An Efficient and Secure Anonymous Authentication Scheme for VANETs Based on the Framework of Group Signatures

XIAOHAN YUE¹, BING CHEN¹, XIBO WANG¹, YONG DUAN¹, MINGCHAO GAO² and YUAN HE¹ (MEMBER, IEEE)

¹School of Information Science and Engineering, Shenyang University of Technology, Shenyang, Liaoning, China
²Crops Resources Research Institute, Jilin Academy of Agricultural Sciences, Changchun, Jilin, China

ABSTRACT Vehicular Ad Hoc Networks (VANETs) have stimulated interests in both academic and industry settings. Once deployed, they would bring a new driving experience to drivers. However, in an open-access environment, privacy is one of the greatest challenges as drivers want to keep their personal information protected. Therefore, many authentication protocols have been proposed as solutions to the privacy issue. In most of the existing protocols, to prevent the revoked entity from generating a valid authentication information, the verifiers must frequently download the revocation list from one or more remote authorities to keep the list up-to-date, which greatly increases the workload of the remote authority. In this paper, to cope with such challenging concerns, based on the idea of group signatures, we propose a novel authentication protocol scheme by using the complete sub-tree method to achieve membership revocation which ensures forward security. In our scheme, the verifiers can verify the authentication information by getting the latest epoch $t$ without having to obtain the latest revocation list. In the aspect of security, the proposed protocol is featured with forward security, CCA2-anonymity, unforgeability, non-frameability and traceability. In addition, our scheme adopts a decentralized group model so that the whole domain of VANETs is divided into sub-regions and any vehicle has to update its non-revoked token periodically from the regional group manager who manages the region where the vehicle stays.

INDEX TERMS Vehicular Ad Hoc Networks, anonymous authentication, forward security, non-frameability, group signatures

I. INTRODUCTION

Vehicular Ad Hoc networks (VANETs) have recently become a popular direction for research, with specific attention to improving driving and road safety [1]. Specifically, VANETs allow vehicles to communicate with other vehicles, i.e., vehicle-to-vehicle (V-2-V) communications, or with roadside units (RSUs), i.e., vehicle-to-infrastructure (V-2-I) communications via the equipped on-board unit (OBU) communication devices as shown in Fig. 1. The specific properties of VANETs allow the development of very attractive services such as the so-called comfort services that include traffic information, weather information, locations of gas stations or restaurants, price information, and interactive communications such as the Internet access. Also, it is possible to offer safety services such as emergency warnings, lane changing assistance, intersection coordination, traffic-sign violation warnings, and road-condition warnings.

Recent research on VANETs has identified several issues, including those in security and privacy, which need to be addressed for widespread adoptions. Security issues in VANETs have been studied in great details [2, 44, 46, 47]. However, we still see a lot of open questions on the issue of privacy. For example, if a broadcast message in the VANETs is not authenticated, the OBU cannot evaluate the
A. RELATED WORKS

In the past few years, security authentication and privacy protection have been an important research direction in VANETs. Many anonymous authentication schemes have been proposed for VANETs. Most of them are based on a public key infrastructure (PKI) that one verifies messages by using traditional digital signatures which require the participation of PKI. Blum and Eskandarian have proposed a secure communication scheme based on PKI and they use a virtual network controlled by cluster heads to stand against collisions caused intentionally by attackers. This approach incurs a remarkable cost. proposes the use of authenticated data structures to issue certificate status information (CSI). However, the revocation service is decentralized to transmit the CSI and still depends on a CA to decide when a node should be evicted from VANETs.

To achieve privacy preservation and security in VANETs, Raya and Hubaux proposed a scheme based on PKI by using an anonymous certificate to hide the user’s identity. Although the certificate does not contain any identity information about the user, the adversary can still find out the user’s identity by recording the user’s certificate in different locations. To defend against this attack, Raya recommends that each user vehicle stores a large number of public/private keys and corresponding certificates, and each vehicle’s signature message uses a random certificate. Keys and certificates are periodically replaced to protect the privacy of the vehicle. This method can guarantee the integrity of the message and the privacy of the sender, but there are also some disadvantages. When the vehicle’s private key is revoked, the system needs to update the certificate frequently, which can take a lot of time. It is also hard for key distribution, management, and storage.

Shim et al. introduced a scheme based on Tso’s work to broadcast authentication messages in wireless sensor networks. The main advantage of this scheme is that it has extremely low communication overhead. But this scheme does not provide any privacy protection.

