al. [14] and our proposed schemes, the velocity of the vehicle affects the key agreement delay to a low extent because the two schemes both only involve V2I communication.

B. Analysis for Scenario 2

In this subsection, we compare our proposed scheme with Ma et al. 's scheme [14], Dua et al. 's scheme [12], and Liu et al. 's scheme [36] in scenario 2, where some services (that should be secured using cryptographic mechanisms) need to be performed among multiple vehicles, one RSU (in our and Liu et al. 's [36] model) or one fog node (in Ma et al. 's model [14]) or one cluster head (in Dua et al. 's scheme [12]), and one TA. It is noted that, in our protocol, scenario 2 is equal to the situation that multiple vehicles pass the Three-Party Authentication Phase and enter Key Agreement Phase for New Vehicle Joining orderly to agree the common session key.

For simulation experiment, we set the max number of vehicles covered by one RSU to be 50, the velocity of vehicles to be 20m/s (we record all related data when the instant velocity of vehicle is about 20m/s). Let key agreement delay in scenario 2 denote as all necessary time cost for generating the common group key of one RSU and multiple vehicles, which is composed of transmission cost and computational cost. And the simulation results are as shown in Fig. 10. From the figure, we can see that the key agreement delay of our proposed scheme is the lowest among the four schemes. The reason why our scheme has an advantage is that all updated vehicles can rapidly compute the common session key by the material sent from nearby RSU, while other schemes [12], [36], [14] have to establish session keys for vehicles and RSU (cluster head/fog node) one by one. We can also observe that the number of vehicles affects the key agreement delay obviously in this scenario. The reason is that the increasing number of vehicles makes RSU (cluster head/fog node) receive more messages, such that the channel congestion is more serious and processing delay is much higher. It is noteworthy that the velocity of vehicles has almost no affection on the key agreement delay because the computational delay is much higher than the transmission delay in this scenario.

C. Analysis for Scenario 3

In this subsection, we analyze the performance of our proposed scheme in scenario 3. In scenario 3, some of authenticated vehicles that have participated the key agreement process in scenario 2 become leaving vehicles, so the rest of joined vehicles have to update the common session key to guarantee session key security. Since schemes [12], [36], [14]

TABLE III. Simulation Parameters

Parameter	Value
Simulation area	$2500 \times 2500(m^2)$
Path loss model	Two-Ray interference
Obstacle shadowing model	Simple obstacle shadowing
Max interference distance	2600m
Transmission power	50mW
Data transmission rate	24Mbps
Sensitivity	-89dBm
Thermal noise	-110dBm
Beacon interval	1s
Used channel	CCH
Acceleration speed of vehicle	$10m/s^2$
Simulation time	200s

did not consider scenario 3, we only analyze the performance of our proposed scheme.

For the simulation experiment, we set the number of initial vehicles covered by one RSU to be 20 and the velocity of all vehicles to be 20m/s. Let the key agreement delay in scenario 3 denote all necessary costs for updating the common session key. According to the key agreement algorithm for the old vehicle leaving operation, different vehicles pay different computing costs for the leaving operations of different vehicles. So the key agreement delay cost in scenario 3 will be computed in three cases, i.e., the best case, the worst case, and the average cost, where the best case is corresponding to First In First Out case (the first vehicle to join, the first to leave), the worst case is corresponding to First In Last Out case (the first vehicle to join, the last to leave), and the average is the mixed case. We simulate our scheme in the simulation environment in terms of the three cases. From the simulation results as shown in Fig. 11, we can see that, in the average case that is the closest to reality, the key agreement delay is still acceptable even if most of vehicles send the leaving request in a short time. We can also observe that the number of leaving vehicles affects the key agreement delay to a great extent in the worst and average cases, but not in the best case. The reason is that the increasing number of messages will bring higher procession delay in the worst and average cases, while the procession delay for the leaving event is almost constant in the best case (since RSU does not need to broadcast the public key of the branch node and the vehicle does not need to perform the scalar multiplication for updating the binary tree).

