An Efficient Message-Authentication Scheme Based on Edge Computing for Vehicular Ad Hoc Networks

Jie Cui, Lu Wei, Jing Zhang, Yan Xu, and Hong Zhong

Abstract—With the progress in wireless communication technology and the increasing number of vehicles, vehicular ad hoc networks (VANETs) have become essential for improving road conditions and enhancing driving experience. The core of the VANETs is the communication between different vehicles, and the security of the communication is based on message authentication. Several schemes have been designed to enhance the efficiency of message authentication. However, these schemes have the disadvantage of redundant authentication, i.e., repeated authentication of the same message, and fail to seek invalid messages from the batch of messages. To solve these problems, this paper introduces a novel edge-computing concept into the message-authentication process of VANETs. In our scheme, the roadside unit can efficiently authenticate messages from nearby vehicles and broadcast the authentication results to the vehicles within its communication range, thereby reducing redundant authentication and enhancing the efficiency of the entire system. The security analysis results show that the proposed scheme satisfies the security requirements of the VANETs. The performance analysis results show that the proposed scheme can not only work well in an ideal environment where the attacker is absent but also capable of quickly identifying valid and invalid messages even if the VANET is attacked.

Index Terms—VANETs, message authentication, redundant authentication, batch authentication, edge computing.

I. INTRODUCTION

W
ith the development of the automobile industry and the improvement in the economy, vehicles have become increasingly important. However, the increase in the number of vehicles has led to rising traffic congestion and frequent traffic accidents. Therefore, there is a need to improve driving experience and enhance driver safety. This has led to the research of vehicular ad hoc networks (VANETs) with the aim of enhancing driver safety through inter-vehicle communications (V2V) and communications with public infrastructure (V2I) [2]. The typical structure of VANETs comprises three parts: a trusted authority (TA), a roadside unit (RSU), and an on-board unit (OBU). The TA, which acts as the trusted management center, is responsible for the registration and issuing of secret key material. The RSU, installed along the roads, serves as a bridge between the vehicles and the TA. The OBU equipped on each vehicle is in charge of the V2V and V2I communications [3], [4]. As V2V and V2I communications are wireless, malicious attackers can modify the message sent from a vehicle, and even disguise themselves as vehicles if there is no adequate security scheme for the VANETs. Therefore, in VANETs, the message recipient should check the integrity and reliability of the received message. Only if the message is credible, the information contained in the message can be trusted.

In recent years, privacy has become a topic of concern with regard to VANETs. No driver would like to have information, such as driving route or identity, be leaked. Thus, the communication protocol in the VANETs should satisfy anonymity, implying that a vehicle should communicate with all entities via pseudo identity instead of a real one [5]. However, a completely anonymous scheme should be avoided because of the following reasons. Although we cannot avoid the appearance of malicious vehicles that could send forged messages or attempt to modify the valid messages, we can trace malicious vehicles and determine their real identities. In the VANETs, we consider schemes with such capacity to satisfy the conditional privacy-preserving (CPP) characteristic [6], [7].

To solve the security and privacy issues in VANETs, Raya and Hubaux [8] proposed a scheme for signature authentication based on public key infrastructure (PKI). In their scheme, all traffic related information exchanged in VANETs need to be authenticated before trusting the information. In terms of checking integrity and authentication, PKI-based schemes [8]–[10] are well-accepted choices. However, these schemes have the following disadvantages. First, all vehicles need to store many pseudonym certificates, thus linearly increasing the transmission overhead of the RSU with the increase in the number of vehicles. Second, as certificates have a relatively large size, network congestion may occur in the communication channel when the number of vehicles is large. Finally, in their schemes, the RSU and vehicles verify the received messages one after another; this process is very inefficient and unsuitable to be deployed in real scenarios.

To address the performance issues of PKI-based schemes, Zhang et al. [11] introduced an efficient batch message signature verification scheme for V2I communications.
Compared to the RSUs in previous schemes wherein the received messages are verified individually, the RSU in this scheme can simultaneously verify multiple received messages, thus significantly reducing the total verification overhead and enhancing the operational efficiency of the VANETs. Moreover, no certificate is required because this scheme is based on identity, thereby making their scheme quite advantageous with regard to communication and computation overhead. Since the scheme proposed by Zhang [11], many improved identity-based batch authentication schemes have been proposed, such as [11] and [12]. Although these schemes improve the efficiency, they fail to deal with unfavorable conditions, such as when the number of vehicles is extremely large.

The computation power of most vehicles is limited, and multiple messages cannot be verified within the specified time. To solve such problems, Chim et al. [14] proposed a scheme, wherein the RSU helps nearby vehicles to verify their received messages so that the vehicles need not verify them independently. In particular, the RSU verifies multiple messages using the batch authentication technology. If the batch authentication is successful, the messages in the batch are valid. Otherwise, at least one invalid message exists; in which case the batch authentication along with a binary search would be executed to find the invalid messages. After specifying the legitimacy of the messages sent by the vehicles, the RSU would set up two bloom filters to store the authentication results. In particular, the RSU is used to place the hash value of a valid message in a positive filter and the hash value of an invalid message in a negative filter. Thereafter, the RSU would broadcast the positive and negative filters to nearby vehicles at a particular frequency. Hence, the vehicles would only have to check the two filters to verify the messages. This significantly reduces redundant authentication and improves the efficiency of the entire system. However, the complete dependence on the computation performance of the RSU would burden it.

To take full advantages of the computational performance, Liu et al. [15] demonstrated that the computation load on the RSU could be shared by nearby vehicles. In this scheme, the system elects proxy vehicles on the basis of the computation power. The proxy vehicles need to share the work done by the RSU in verifying the messages and send the verification results back to the RSU. The RSU will then check whether the result is accurate. Although the proposed scheme significantly improves the verification performance of the RSU, the performance of the scheme is insufficient because the basic operation comprises bilinear pairing and map-to-point operation with huge overhead. In addition, if a batch of messages contains invalid messages, i.e., the signature of a message is invalid, the RSU fails to confirm whether the original signature is invalid or whether the proxy vehicle tampered with the legitimate signature.

The above schemes using bilinear pairings with a huge overhead would lead to performance issues. To minimize the computation overhead, He et al. [16] proposed an elliptic-curve-cryptography (ECC)-based scheme instead of using bilinear pairings. The computation overhead of the scheme is significantly reduced by discarding the bilinear pairing operation and the hash-to-point function. The literature [17] is improved in this scheme [16], simplifying the signature generation process and improving the efficiency without compromising the security requirement.