Zhang et al. have proposed an efficient identity-based batch (IBV) verification scheme for communications between vehicles and RSUs. IBV achieves conditional privacy protection through signature based one-time verification. However, IBV depends on tamper-proofing equipment severely and this equipment preassembles privacy arguments, that is, once the equipment is attacked, the whole system will be in danger. Another disadvantage of IBV is that it can trace the real identity of the vehicle, which is out of privacy requirements.

Based on group signatures, many anonymous authentication schemes in VANETs were proposed. Lu et al. proposed an efficient conditional privacy preservation protocol (called ECPP) which is based on the use of bilinear maps to attain conditional privacy for vehicles. The primary limitation of ECPP is that RSUs suffer from high latency during the pseudonym generation process. Lin et al. proposed a security conditional privacy protection protocol with ID-based signatures. In this work, each group member can replace group signatures, and only the group administrator can uncover the identity of the signer. Their protocol can solve the user’s anonymity problem. However, it was an unsatisfactory when updating the public key of an unrevoked vehicle.

In recent years, Shao et al. proposed a new VANETs authentication protocol by using the new group signatures scheme. In the protocol, anonymous tracking is not threshold tracking. However, it may lead to random tracking and reduce the user’s privacy. Liu et al. proposed an efficient anonymous authentication protocol based on signature and message recovery. The proposed protocol verifies multiple signatures by using batch operations. However, the proposed protocol was not satisfied with effective revocation. Song et al. had a research on anonymous authentication scheme based on bilinear pairing of VANETs, which needs to update the group public key when it is revoked. Azeez et al. X. Yue et al.: An Efficient and Secure Anonymous Authentication Scheme for VANETs Based on the Framework of Group Signatures
proposed an efficient anonymous authentication scheme with an efficient tracing method. However, the non-frameability was not considered, which can guarantee that a member’s signature cannot be forged by others (including TM or RSU). Zhang et al. [38] proposed a novel distributed aggregate privacy-preserving authentication solution for VANETs. In their solution, an RSU is responsible for a subgroup of VANETs and holds a private key used to produce secret shares for vehicles. Although, they give some assumptions guaranteeing that no other item can learn the secrets in a vehicle’s tamper-proof device, if a vehicle is corrupted in an RSU, the private key of the RSU would be calculated by the malicious adversary. Gao et al. [28], based on the identity-based [50] group signatures, presented a novel anonymous authentication scheme but ignored the revocation mechanism. Li et al. [26], by using the pseudo-identity method [25], constructed an anonymous conditional privacy-preserving authentication scheme. In this scheme, each OBU should restore a large number of pseudo-identities to keep the privacy of its identity.

In summary, most of the above cryptography based anonymous authentication methods do not consider the soundness of secure properties in VANETs, including forward security, anonymity, unforgeability, non-frameability and traceability; regarding the forward security in practice, that is, once a user was revoked or his/her private key was exposed, he/she or an attacker could not generate a valid signature with illegal messages that will pass the verification of other users.

B. OUR CONTRIBUTIONS

First, our scheme realizes the forward security and efficient revocations by using the complete sub-tree method. In VANETs, this is very important for preventing the revoked user from creating a valid authentication information.

Second, the proposed protocol adopts a decentralized management model, this frees the trusted third party from the heavy workload of generating the OBU group certificate and allows the RSU to obtain the revocation list from the trusted third party while the OBU can verify the availability of signature without downloading the revocation list.

Third, high-efficiency traceability is supported by our scheme. The tracing manager can track the identity of group signatures with one exponential arithmetic and one division operation of a bilinear group.

Finally, our scheme is featured with forward security, CCA2-anonymity, unforgeability, non-frameability and traceability in the aspect of cryptographic security. We also give the formal proof of the above secure properties.

Other sections are organized as follows. Section II gives the definition of the system and security models, and our purposes. In Section III, we introduce some basic knowledge of cryptography. Based on the knowledge, we propose an anonymous authentication protocol in Section IV. Section V conducts security analysis. Section VI assesses the performance. Finally, Section VII gives the conclusion.

II. SYSTEM MODEL AND DESIGN GOALS

In this section, we demonstrate the system and security models, and introduce our design goals.

A. THE SYSTEM MODEL

The system model for our anonymous authentication scheme is illustrated in Fig. 2. It consists of three major components, namely, a TM, RSUs and vehicles (OBUs are embedded).

FIGURE 2: The system model.