VII. Conclusion

In this paper, we proposed a protocol that focuses on securing V2V and V2I communications simultaneously in VANETs. Our research aimed to establish a common session key for authenticated vehicles and RSUs, and the established key, together with existing message authentication algorithms and encryption algorithms, can be used to secure V2V and V2I communications simultaneously. The essential components of our protocol can be divided into two parts. The first part is the three-party authentication process in which three parties, including vehicles, RSUs, and TAs, authenticate each other. The second part is the key agreement process, which is used in the key generation or updating process. For this part, we designed a tree-based, lightweight, secure, and practical key agreement algorithm that considers two scenarios, i.e., the joining of an authenticated vehicle and the leaving of a joined vehicle. The security proof showed that our proposed scheme satisfies session key security, and the security analysis demonstrated that our proposed scheme satisfies essential security requirements. Finally, the simulation experiments illustrated that our proposed scheme has a performance advantage over three related schemes. In the future, we will work to design a more efficient key generating and updating algorithm that involves lower computational and communication overhead.

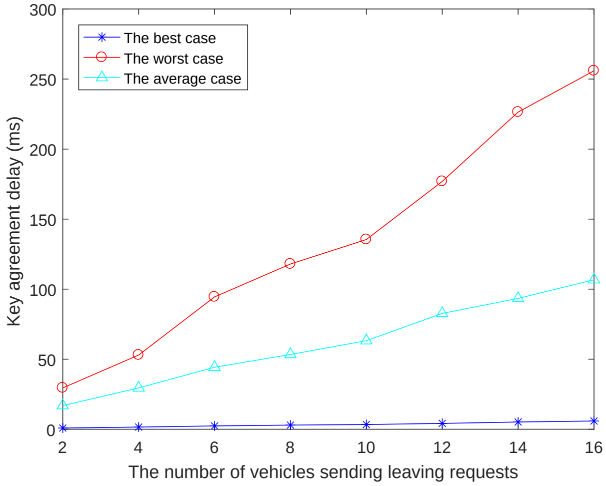


Fig. 11. The Key Agreement Delay of The Proposed Scheme in Scenario 3

VIII. Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 61872001, Grant 62011530046, and Grant U1936220 and in part by the Open Fund of Key Laboratory of Embedded System and Service Computing, Ministry of Education, Tongji University, under Grant ESSCKF2018-03. The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.

References

[1] 3GPP, “Service requirements for v2x services - stage 1 (release 14),” TS 22.185 V14.3.0, 2017.

[2] T. Yoshizawa and B. Preneel, “Survey of security aspect of v2x standards and related issues,” in *2019 IEEE Conference on Standards for Communications and Networking (CSCN)*. IEEE, 2019, pp. 1–5.

[3] T. Nawaz, M. Seminara, S. Caputo, L. Mucchi, F. S. Cataliotti, and J. Catani, “Ieee 802.15. 7-compliant ultra-low latency relaying vlc system for safety-critical its,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 12, pp. 12 040–12 051, 2019.

[4] T. Nawaz, M. Seminara, S. Caputo, L. Mucchi, and J. Catani, “Low-latency vlc system with fresnel receiver for i2v its applications,” *Journal of Sensor and Actuator Networks*, vol. 9, no. 3, p. 35, 2020.

[5] M. Seminara, T. Nawaz, S. Caputo, L. Mucchi, and J. Catani, “Characterization of field of view in visible light communication systems for intelligent transportation systems,” *IEEE Photonics Journal*, vol. 12, no. 4, pp. 1–16, 2020.