In [14], the vehicle transfers the task of authenticating messages to the RSU, and the vehicle itself no longer performs the authentication operation. Essentially, the RSU acts as the cloud for the vehicle, providing message authentication service to the vehicle. In this scenario, the vehicle plays only the role of a data consumer. Although message authentication processed by the RSU can reduce redundant authentication and improve the efficiency of the entire VANETs system, large number of vehicles will lead to a decrease in the computation performance of the RSU and cause significant delay.

Similar to the VANETs, most data generated from the edge of the VANETs (vehicles), and the edge of the data processed by the cloud are inefficient [18]. To solve this problem, edge computing has been introduced. The so-called edge computing is a technique that allows computation at the edge of the network (close to the data source), instead of only at the cloud center [19]. The most important characteristic of edge computing is that the edge ends play the role of data consumers and data producers simultaneously. While in cloud computing, the edge of the network acts only as a consumer.

In this paper, we incorporate a novel edge-computing concept into the message authentication process of the VANETs. In our scheme, the RSU acts as the cloud of the vehicle, and a part of the vehicle acts as an edge-computing node to assist the RSU with the message authentication task. In particular, the system elects a number of edge computing vehicles (ECVs) to assist the RSU in authenticating the message signature sent by nearby vehicles and subsequently sending the results to the RSU on the basis of the available computation power of the vehicle. The RSU verifies the results sent from the ECVs, obtains the legitimacy of these messages, and finally, broadcasts the information regarding the legitimacy to the vehicle through a cuckoo filter. Consequently, the vehicle needs to only query the filter to determine whether the message is valid and need not verify the received messages independently in most cases. The main contributions of this study are as follows:

1) We introduce the concept of edge computing into the VANETs by selecting sufficient number of ECVs to share the load on the RSU in authenticating the messages. In addition, this paper presents an efficient authentication mechanism that allows the RSU to quickly verify the feedback given by the ECVs.

2) We centralize the authentication tasks of all vehicles on the RSU and ECVs and broadcast the results through the cuckoo filter, thus significantly reducing the redundant authentication of the same messages.

II. BACKGROUND

A. Network Model

As shown in Fig.1, the VANETs model used in our paper includes three parts: trusted authority (TA), roadside unit (RSU), and the vehicle equipped with an onboard unit (OBU), wherein the vehicle participating in the message authentication is referred to as the edge computing vehicle (ECV).
achieve the following security objectives:

- **TA**: The TA, as the registry center of RSUs and vehicles, is trusted by all entities in the VANETs and responsible for distributing key materials to all entities. And the TA has the ability to trace the real identity of the vehicle if necessary. For secure communication, a wired secure transport protocol such as Transport Layer Security (TLS) is used between TA and RSU. In order to avoid the formation of performance bottlenecks and improve reliability, redundant TA is usually set.

- **RSU**: The RSU is located on the roadside and can communicate directly with the vehicle as a bridge between the TA and vehicles. The RSU can authenticate its received messages, and if necessary, process the results locally, or send the results to the traffic management center for data analysis [16]. In order to ensure that RSU can help the vehicle to carry out message authentication, RSU’s computation performance should be far greater than the vehicle [20].

- **Vehicle**: The OBU mounted on the vehicle periodically broadcast the traffic-related information to improve the operation efficiency of regional traffic and traffic security. Each vehicle has a tamper-proof device (TPD) to store received key material from the TA securely and we assume TPD unhackable [3]. In this paper, vehicles are divided into two types: ordinary vehicles (OVs) and ECVs. Ordinary vehicles only play the role of data processors and consumers at the same time. The ECVs have obligations to help the RSU to authenticate the messages they receive, but also have the right to enjoy the convenience from the RSU’s broadcast. The model diagram is shown in Fig. 1.

**B. Security Objectives**

A well-designed message authentication scheme should achieve the following security objectives:

1) **Message Authentication and integrity**: After receiving a message, the recipient of the message must first determine the legitimacy of the message owner, and whether the message itself is tampered with or whether it is forged.

2) **Identity Privacy Preserving**: The real identity of the vehicle should remain anonymous, which means that the vehicle should use the pseudo identity when sending traffic-related information. No third party except for TA can extract the real identity of the vehicle based on multiple messages sent from the same vehicle.

3) **Traceability**: TA can trace the real identity of the vehicle by analyzing its pseudo identity extracted from its message.

4) **Replay Attack**: A malicious vehicle cannot collect and store a signed message and attempt to deliver it later when the original message expired.

**C. Elliptic Curve Cryptosystem**

In 1984, Miller [20] applied the elliptic curve to cryptography for the first time. After Koblitz [21] built the elliptic curve cryptography (ECC) with elliptic curve discrete logarithm problem (ECDLP), ECC began to be widely applied to encryption, protocol and other safety-related areas. Let \( F_p \) be a finite field, which is determined by a prime number \( p \). Let a set of elliptic curve point \( E \) be a finite field, which is determined by a prime number \( p \), and forms an Abelian group. And the TA has the ability to trace the real identity of the vehicle [20] and vehicles, is trusted by all entities in the VANETs and responsible for distributing key materials to all entities.

**D. Cuckoo Filter**

For massive data, we need an index data structure to help with the query operation, to quickly determine the existence of data records. And the most frequent choice is Bloom filter and its extended version. However, Bloom filter has high false positive and low space usage rate. Besides, a limitation of Bloom filter is that one cannot remove existing items without rebuilding the entire filter. Cuckoo filter is a new data structure for set-membership tests, which has lower false positive, lower space overhead and better performance than traditional Bloom filter. And cuckoo filter supports adding and removing items dynamically while achieving high performance [23]. A cuckoo filter is essentially a hash table consisting of a set of buckets,
and each bucket contains a fixed number of fingerprints (the hash function with fewer output bits), as shown in Fig. 2(a).

The cuckoo filter algorithm includes three functions: query, insert and delete. The query function first calculates the fingerprint of the query item and gets the location of the query item in the corresponding hash table according to the fingerprint. If the location is found, the query is successful, otherwise return false. The insert function calculates the fingerprint of the inserted item first, and gets the position of the inserted item in the corresponding hash table according to the fingerprint. If the location is not occupied, insert directly into the inserted item, and insert the removal item into an empty position, as shown in Fig. 2(b). The delete function first query the hash table, if the query is successful, regardless of the query to the bucket in the number of fingerprints meet the requirements, only delete one fingerprint.