Tracing Manager: The TM is responsible for maintaining the whole VANETs system and is trusted. The TM is also responsible for the registration of RSUs and OBUs, when they join the network. In our scheme, the main functions of TM are as follows:

- Trace the real identity of the user vehicle who broadcasts illegal messages to the network.
- Revoke the privacy of the malicious user vehicle, after tracing its real identity.
- Maintain and update the revocation list (RL) at each revocation epoch where the revoked identity is placed in.
- Generate the initial security parameters for all OBUs and RSUs and these parameters are issued to them after the successful completion of their registration.

Due to the fact that our scheme is based on group signatures, the TM can act the role of tracing manager. We assume an anonymous authentication scheme that have its lifetime divided into revocation epochs at the beginning of which the tracing manager updates its revocation list.

RSUs: RSUs are fixed infrastructures deployed on the road-side. RSUs act as the router between the TM and vehicle users, which connect with TM by securing wire links and vehicles through a wireless channel. In our scheme, main tasks of RSUs are as follows:

- Issue the group certificate to each OBU when the vehicle enters the range of an RSU.
- Update the RL at the revocation epoch $t$.

VOLUME 4, 2016 3
In our scheme, each RSU plays the role of group manager in the definition of group signatures. Therefore, the whole VANETs can be divided into many groups and each group has an RSU as the group manager. When a vehicle moves from one group to another group, it will obtain a new group certificate.

OBUs: Each vehicle in the VANETs is installed with an OBU, which permits the vehicle to communicate with other vehicles and RSUs to share messages to make driving more comfortable. In our scheme, functions of the OBU are as follows:

- Join a new group and get the corresponding group certificate when entering the range of an RSU.
- Request the non-revoked token from an RSU at the epoch $t$.
- Generate the group signature by using its secret key and group certificate.
- Verify the validity of group signatures from other OBUs.

To be noted, the RSU will not send group certificates and non-revoked tokens to OBUs which have already existed in the revocation list under the administration of TM.

B. THE ATTACK MODEL
The attackers in a VANETs system are primarily classified as internal attackers and external attackers. Since the external attackers are not part of the system, they are believed to be the most powerful attackers compared to internal attackers.

The compromised user vehicles and RSUs are regarded as internal attackers. Internal attackers have the right to use secret parameters since they are part of the system. In this section, we give some main potential attacks in the VANETs system.

- **Forgery Signatures**: An adversary can forge a valid group signature with forgery of the group certificate or non-revoked tokens.
- **Identity Disclosure**: After gathering some group signatures and group signing keys, the adversary can reveal the identity of a signer. In this attack, the RSU may be corrupted.
- **Untraceable Signatures**: An adversary can generate a group signature which can confuse the TM to track the real signer.
- **Revoked Key Exposure**: An adversary can still generate valid group signatures to other verifiers by using the signing key of the revoked user for past or future time periods.
- **Framing Attack**: An adversary is able to create a judge-accepted proof that an honest signer produced a certain valid signature unless this signer really did produce this signature. In this attack, the resource of the whole system will be given, including secret keys of the TM and RSUs.

C. DESIGN GOALS
The design target of this scheme is to develop an efficient and secure anonymous authentication protocol for VANETs. It should have the following required properties:

- **Efficient Verification**: OBUs can verify the validity of the signature without downloading the revocation list.
- **Efficient Revocation**: The TM can reveal the identity of any signer at a constant level of computation and communication costs. In the meantime, OBUs do not need to retrieve the latest revocation list from a remote CA or TM.
- **Unforgeability**: Only an OBU holding one group certificate from the RSU can generate a valid signature on behalf of the group maintained by the RSU.
- **Anonymity**: Only the TM can reveal the identity of the signer. In other words, if the OBU is out of its communication range, even the RSU will not be able to display the location of any OBU.
- **Traceability**: No OBU can generate any valid signature tracking to other OBUs.
- **Forward Security**: Compromise of the OBU’s group signing key of the current period does not enable an attacker or revoked user to forge group signatures pertaining to the past or future time periods.
- **Non-frameability**: The non-frameability can guarantee that any OBU (including the TM or RSUs) should not be able to produce a valid group signature on behalf of another OBU.

III. PRELIMINARIES
This section introduces some basic knowledge related to anonymous authentication protocols, including group signatures, bilinear groups and mathematical assumptions, complete sub-tree methods, and BBS+ signatures.