TABLE II. The Execution Time of Several Cryptographic Operations

Operation	Definition	Symbolic description	Execution time
T_{sm-ecc}	The execution time of scale multiplication operation $x \cdot P$ related to ECC	$x \cdot P, i.e., P + P + ... + P$	0.351
T_h	The execution time of one-way hash function	$\{0,1\}^* \rightarrow \mathbb{Z}_q^*$	0.002
T_{bp}	The execution time of bilinear paring operation	$e(A, B)$ where $A, B \in \mathbb{G}$	5.121
T_{sm-bp}	The execution time of scale multiplication operation $x \cdot P$ related to bilinear pairing	$x \cdot P$ where $P \in \mathbb{G}_1$	0.626
T_{mtp}	The execution time of map-to-point function	$\{0,1\}^* \rightarrow \mathbb{G}_1$	0.103
T_{ed}	The execution time of encryption or decryption operation of AES-CBC	$Enc_k(\cdot)$ or $Dec_k(\cdot)$	0.246

[6] J. Cui, L. Wei, J. Zhang, Y. Xu, and H. Zhong, "An efficient message-authentication scheme based on edge computing for vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1621–1632, 2018.

[7] L. Zhang, X. Men, K.-K. R. Choo, Y. Zhang, and F. Dai, "Privacy-preserving cloud establishment and data dissemination scheme for vehicular cloud," *IEEE Transactions on Dependable and Secure Computing*, 2018.

[8] J. B. Kenney, "Dedicated short-range communications (dsrc) standards in the united states," *Proceedings of the IEEE*, vol. 99, no. 7, pp. 1162–1182, 2011.

[9] J. Cui, X. Zhang, H. Zhong, J. Zhang, and L. Liu, "Extensible conditional privacy protection authentication scheme for secure vehicular networks in a multi-cloud environment," *IEEE Transactions on Information Forensics and Security*, 2019.

[10] R. Canetti and H. Krawczyk, "Universally composable notions of key exchange and secure channels," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2002, pp. 337–351.

[11] M. N. Mejri, N. Achir, and M. Hamdi, "A new group diffie-hellman key generation proposal for secure vanet communications," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2016, pp. 992–995.

[12] A. Dua, N. Kumar, A. K. Das, and W. Susilo, "Secure message communication protocol among vehicles in smart city," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4359–4373, 2017.

[13] S. H. Islam, M. S. Obaidat, P. Vijayakumar, E. Abdulhay, F. Li, and M. K. C. Reddy, "A robust and efficient password-based conditional privacy preserving authentication and group-key agreement protocol for vanets," *Future Generation Computer Systems*, vol. 84, pp. 216–227, 2018.

[14] M. Ma, D. He, H. Wang, N. Kumar, and K.-K. R. Choo, "An efficient and provably-secure authenticated key agreement protocol for fog-based vehicular ad-hoc networks," *IEEE Internet of Things Journal*, 2019.

[15] V. Vukadinovic, K. Bakowski, P. Marsch, I. D. Garcia, H. Xu, M. Sybis, P. Sroka, K. Wesolowski, D. Lister, and I. Thibault, "3gpp c-v2x and ieee 802.11 p for vehicle-to-vehicle communications in highway platooning scenarios," *Ad Hoc Networks*, vol. 74, pp. 17–29, 2018.

[16] Y. Kim, A. Perrig, and G. Tsudik, "Tree-based group key agreement," *ACM Transactions on Information and System Security (TISSEC)*, vol. 7, no. 1, pp. 60–96, 2004.

[17] M. Raya and J.-P. Hubaux, "Securing vehicular ad hoc networks," *Journal of computer security*, vol. 15, no. 1, pp. 39–68, 2007.

[18] K. Plößl and H. Federrath, "A privacy aware and efficient security infrastructure for vehicular ad hoc networks," *Computer Standards & Interfaces*, vol. 30, no. 6, pp. 390–397, 2008.

[19] R. Lu, X. Lin, H. Zhu, P.-H. Ho, and X. Shen, "Ecpp: Efficient conditional privacy preservation protocol for secure vehicular communications," in *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*. IEEE, 2008, pp. 1229–1237.

[20] C. Zhang, R. Lu, X. Lin, P.-H. Ho, and X. Shen, "An efficient identity-based batch verification scheme for vehicular sensor networks," in *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*. IEEE, 2008, pp. 246–250.

[21] C.-C. Lee and Y.-M. Lai, "Toward a secure batch verification with group testing for vanet," *Wireless networks*, vol. 19, no. 6, pp. 1441–1449, 2013.