E. Fuzzy Logic Control System

A fuzzy control system is a control system based on fuzzy logic mathematical system that analyzes analog input values in terms of logical variables that take on continuous values between 0 and 1, in contrast to classical or digital logic, which operates on discrete values of either 1 or 0 (true or false, respectively). Fuzzy logic is widely used in machine control. The term fuzzy refers to the fact that the logic involved can deal with concepts that cannot be expressed as the true or false but rather as partially true. Fuzzy logic has the advantage that the solution to the problem can be cast in terms that human operators can understand, so that their experience can be used in the design of the controller. This makes it easier to mechanize tasks that are already successfully performed by humans [24], [25]. A typical fuzzy logic control system mainly contains the following three steps:

- **Fuzzification** Fuzzification refers to the process which convert the crisp input value to the fuzzy value.
- **Fuzzy rules defining** The fuzzy rules are several If-Then statements which input several fuzzy values and output a fuzzy value.
- **Defuzzification** Defuzzification is responsible for choose an appropriate representative value as a final output which is a crisp value. The most common defuzzification method is center of gravity(COG) method.

III. OUR PROPOSED SCHEME

In this section, we present our proposed scheme. The definition of notations in our scheme is presented in Table I.

A. System Initialization Phase

At this phase, the TA generates the necessary system parameters and then the TA preloads these system parameters into all vehicles’ TPD and all RSUs’ memory. Because the RSU connect with the TA via secure wired network, the parameter transmission can be processed anytime. And the vehicle can get the parameters in some special situation like ETC gate or vehicle inspection under pre-store strategy [3]. Specific steps in system initialization phase are as follows:

1) TA randomly selects two large prime \( p, q \), and a non-singular elliptic curve \( E \) where is defined as \( y^2 = x^3 + ax + b \mod q \), and the generator element \( P \) is randomly selected in the group.
2) TA randomly selects \( s \in Z^*_q \) as the system private key, and calculates the system public key \( P_{pub} = sP \).
3) TA randomly selects \( x \in Z^*_q \) as the private key of RSU, and calculates the RSU’s public key \( PK_R = xP \).
4) TA chooses the secure hash function \( h : \{0, 1\}^* \rightarrow Z_q \).
5) TA assigns the real identity \( RID \) and password \( PWD \) to each vehicle, and preloads \( \{RID, PWD, s\} \) to the TPD of the vehicle.
6) TA sends the private key \( x \) to the RSU.
7) TA publishes system public parameters \( \{p, q, a, b, P, P_{pub}, PK_R, h\} \) to the RSU and all vehicles.

B. The Generation Phase of Vehicles’ Pseudo Identity and Signature

Before sending a message, the vehicle must provide a signature of the message in order to ensure the authenticity. The following work is completed by the TPD, used to generate the vehicle’s pseudo identity, public key and signature.

1) The vehicle sends its own real identity \( RID \) and password \( PWD \) to the TPD for identity check. If the two values are equal to the pre-stored values in the TPD, the vehicle passes the authentication and proceeds to the following steps. Otherwise, the authentication fails and the service is rejected.
2) TPD randomly selects a number \( r_i \in Z^*_q \), and calculates the pseudo identity \( PID_i = \{PID_1^i, PID_2^i\} \), where \( PID_1^i = r_i \cdot P, PID_2^i = RID \oplus h (r_i \cdot P_{pub}) \).
3) The vehicle gets message signature by combining message \( M \) and current timestamp \( T \), and TPD inputs

![Fig. 2. (a) The structure of Cuckoo filter; (b) The insertion operation.](image)

<table>
<thead>
<tr>
<th>Table I: Definition of Notations in Our Proposed Scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td>Notation</td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>( PID_e )</td>
</tr>
<tr>
<td>( RID_e )</td>
</tr>
<tr>
<td>( h(\cdot) )</td>
</tr>
<tr>
<td>( PK_e )</td>
</tr>
<tr>
<td>( SE )</td>
</tr>
<tr>
<td>( SIG(\cdot) )</td>
</tr>
<tr>
<td>( \oplus )</td>
</tr>
<tr>
<td>( \parallel )</td>
</tr>
<tr>
<td>( fingerprint(\cdot) )</td>
</tr>
<tr>
<td>( T_{op} )</td>
</tr>
</tbody>
</table>
CUI et al.: EFFICIENT MESSAGE-AUTHENTICATION SCHEME BASED ON EDGE COMPUTING

4) The vehicle broadcasts \{PID_i, M_i, \sigma_i\}.

C. The Election Strategy of ECV

In order to make the possibility of success that ECV help the RSU authenticate message signatures as much as possible, we choose the qualified general vehicles which contain two characters as ECVs: short distance to the RSU and enough available computation resource. We assume all vehicles exchange position and available computation resource information through BSM (Basic Safety Message) and we only consider the two metrics to select ECVs. However, the two metrics are always inaccurate. For example, the vehicle which is close to the RSU may have very little available computation resource. Besides, the information (position and available computation resource) is always inaccurate. It would be very costly if we use traditional mathematical model to solve the issue. Luckily, fuzzy logic can solve the uncertain and inaccurate information.

Fuzzification: We use the following membership function to convert distance and available performance to fuzzy values.

1) Distance Membership Function: Because the ECVs share the RSU’s authentication task, the distance between the ECVs and the RSU should be relatively short which can decrease the transmission delay and guarantee relatively better communication quality. The RSU can calculate the distance according the GPS positions of itself and the vehicle V_x. Let \(d(x)\) denotes the distance between the vehicle \(V_x\) and the RSU, \(R\) denotes the max transmission range of the RSU. Then we can define distance metric membership function as Equation (1). And Fig. 3 shows the membership function of distance metric.

\[
DM(x) = \begin{cases} 
1, & d(x) \leq \frac{R}{2} \\
\frac{R - d(x)}{R}, & \frac{R}{2} < d(x) \leq R \\
0, & d(x) > R 
\end{cases}
\] (1)

2) Available Performance Metric Membership Function: Message authentication requires enough computation resource, so we need the vehicle which has enough available computation resource to be ECV. We assume all vehicles have the similar computation loads. Let \(UCR(x)\) denote used computation resource of vehicle \(V_x\), \(MCL(x)\) denote maximum computation load of vehicle \(V_x\). Then we can define available performance membership function as Equation (2). And Fig. 4 shows the membership function of available performance metric.

\[
APM(x) = \frac{MCL(x) - UCR(x)}{MCL(x)}
\] (2)

Fuzzy Rules: Here we map previous defined fuzzy values to the following IF/THEN rules by using Min-Max method. And the fuzzy rules are as shown in Table II.