A. DEFINITIONS OF GROUP SIGNATURES
Group signatures [12] is a special digital signature, where there exist three kinds of entities: a group manager, a group tracer and some group members. It allows group members to sign messages standing for the group, others can not reveal the identification of the signer except the group tracer. The scheme is composed of the following five algorithms: SETUP, CERTGEN, SIGN, VERIFY and OPEN.

The SETUP algorithm takes security parameters as the input and outputs public/private key pairs for each entity (the group manager, tracker, and group members) in the system.

The CERTGEN algorithm uses the private key of the group administrator, the group member and the public key of the tracer as the input, and outputs a group certificate corresponding to the public key of the input group member.

The SIGN algorithm inputs the group certificate, public key of the group member and a message $M$, and outputs the corresponding signature.

The VERIFY algorithm verifies the validity of the message and the message signature and outputs 1 if the signature is valid, or 0 otherwise.
The OPEN algorithm is operated by the tracer, and using the tracer’s private key, message, and valid signature as the input, and outputs the group members who generated the signature.

On the other hand, schemes based on group signatures should also satisfy the following three security attributes: unforgeability, unannonymity and traceability. The first security attribute ensures that only group members can generate signatures on behalf of the group. The second security attribute ensures that no one other than the tracker can reveal the identity of its signer. The third security attribute ensures that all valid signatures can be tracked by the tracer.

**B. BILINEAR GROUPS AND COMPLEXITY ASSUMPTIONS**

Let $G_1$ and $G_2$ be cyclic groups of prime order $p$, and $g_1, g_2$ are generators of $G_1, G_2$ respectively. Let $G_T$ be a multiplicative cyclic group with the same order, and define $\text{par}_{\text{Bilinear}} = (p, G_1, G_2, G_T, e, g_1, g_2)$ as the set of pairing group parameters. Bilinear pair $e(G_1, G_2) \rightarrow G_T$ is a map that satisfies the following properties:

- **Bilinearity**: $e(g_1^m, g_2^n) = e(g_1, g_2)^{mn}$ for all $m, n \in \mathbb{Z}_p$, any $g_1 \in G_1$ and $g_2 \in G_2$.
- **Non-degeneracy**: The bilinear map $e(G_1, G_2)$ generates $G_T$.
- **Computability**: The function $e$ is efficiently computable.

There are some complexity assumptions used in the implementation of our protocol.

**Definition 1** (The Discrete Logarithm assumption (DL)). The DL assumption holds in $G_1$, if the following probability is negligible in the security parameter $\kappa$, for all adversaries $A$ and all parameters $\text{par}_{\text{Bilinear}}$:

$$\text{Adv}_{\text{DL}}^{\text{DL}}(\kappa) = \text{Pr}[x \leftarrow \mathbb{Z}_p; u = v^x, v \leftarrow G_1 : A(u, v, \text{par}_{\text{Bilinear}}) \rightarrow x]$$

**Definition 2** (The Decisional Diffie-Hellman assumption). The DDH assumption holds if the following probability is negligible in the security parameter $\kappa$, for all adversaries $A$:

$$\text{Adv}_{\text{DDH}}^{\text{DDH}}(\kappa) = \text{Pr}[\Lambda(u, u^\alpha, u^\beta, z) = 1 | z = u^{\alpha\beta}] - \text{Pr}[\Lambda(u, u^\alpha, u^\beta, z) = 1 | z = u^\gamma]$$

**Definition 3** (The eXternal Diffie-Hellman assumption). Let $e : G_1 \times G_2 \rightarrow G_T$ be an asymmetric bilinear map, if DDH assumption is hard in group $G_1$, then XDH assumption holds.

**Definition 4** (The $\eta$-Strong Diffie-Hellman assumption). The $\eta$-SDH assumption holds if the following probability is negligible in the security parameter $\kappa$, for all adversaries $A$ and all parameter sets $\text{par}_{\text{Bilinear}}$:

$$\text{Adv}_{\text{sdh}}^{\eta-\text{SDH}}(\kappa) = \text{Pr}[x \leftarrow \mathbb{Z}_p; g_1^x, g_1^2, \ldots, g_1^\eta \leftarrow G_1; g_2^x \leftarrow G_2 : (g_1^x, g_1^2, \ldots, g_1^\eta, g_2^x, \text{par}_{\text{Bilinear}}) \rightarrow (g_1^{1/(\eta+\epsilon)}, c \in \mathbb{Z}_p)]$$

**C. BBS+ SIGNATURE**

The BBS+ signature scheme [19] is introduced in the following statements:

Given $(p, G_1, G_2, G_T, e)$, let $g_0, g_1, \ldots, g_L, g_{L+1}$ be the generators of $G_1$, $h$ be a generator of $G_2$.