[22] T. W. Chim, S.-M. Yiu, L. C. Hui, and V. O. Li, "Specs: Secure and privacy enhancing communications schemes for vanets," *Ad Hoc Networks*, vol. 9, no. 2, pp. 189–203, 2011.

[23] D. He, S. Zeadally, B. Xu, and X. Huang, "An efficient identity-based conditional privacy-preserving authentication scheme for vehicular ad hoc networks," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2681–2691, 2015.

[24] J. Li, K.-K. R. Choo, W. Zhang, S. Kumari, J. J. Rodrigues, M. K. Khan, and D. Hogrefe, "Epa-cppa: An efficient, provably-secure and anonymous conditional privacy-preserving authentication scheme for vehicular ad hoc networks," *Vehicular Communications*, vol. 13, pp. 104–113, 2018.

[25] M. R. Asaar, M. Salmasizadeh, W. Susilo, and A. Majidi, "A secure and efficient authentication technique for vehicular ad-hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 5409–5423, 2018.

[26] J. Zhang, J. Cui, H. Zhong, Z. Chen, and L. Liu, "Pa-crt: Chinese remainder theorem based conditional privacy-preserving authentication scheme in vehicular ad-hoc networks," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[27] D. Chaum and E. Van Heyst, "Group signatures," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1991, pp. 257–265.

[28] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Annual International Cryptology Conference*. Springer, 2004, pp. 41–55.

[29] X. Sun, X. Lin, and P.-H. Ho, "Secure vehicular communications based on group signature and id-based signature scheme," in *2007 IEEE International Conference on Communications*. IEEE, 2007, pp. 1539–1545.

[30] Y. Hao, Y. Cheng, and K. Ren, "Distributed key management with protection against rsu compromise in group signature based vanets," in *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*. IEEE, 2008, pp. 1–5.

[31] A. Wasef and X. Shen, "Efficient group signature scheme supporting batch verification for securing vehicular networks," in *2010 IEEE International Conference on Communications*. IEEE, 2010, pp. 1–5.

[32] J. Shao, X. Lin, R. Lu, and C. Zuo, "A threshold anonymous authentication protocol for vanets," *IEEE Transactions on vehicular technology*, vol. 65, no. 3, pp. 1711–1720, 2015.

[33] J.-L. Huang, L.-Y. Yeh, and H.-Y. Chien, "Abaka: An anonymous batch authenticated and key agreement scheme for value-added services in vehicular ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 1, pp. 248–262, 2010.

[34] S. Taha and X. Shen, "A link-layer authentication and key agreement scheme for mobile public hotspots in nemo based vanet," in *2012 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2012, pp. 1004–1009.

[35] X. Wang, Z. Huang, Q. Wen, and H. Zhang, "An efficient anonymous batch authenticated and key agreement scheme using self-certified public keys in vanets," in *2013 IEEE International Conference of IEEE Region 10 (TENCON 2013)*. IEEE, 2013, pp. 1–4.

[36] Y. Liu, Y. Wang, and G. Chang, "Efficient privacy-preserving dual authentication and key agreement scheme for secure v2v communications in an iov paradigm," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 10, pp. 2740–2749, 2017.

[37] Q. Jiang, J. Ni, J. Ma, L. Yang, and X. Shen, "Integrated authentication and key agreement framework for vehicular cloud computing," *IEEE Network*, vol. 32, no. 3, pp. 28–35, 2018.

[38] L. Dang, J. Xu, X. Cao, H. Li, J. Chen, Y. Zhang, and X. Fu, "Efficient identity-based authenticated key agreement protocol with provable security for vehicular ad hoc networks," *International Journal of Distributed Sensor Networks*, vol. 14, no. 4, p. 1550147718772545, 2018.

[39] L. Liu, Y. Wang, J. Zhang, and Q. Yang, "A secure and efficient group key agreement scheme for vanet," *Sensors*, vol. 19, no. 3, p. 482, 2019.