Defuzzification: Based on the output membership function and corresponding membership degrees, we can use COG (Center of Gravity) method which is shown in Equation (3), to defuzzify the fuzzy results. And the output membership function is shown in Fig 5. In our scheme, the COG result (qualified degree to be ECV) represents the fitness of the vehicle \(V_i\) to be the ECV. If \(QD \geq k\) (\(k\) is positively correlated with with the traffic density. If the number of vehicle in the proximity of RSU is relatively large, the computing load of the ECV will be much higher, and the number of malicious attacker will also be larger in probability that requires the ECV to have better computation performance and be in better position between other vehicles and RSU.),

\[
QD = \frac{MCL(x) - UCR(x)}{MCL(x)}
\] (3)
it indicates that $V_i$ is suitable to be ECV and the RSU would elect $V_i$ to be ECV.

$$QD = \int \mu(x)dx \int \mu(x)dx \quad (3)$$

### D. The Batch Authentication of ECV

In this phase, the ECV needs to go through the following two stages from the beginning to the end of the task: task determination, batch authentication, and result feedback.

1) **Task Determination Phase**: In this phase, the RSU allocates the pseudo identity list $PID = \{PID_1, PID_2, \cdots, PID_n\}$ of a part of its received message owners to $ECV_i$. The RSU then sends the message $\{PID, SIG_{SK_{RSU}}(PID)\}$ to $ECV_i$.

2) **Batch Authentication and Results Feedback Stage**: After receiving messages $\{PID, SIG_{SK_{RSU}}(PID)\}$, $ECV_i$ verifies it by using the public key $PK_{RSU}$ of the RSU. If it is not legal, rejects the message, otherwise, processes to the next step. For the received $PID$ list as $\{PID_1, PID_2, \cdots, PID_n\}$, $ECV_i$ searches the corresponding message signature pair $\{PID, M_i, \sigma_i\}$($1 \leq i \leq n$) which it needs to authenticate in its own message cache, then proceed as follows:

1) $ECV_i$ checks the freshness of these messages $ECV_i(1 \leq i \leq n)$. If there is an expired message, the RSU sends a report to the RSU. And the RSU will remove the corresponding pseudo identity of the expired message from the corresponding message owner list $PID$.

2) $ECV_i$ randomly selects a small integer $t$ of 10 bits length, generates the random small factor $v_i$($v_i(1 \leq i \leq n) \in [1, 2^t]$), and then verifies whether the batch authentication Equation (4) holds or not. If the Equation (4) is established, it indicates that the batch of message passes the check. Otherwise, it means that the message contains at least one invalid message, i.e., the message fail to pass one of the checks: the integrity of message, the legitimacy of message signature.

$$\left( \sum_{i=1}^{n} v_i \cdot \sigma_i \right) P = \left( \sum_{i=1}^{n} v_i \cdot h(PID_i) \right) \cdot P_{pub} + \sum_{i=1}^{n} v_i \cdot PID_i \cdot h(M_i) \quad (4)$$

3) In order to be able to quickly find the invalid messages contained in the batch of messages, we set a list as $List = \{M_1, M_2, \cdots, M_n\}$ for storing messages to be processed, and a empty list $List_1$ for storing invalid messages. And $ECV_i$ executes $batchSearch(List, List_1, low, high)$ in which $batchSearch(List, List_1, low, high)$ is defined in Algorithm 1.

4) Through the above steps, $ECV_i$ can get the invalid message list $List_1$ which can be used to determine the corresponding identity list $InvalidList_{ECV_i} = \{PID_{1}, PID_{2}, \cdots, PID_{n}\}$ in the message buffer pool. And then $ECV_i$ feeds back the authentication results $InvalidList_{ECV_i}$ to RSU. It is worth noting that, in order to save network bandwidth, $ECV_i$ only needs to send invalid message owner’s pseudo identity because most of the message signatures are valid in VANETs, and the RSU can extract the identities of valid messages by executing $List - List_1$ operation.

The validity of Equation (4) can be verified as follows:

$$\left( \sum_{i=1}^{n} v_i \cdot \sigma_i \right) P = \left( \sum_{i=1}^{n} v_i \cdot (s \cdot h(PID_i) + r_i \cdot h(M_i)) \right) P$$

$$= \left( \sum_{i=1}^{n} v_i \cdot s \cdot h(PID_i) \right) P + \sum_{i=1}^{n} v_i \cdot r_i \cdot h(M_i) \cdot P$$

$$= \sum_{i=1}^{n} v_i \cdot h(PID_i) \cdot P_{pub} + \sum_{i=1}^{n} v_i \cdot PID_i \cdot h(M_i) \quad (5)$$

### E. The RSU Checks the Authentication Result of ECV

In the real VANETs environments, packet loss and delay always exist. If the RSU doesn’t receive the authentication result from $ECV_i$ with the $\Delta T$, it can assign the authentication task to another $ECV$. Otherwise, the RSU first checks whether
the signature of the message is valid or not, if invalid, then
rejects the message, if valid, follows these steps:

1) The RSU searches for pseudo identity list \( \text{List}_{ECV_i} = \{PID_1, PID_2, \ldots, PID_n\} \) which are verified by \( ECV_i \) from its memory. Afterwards, the RSU gets pseudo identity list which belongs to invalid messages by executing \( \text{ValidList}_{ECV_i} = \text{List}_{ECV_i} - \text{InvalidList}_{ECV_i} \).

2) For the message signatures corresponding to all elements in \( \text{ValidList}_{ECV_i} \), the batch authentication is carried out as shown in Equation (4); if passed, output \( a = \text{True} \); if not, output \( a = \text{False} \). For the message signatures corresponding to all elements in \( \text{InvalidList}_{ECV_i} \), the single authentication is carried out as shown in the follow Equation (6); if none of those authentication passed, output \( b = \text{True} \); if not, output \( b = \text{False} \). Through the two checks, the \( ECV_i \) can not deceive the RSU by claiming that the invalid messages are valid or the valid messages are invalid.

\[
\sigma_i \cdot P = h (PID_i) \cdot P_{pub} + h (M_i) \cdot PID_i^1 \tag{6}
\]

3) If \( a = \text{True} \land b = \text{True} \), then message authentication result of \( ECV_i \) is reliable, and the RSU will send \( PID_{ECV_i} \) to the TA to increase the credit of \( ECV_i \), and then go to step 4); otherwise, it indicates that \( ECV_i \) tries to deceive RSU and RSU would identify \( ECV_i \) as a malicious vehicle, cancel the ECV qualification of \( ECV_i \), and send \( PID_{ECV_i} \) to TA in order to revoke \( ECV_i \) and decrease the credit of \( ECV_i \).

4) After step 3), the RSU can confirm that the authentication results from \( ECV_i \) are reliable. The RSU initializes two cuckoo filters, i.e., the positive filter \( \text{posFilter} \) and the negative filter \( \text{negFilter} \). The RSU calculates the fingerprint \( \text{fingerprint}(M_i) \) corresponding to every element in \( \text{ValidList}_{ECV_i} \) and puts the fingerprint into the positive filter \( \text{posFilter} \). For every pseudo identity in \( \text{InvalidList}_{ECV_i} \), the RSU calculates its fingerprint and puts it into the negative filter \( \text{negFilter} \). After initialization, the RSU broadcasts the positive filter \( \text{posFilter} \), the negative filter \( \text{negFilter} \), and the corresponding signature \( \text{SIG}_{SK_{RSU}}(\text{posFilter}, \text{negFilter}) \) to the nearby vehicles in the area.

5) If the vehicle density near the RSU is high, the size of cuckoo filters would be slightly large which could cause high packet loss rate and frequent wireless channel contention and retransmission. One efficient solution is to cut the filters into slices and distribute the slices to nearby ECVs for relaying these slices to nearby ordinary vehicles by ECVs. The detail steps are as shown below. First, the RSU cuts the positive filter and the negative filter into slice sets \( \{PF_1, PF_2, \ldots, PF_m\} \) and \( \{NF_1, NF_2, \ldots, NF_n\} \) respectively. Second, the RSU sends these slices to nearby ECVs. Last, the RSU calculates \( \sigma_{filters} = \text{SIG}_{SK_{RSU}}(h(PF_1, PF_2, \ldots, PF_m, NF_1, NF_2, \ldots, NF_n)) \) using ECDSA signing algorithm and broadcasts the signature to nearby vehicles.

<table>
<thead>
<tr>
<th>Case</th>
<th>The query result of ( \text{posFilter} )</th>
<th>The query result of ( \text{negFilter} )</th>
<th>Conclusion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Success</td>
<td>Fail</td>
<td>Valid message</td>
</tr>
<tr>
<td>2</td>
<td>Fail</td>
<td>Success</td>
<td>Invalid message</td>
</tr>
<tr>
<td>3</td>
<td>Fail</td>
<td>Fail</td>
<td>Failed query</td>
</tr>
<tr>
<td>4</td>
<td>Success</td>
<td>Success</td>
<td>Failed query</td>
</tr>
</tbody>
</table>

F. The Message Authentication of the Ordinary Vehicles

In our scheme, the vehicle no longer needs to independently verify the messages sent by other vehicles in almost all cases, because the two filters already contain the message signature legitimacy information of nearby vehicles. If the vehicle would like to verify one message valid or not, it just need to process the following steps. First, the vehicle extracts the slices of the filters from nearby ECVs and check the validity of extracted slices by using ECDSA verifying algorithm. Second, the vehicle calculates the corresponding fingerprint \( f = \text{fingerprint}(M_i) \) of message \( M_i \) that needs to be verified, respectively query whether the \( f \) value can be queried successfully in the positive filter \( \text{posFilter} \) and negative filter \( \text{negFilter} \), and determine whether the message valid or not according to the Table III.

If the query result satisfies case 1 or case 2, the conclusion will be definite. However, there is a certain probability that case 3 or case 4 will happen. If case 3 happens, it indicate that the RSU has not yet been updated the authentication result of the query message. The vehicle can wait for the next update. Case 4 may happen because of the potential false positive of cuckoo filter. The next section would indicate that the probability of case 4 is very small.

IV. SECURITY PROOF AND ANALYSIS

A. Security Proof

In this section, we first show that our proposed signature scheme is secure against an alternatively chosen message attack under the random oracle model.

1) Forking Lemma: Let \( \mathcal{A} \) be a probabilistic polynomial time Turing machine whose input only consists of public data. We let \( Q \) denote the max number of queries in which \( \mathcal{A} \) can ask the random oracle, and let \( R \) denote the max number of queries in which \( \mathcal{A} \) can ask the signer. And we also assume that \( \mathcal{A} \) can produce a valid signature \( \{m, \sigma_1, h, \sigma_2\} \). If the triples \( \{\sigma_1, h, \sigma_2\} \) can be simulated without knowing the secret key, with an indistinguishable distribution probability, then there is another machine that has control over the machine obtained from \( \mathcal{A} \) replacing interaction with the signer by simulation and produces two valid signatures \( \{m, \sigma_1, h, \sigma_2\} \) and \( \{\sigma_1, h, \sigma_2\} \) such as \( h \neq h' \) in expected time \( T' \geq 120686QT/e \).

Let \( \mathcal{A} \) be an adversary who performs an existential forgery under an alternatively chosen attack against our proposed scheme within a time bound \( T \) and with a probability of \( \epsilon \), and we can reach the following conclusion.
Theorem 1: Our proposed scheme can resist chosen message attack under the random oracle model.

Proof: Suppose that an ECDLP instance \((P, Q = xP)\) is given, where \(P, Q\) are two points on \(E\) and there is an adversary \(A\) that could forge message \(\{PID_i, M_i, \sigma_i\}\). We also set a challenger \(C\) which is able to solve the ECDLP by running \(A\) as a subroutine with a probability that cannot be ignored.

2) Setup: The Setup algorithm takes a secure parameter \(n\) as input, and then \(C\) chooses a random number \(s\) as system secret key and computes its corresponding public key by \(P_{pub} = sP\). Next, \(C\) generates public parameters \(\text{Params} = \{P, q, G, E, P_{pub}, h_1, h_2\}\) and publishes it.

3) \(H_1\) Hash Query: When \(A\) makes an \(H_1\) query with message \(M_i\), \(C\) checks whether the tuple \((M_i, \tau_{H_i})\) is in the list \(L_{H_i}\) or not. If yes, \(C\) sends \(\tau_{H_i} = H_1(M_i)\) to \(A\). Otherwise, \(C\) chooses a random \(\tau_{H_i} \in Z_q^*, \text{adds} \{(M_i, \tau_{H_i})\}\) into the list \(L_{H_i}\), and then sends \(\tau_{H_i} = H_1(M_i)\) to \(A\).