[40] Q. Li, C.-F. Hsu, K.-K. Raymond Choo, and D. He, "A provably secure and lightweight identity-based two-party authenticated key agreement

protocol for vehicular ad hoc networks," *Security and Communication Networks*, vol. 2019, 2019.

[41] S. A. Eftekhari, M. Nikooghadam, and M. Rafighi, "Security-enhanced three-party pairwise secret key agreement protocol for fog-based vehicular ad-hoc communications," *Vehicular Communications*, p. 100306, 2020.

[42] P. P. Lee, J. C. Lui, and D. K. Yau, "Distributed collaborative key agreement and authentication protocols for dynamic peer groups," *IEEE/ACM Transactions On Networking*, vol. 14, no. 2, pp. 263–276, 2006.

[43] L. Veltri, S. Cirani, S. Busanelli, and G. Ferrari, "A novel batch-based group key management protocol applied to the internet of things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2724–2737, 2013.

[44] C. Xu, R. Lu, H. Wang, L. Zhu, and C. Huang, "Tjet: ternary join-exit-tree based dynamic key management for vehicle platooning," *IEEE Access*, vol. 5, pp. 26 973–26 989, 2017.

[45] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ecdsa)," *International journal of information security*, vol. 1, no. 1, pp. 36–63, 2001.

[46] B. Fan, D. G. Andersen, M. Kaminsky, and M. D. Mitzenmacher, "Cuckoo filter: Practically better than bloom," in *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, 2014, pp. 75–88.

[47] H. Song, S. Dharmapurikar, J. Turner, and J. Lockwood, "Fast hash table lookup using extended bloom filter: an aid to network processing," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 181–192, 2005.

[48] 3GPP, "Security architecture and procedures for 5g system (release 15)," TS 33.501 V15.1.0, 2018.

[49] H. Krawczyk, M. Bellare, and R. Canetti, "Rfc2104: Hmac: Keyed-hashing for message authentication," 1997.

[50] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *International conference on the theory and applications of cryptographic techniques*. Springer, 2000, pp. 139–155.

[51] M. Fersch, E. Kiltz, and B. Poettering, "On the provable security of (ec) dsa signatures," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1651–1662.

[52] 3GPP, "Security aspect for lte support of vehicle-to-everything (v2x) services (release 14)," TS 33.185 V14.0.0, 2017.

[53] "Miracl cryptographic sdk," https://github.com/miracl/MIRACL/, accessed 29 Nov, 2019.

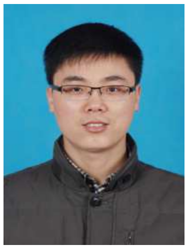
[54] "Crypto++ library 8.2," https://www.cryptopp.com, accessed 29 Nov, 2019.

[55] "Omnet++ discrete event simulator," https://omnetpp.org/, accessed 29 Nov, 2019.

[56] "Simulation of urban mobility," https://sumo.dlr.de/docs/index.html, accessed 29 Nov, 2019.

[57] "The open source vehicular network simulation framework," https://veins.car2x.org, accessed 29 Nov, 2019.

Lu Wei is a PhD student in the School of Computer Science and Technology, Anhui University. His research focuses on security and privacy issues in vehicular ad hoc networks.



Jie Cui was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 90 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Emerging Topics in Computing, IEEE Transactions on Circuits and Systems and IEEE Internet of Things Journal), academic books and international conferences.

Hong Zhong was born in Anhui Province, China, in 1965. She received her PhD degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 120 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Big Data and IEEE Internet of Things Journal), academic books and international conferences.

Yan Xu is currently an associate professor of School of Computer Science and Technology at Anhui University. She received the BS and MS degrees from Shandong University in 2004 and 2007, respectively, and the PhD degree from University of Science and Technology of China in 2015. Her research interests include information security and applied cryptography.

Lu Liu is the Professor of Informatics and Head of School of Informatics in the University of Leicester, UK. He received the Ph.D. degree from University of Surrey, UK and MSc in Data Communication Systems from Brunel University, UK. Prof Lius research interests are in areas of cloud computing, service computing, computer networks and peer-to-peer networking. He is a Fellow of British Computer Society (BCS).