4) \(H_2\) Hash Query: When \(A\) makes an \(H_2\) query with message \(M_i\), \(C\) checks whether the tuple \(\sigma, h_{i,1}, h_{i,2} \in Z_q^*\) is already in the list \(L_{H_i}\) or not. If yes, \(C\) sends \(\tau_{H_i} = H_2(PID_i, PID^2_i)\) to \(A\). Otherwise, \(C\) chooses a random \(\tau_{H_i} \in Z_q^*, \text{adds} \{(PID^1_i, PID^2_i, \tau_{H_i})\}\) into the list \(L_{H_i}\), and then sends \(\tau_{H_i} = H_2(PID_i, PID^2_i)\) to \(A\).

5) Sign Query: When \(A\) makes a query on message \(M_i\), \(C\) generates three random numbers \(\sigma, h_{i,1}, h_{i,2} \in Z_q^*\) and calculates \(PID^1_i = (\sigma P - h_{i,1} P_{pub})/h_{i,2}\). And then \(C\) chooses a random point \(PID^2_i \in G\), and then adds the tuple \((M_i, h_{i,1})\) into the list \(L_{H_i}\) and the tuple \((PID^1_i, PID^2_i, h_{i,2})\) into the list \(L_{H_2}\).

By using Forking Lemma [26], after replaying \(A\) with the same random element, \(C\) achieves two valid signatures \(\sigma = r_i h_{i,1} + s h_{i,2}\) and \(\sigma' = r_i h_{i,1}' + s h_{i,2}'\) within polynomial time, where \(C\) can successfully achieve the value \(x\) by computing:
\[
\frac{h_{i,2}' h_{i,1} - h_{i,1}' h_{i,2}}{h_{i,2}} \bmod q = s
\]
As a result, \(C\) can break the ECDLP within expected time less than \(120686Q\sqrt{q}/\epsilon\), if \(\epsilon \geq 10(R + 1)(R + Q)/q\). However, it contradicts with the difficulty of solving the ECDLP. Therefore, the signature scheme in our scheme is secure against forgery under the alternatively chosen message attack in the random oracle model.

B. Security Analysis

In this section, we show that our proposed scheme satisfies several security requirements.

1) Message Authentication: In this scheme, the signature of the message guarantees the integrity of the message and the legitimacy of the message owner. Theorem 1 has proved that the signature used in our scheme cannot be forged under the random oracle model. Therefore, malicious attackers cannot forge valid signatures. In theorem 2, we show that ECV cannot cheat the RSU successfully.

Theorem 2: ECV cannot successfully fake valid message signature authentication results in order to cheat the RSU.

Proof: Let \(T_{TaskList_{ECV}}\) denote the message signature list which is verified by \(ECV_i\), and let \(InvalidList_{TaskList_{ECV}}\) denote the invalid signature list which is the output of \(ECV_i\). The RSU can get the valid signature list by executing \(ValidList_{ECV} = TaskList_{ECV} - InvalidList_{ECV}\). For all signatures in list \(ValidList_{ECV}\), the RSU performs batch authentication as described in Equation (4). Obviously, if the batch authentication is passed, it would imply that the RSU is not deceived by \(ECV_i\). For all signatures in \(ValidList_{ECV}\), the RSU performs single authentication as described in Equation (5) one by one. If \(ECV_i\) had not deceived RSU, then none of these single authentication would pass, otherwise, at least one single authentication would get passed. Because the RSU checks the lists \(ValidList_{ECV}\) and \(InvalidList_{ECV}\) at the same time, \(ECV_i\) cannot deceive the RSU successfully.

2) Identity Conditional Privacy Preserving: In the V2V communication process, the vehicle does not use its own real identity, but use its pseudo identity \(PID_i = \{PID^1_i, PID^2_i\}\) where \(PID^1_i = r_i \cdot P\) and \(PID^2_i = RID \oplus h(r_i \cdot P_{pub})\). According to the discrete logarithm problem (DLP), other vehicles cannot get the private key of the vehicle through \(PID^1_i\) and \(P\). At the same time, the key material is stored in the TPD of the vehicle which is unable to be cracked, so even the vehicle itself cannot leak any information stored in the TPD. Thus, for any vehicle other than the owner of a pseudo identity, the real identity which is corresponding to the pseudo identity cannot be obtained on the basis of the public pseudo identities and known public information. In order to be able to investigate the responsibility of a malicious vehicle, the TA needs to trace the real identity of the vehicle. If the TA needs to trace the real identity of the vehicle, it can get the real identity through using the corresponding pseudo identity \(PID_i = \{PID^1_i, PID^2_i\}\) according to the following equation.
\[
PID^2_i \oplus h(s \cdot PID^1_i) = RID \oplus h(r_i \cdot P_{pub}) \oplus h(s \cdot r_i \cdot P) = RID \oplus h(r_i \cdot P_{pub}) \oplus h(r_i \cdot P_{pub}) = RID
\]

3) Replay Attack Resistance: In order to avoid outdated message, the message receiver first checks whether the message is expired or not. Let \(T_g\) indicate the receiving time of the message, \(T\) indicate the message departure time, \(\Delta t_1\) indicate the time difference between the vehicle’s clock and the system clock, and \(\Delta t_2\) indicates the network delay. If \(|T - T_g| < \Delta t_1 + \Delta t_2\), then the receiver accepts the message, otherwise, it indicates that the message is expired and the recipient would reject the message.

4) Unlinkability: As shown in subsection B of section III, the pseudo identity which the vehicle uses contains the random number \(r_i\). The message signature uses a random number and the system private key which is safety saved in the unhackable TP. Therefore, there is no malicious attacker can link the two pseudo identities or two signatures generated by the same vehicle.

V. PERFORMANCE ANALYSIS

A. Computation Overhead Analysis

In this subsection, we compare the computation overhead of our proposed scheme with three recent schemes that consist
of two schemes based on bilinear pairing and one ECC-based scheme.

For ID-based schemes using bilinear pairings, we use a bilinear pairing \( \epsilon : G_1 \times G_1 \to G_2 \) to achieve the security level of 80 bits, where \( G_1 \) is an additive group generated by a point \( \hat{P} \) with the order \( q \) on the super singular elliptic curve \( E : y^2 = x^3 + x \mod \hat{p} \) with embedding degree 2, where \( \hat{p} \) is a 512 bit prime number and \( q \) is a 160 bit Solinas prime number. For ID-based schemes using the ECC, we construct an additive group generated by a point \( P \) on a non-singular elliptic curve \( E : y^2 = x^3 + ax + b \mod p \), and its order is \( q \), where \( p,q \) are two 160 bit prime numbers, and \( a, b \in \mathbb{Z}_p^* \). On the platform of 3.4GHZ i7-4770 [16], we can get the basic cryptographic operation execution time by using MIRACL library [27] as shown in Table IV.

In our scheme, the RSU can quickly find the invalid message signatures according to the authentication result sent from ECVs. In the following part, we consider the RSU’s authentication overhead and the RSU’s authentication overhead plus authentication overhead from nearby vehicles respectively. In ideal situations, there is no malicious attacker in VANETs; while in some situations, the malicious attacker exists. So we also consider the two kind of situations respectively.

First, we only consider the computation overhead of the RSU in ideal situation. For the batch authentication phase of Chim et al.’s scheme [14], the RSU needs to execute two bilinear pairing operations, 2n scalar multiplication operations related to the bilinear pairing, and n hash-to-point operations related to the bilinear pairing. Therefore, the execution time of the phase is \( 2T_{bp} + 2nT_{sm-bp} + nT_{mp} = 7.824n + 8.422 \text{ms} \). In the batch authentication phase of He et al.’s scheme [16], the RSU needs to execute n+2 scalar multiplication operations related to the ECC and 2n small scalar multiplication operations related to the ECC. Therefore, the execution time of the phase is \( (n + 2)T_{sm-ecc} + 2nT_{sm-ecc-s} = 0.4972n^2 + 0.884n + 0.884 \text{ms} \).

For the batch authentication phase of Zhang et al.’s scheme [12], the RSU needs to execute 3 bilinear pairing operations, \( n+1 \) scalar multiplication operations related to the bilinear pairing. Therefore, the execution time of the phase is \( 3T_{bp} + (n + 1)T_{sm-bp} = 1.709n + 14.342 \text{ms} \). For the batch authentication phase of our proposed scheme, the RSU needs to execute \( n+2 \) scalar multiplication operations related to the ECC and \( n \) small scalar multiplication operations related to the ECC. Therefore, the execution time of the phase is \( (n+2)T_{sm-ecc} + nT_{sm-ecc-s} = 0.4696n + 0.884 + 0.4696(n-1) = 0.9392n + 0.4144 \text{ms} \).

- **Table IV**

<table>
<thead>
<tr>
<th>Cryptographic operation</th>
<th>Symbol</th>
<th>Execution time/ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>The bilinear pairing operation, i.e., ( \epsilon(P, Q) )</td>
<td>( T_{bp} )</td>
<td>4.211</td>
</tr>
<tr>
<td>The scale multiplication operation ( x \cdot P ) of bilinear pairing</td>
<td>( T_{sm-bp} )</td>
<td>1.709</td>
</tr>
<tr>
<td>The scale multiplication operation ( x \cdot P ) of ECC</td>
<td>( T_{sm-ecc} )</td>
<td>0.442</td>
</tr>
<tr>
<td>The small scale multiplication of ( v_i \cdot P ) of ECC where ( v_i \in [1, 2^l] )</td>
<td>( T_{sm-ecc-s} )</td>
<td>0.0276</td>
</tr>
<tr>
<td>The hash-to-point function</td>
<td>( T_{mp} )</td>
<td>4.406</td>
</tr>
</tbody>
</table>

**Fig. 6.** The relationship between the overhead of RSU and the number of messages.
data, we can get the relationship between the total overhead of the message authentication and the number of vehicles in one RSU’s domain when all of batch authentication pass successfully, as shown in Fig. 7.

It can be seen from Fig. 7 that the efficiency of our scheme is significantly better than that of other schemes when the system does not have any invalid signature.

In the real VANETs scenario, there are often malicious attackers which may produce invalid signatures. In this case, it would be very time-consuming if the RSU verifies the message signatures independently because the batch authentication have a high probability to fail. In our proposed scheme, the task to find invalid signatures is outsourced to ECVs, and the RSU just needs to check the results from ECVs. Let n indicate the number of messages in one RSU’s domain, and ε indicate the percentage of invalid signatures. When there is at least one invalid signature, the overhead of the RSU is $T_{RSU} = n \epsilon \cdot 1.326 + 0.4696(n - n \epsilon) + 0.884 = 0.8564n \epsilon + 0.4696n + 0.884$ ms. Fig. 8 shows the relationship between the overhead of the RSU and the number of messages, and the percentage of invalid message signatures in this situation.

As is shown in Fig. 8, even the invalid message signature ratio is as high as 0.1 (the literature [29] pointed out that the attacker can compromise at most 10% vehicles), the RSU can verify more than 500 messages which contain 10% invalid messages within the message update period of the vehicle, i.e., 300ms. While the other scheme [14] do not employ the idea of edge computing, the binary search method combined with batch authentication is executed on RSU’s side. If the ratio of the invalid signatures is slightly larger, the overhead of the RSU would be very large and we can see the phenomenon from Fig. 9. Comparing Fig. 8 with Fig. 9, we can draw a conclusion that the RSU in our proposed scheme has a significantly lower authentication overhead.

In terms of the vehicle, the general vehicle is same with that in Chim et al.’s scheme except in case 4, that is, there is no message authentication overhead. And we will analyse case 4 in the follow subsection.

### B. Communication Cost Analysis

In this subsection, we analyze the communication cost of relevant ID-based schemes for VANETs. Since the sizes of $\tilde{p}$ and $p$ are 64 bytes and 20 bytes respectively, the sizes of the elements in group $G_1$ and $G$ are 128 bytes and 40 bytes respectively. Besides, let the sizes of the hash function’s output and timestamp be 20 bytes and 4 bytes respectively. The comparison of computation costs is presented in Table V.

In Chim et al.’s scheme [13], the vehicle broadcasts the pseudo identity and signatures $\{PID_i, M_i, \sigma_i\}$ to the verifier, where $PID_i = \{PID_i^1, PID_i^2\}$, $PID_i^1, PID_i^2, \sigma_i \in G_1$. Therefore, the communication cost of their scheme is $128 \times 3 = 384$ bytes. In He et al.’s scheme [15], the vehicle broadcasts the pseudo identity and signatures $\{PID_i, R_i, \sigma_i, T_i\}$ where $PID_i = \{PID_i^1, PID_i^2\}$, $PID_i^1, PID_i^2, R_i \in G_1$. Therefore, the communication cost of their scheme is $40 \times 3 + 20 + 4 = 144$ bytes. In Zhang et al.’s scheme [11], the vehicle broadcasts the pseudo identity and signatures $\{PID_i, \sigma_i, T_i\}$ where $PID_i = \{PID_i^1, PID_i^2\}$, $PID_i^1, PID_i^2, \sigma_i \in G_1$. Therefore, the communication cost of their scheme is $128 \times 3 + 4 = 388$ bytes. In our proposed scheme, the vehicle broadcasts the pseudo identity and signatures $\{PID_i, \sigma_i, T_i\}$, where
We set \( b \) fingerprint size of the item, \( R_i \in G, \sigma_i, \) \( PID_i^1, PID_i^2 \in \mathbb{Z}_q^* \). Therefore, the communication cost of our scheme is \( 40 + 20 \times 2 + 4 = 84 \) bytes.

### C. Analysis on Cuckoo Filter

In this subsection, we analyze the cuckoo filter approach because the cuckoo filter have the relatively small false positive rate, which refer to the probability of falsely rejecting the null hypothesis for a particular test. The false positive rate has an influence on the efficiency of our proposed scheme. The following work shows that the impact is very small.

According to [23], the false positive of the cuckoo filter rate is \( Pr(FP) = 1 - (1 - 1/2^y) \), in which \( f \) refers to the fingerprint size of the item, \( b \) refers to the bucket size. Here, we set \( b = 4 \) because four buckets can achieve more than 95\% space utilization rate [23]. Thus, the probability of case 3 is \( Pr(case4) = 1 - (1 - Pr(FP))^2 = 1 - (1 - (1/2)^y)^2 \). We can find that the larger the value of \( f \), the lower the value of \( Pr(case4) \) and the larger space overhead the cuckoo filter. So we need adaptively change the value of \( f \). When \( f = 12 \) bits, \( Pr(case4) \) drops to about 0.00195. That is, if there are 1000 signatures which the RSU in a batch, on average only 2 signatures will be affected by the false positive and need the cuckoo filter receiver to authentication the signatures independently. The use of cuckoo filter will bring more computation and communication overhead.

First, we analyze the computation overhead. In order to get the overhead of the insert and query operation, we calculate the two operation execution time in the cuckoo filter which is written in C++ with the parameter settings as shown in Table VI. And we find that the execution time of 1000000 times operations of insertion \( T_{cuckoo-insert} \) and query \( T_{cuckoo-query} \) are 173ms and 66ms respectively. So we can get a conclusion that the insertion and query operation overhead of cuckoo filter can be neglected. However, when case 4 happens, the vehicle need to verify the message signature independently which will bring extra computation overhead. So the vehicle which needs to query one message has the average computation overhead \( T_{avg} = Pr(case4) \cdot 3T_{sm-ec} + T_{cuckoo-query} = 1.326(1 - (1 - (1 - (1/2)^y)^2) + T_{cuckoo-query} \) ms. Using the above parameter settings, the average computation overhead for the cuckoo filter query operation is only about 0.00265ms which is negligible.

Next, we analyze the communication overhead. We assume that the RSU need to send \( n \) message legitimacy information and the fingerprint size is 12 bits. After the RSU puts \( n \) fingerprints into the two cuckoo filters, it needs to sign these filters using an ECDSA signature which is 56 bytes long. Together with a message header of 2 bytes long, the total message overhead for verifying a batch of \( n \) signatures is \( 2 + [12n/8] + 56 = [12n/8] + 58 \) bytes.

### D. Analysis on the Selection Strategy on ECVs

RSU can get the instantaneous distance information by calculating the coordination which is contained in BSM packets sent from nearby vehicles. It’s noteworthy that knowing the instantaneous distance is enough to determine whether the corresponding vehicle is qualified to be the ECV. In our scheme, the ECV can finish one message authentication task at most in 500ms in most cases even if the number of messages reached one thousand. In China, the speed limit in highway is 120km/h, so the ECV only moved at most a dozen meters during the message authentication task, which is negligible compared with the valid max transmission range in DSRC standard (1000meters) [28].

We test the ECV selection algorithm based on scikit-fuzzy [30] which is a Python written fuzzy logic library. And we find that the total computation overhead of fuzzification, fuzzy roles mapping and defuzzification is only about 1.2ms on Intel i7-6700 platform. As long as RSU is equipped with a modern multi-core CPU, the computation overhead of ECV selection algorithm is negligible for RSU. The following two reasons guarantee the ECV selection strategy can be carried out all the time. First, There are always existing many vehicles moving toward the RSU which means that some of them have the potential to be ECVs at certain points. Second, the vehicle always has the extra computation resource.

### VI. Conclusion and Future Work

The instantaneous characteristics of VANETs significantly reduce the efficiency of message signing and authentication. Therefore, this paper introduces an edge-computing concept to improve the efficiency of message authentication. In our scheme, the ECVs and RSU share the authentication tasks of all the vehicles in the domain of the RSU, thus reducing the burden on the RSU, significantly decreasing redundant authentication, and improving the operational efficiency of the entire VANET system. The security analysis results show that the proposed scheme meets the basic security requirements of VANETs. The performance analysis results show that the performance of the proposed scheme is better than that of the conventional scheme, making it more suitable for deployment in real VANET scenarios.

In our proposed scheme, edge computing is only used in the message authentication process of the VANETs, i.e., only for computation, without requiring any storage. In the future, we plan to incorporate edge computing into the VANETs by allowing high frequency access information to be temporarily stored in the ECVs to enhance the efficiency of the entire VANET system.

### ACKNOWLEDGMENT

The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.
REFERENCES


Jie Cui received the Ph.D. degree in computer science and technology from University of Science and Technology of China in 2012. He is currently an Associate Professor with the School of Computer Science and Technology, Anhui University, China. He has authored over 50 papers. His current research interests include applied cryptography, Internet of Things security, vehicular ad hoc networks, and software-defined networking.

Lu Wei is currently a Research Student with the School of Computer Science and Technology, Anhui University, Hefei, China. His research focuses on vehicle ad hoc networks.

Jing Zhang is currently a Research Student with the School of Computer Science and Technology, Anhui University. Her research interest is vehicle ad hoc networks.

Yan Xu received the Ph.D. degree from University of Science and Technology of China in 2015. She is currently a Lecturer with the School of Computer Science and Technology, Anhui University. Her research interests include network and information security.

Hong Zhong received the Ph.D. degree from University of Science and Technology of China in 2005. Since 2009, she has been a Professor with the School of Computer Science and Technology, Anhui University, China, where she is currently the Dean. She has authored over 100 papers. Her research interests include applied cryptography, Internet of Things security, vehicular ad hoc networks, and software-defined networking.