**Key Generation**: Select $\gamma \leftarrow \mathbb{Z}_p$ randomly, and let $w = h^\gamma$. The secret key is $sk = \gamma$ and the verification key is $vk = w$.

**Signing**: For the message $(m_1, \ldots, m_L)$, choose $\eta, \zeta \leftarrow \mathbb{Z}_p$ randomly and compute $s = (g_0 g_1^\eta g_2^m \ldots g_{L+1}^\zeta)^\lambda$, where $\lambda = (\eta + \gamma)^{-1}$. Let the signature be $\sigma = (s, \eta, \zeta)$.

**Verifying**: For the signature $\sigma = (s, \eta, \zeta)$ and the message $(m_1, \ldots, m_L)$, if $e(s, h^\eta v k) = e(g_0 g_1^\eta g_2^m \ldots g_{L+1}^\zeta, h)$ then output 1, otherwise output 0.

This signature scheme has unforgeability against chosen message attack (CMA) under the $\eta$-SDH assumption [20].

**D. THE NNL FRAMEWORK**

A subset cover framework was proposed by Naor et al. [21], which can be used to revoke members and trace illegal members. The technique is also used to construct the broadcast encryption [51]. This framework can be implemented in two ways: subset difference (SD) and complete sub-tree (CS). Let $N$ be the set of all group members, and $R \subset N$ be the set of revoked members. Such that, the set of non-revoked members are divided into $m$ disjoint sets where $m$ is the number of subsets.

Using the complete sub-tree method, a private key is assigned to each node of the binary tree, each user is assigned to a leaf node of the binary tree, and let $\{n_0, n_1, \ldots, n_l\}$ be the path from the root node to the leaf node, where $l$ is the height of the complete binary tree. Then, the user obtains a key associated with each $n_i \in \{n_0, n_1, \ldots, n_l\}$. A ciphertext is computed by keys of nodes defined by the method. Let $\Theta = \{n_0', n_1', \ldots, n_m'\}$ be a set of nodes and their corresponding keys are used to encrypt. If a user’s path is $\{n_0, n_1, \ldots, n_l\}$, which is indicated as the authorized receiver, then there is a node $\varepsilon$ such that $\varepsilon \in \{n_0, n_1, \ldots, n_l\} \cap \{n_0', n_1', \ldots, n_m'\}$. Therefore, the user can decrypt the ciphertext using the private key corresponding to the node $\varepsilon$.

In our scheme, we will use the complete sub-tree method to construct our Revoke and Update algorithms.

**IV. ANONYMOUS AUTHENTICATION PROTOCOL FOR VANETS**

In this section, according to the idea of group signatures, we firstly present the formal definition of our scheme which consists of eight probabilistic polynomial-time algorithms. Then based on the definition, a new anonymous authentication protocol is presented for VANETs. The notations used in this paper are defined in TABLE I.

**A. DEFINITION OF OUR SCHEME**

Our anonymous authentication scheme consists of the following eight algorithms.
TABLE 1: Notations

<table>
<thead>
<tr>
<th>Notations</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>gpk</td>
<td>The global public key generated by TM</td>
</tr>
<tr>
<td>tkTM</td>
<td>A tracing key of TM</td>
</tr>
<tr>
<td>rkTM</td>
<td>A non-revoked token secret key of TM</td>
</tr>
<tr>
<td>skGM</td>
<td>A secret key of RSU</td>
</tr>
<tr>
<td>pkGM</td>
<td>A group public key of RSU</td>
</tr>
<tr>
<td>skOBU</td>
<td>A secret key of OBU</td>
</tr>
<tr>
<td>pkOBU</td>
<td>A public key of OBU</td>
</tr>
<tr>
<td>certOBU</td>
<td>A public key certificate of OBU</td>
</tr>
<tr>
<td>t</td>
<td>A revocation epoch</td>
</tr>
<tr>
<td>RL_i</td>
<td>A revocation list for a revocation epoch t</td>
</tr>
<tr>
<td>RC_i</td>
<td>A set of revoked public key certificates</td>
</tr>
<tr>
<td>token_{i,t}</td>
<td>A non-revoked token used by OBU_{i}</td>
</tr>
<tr>
<td>RC</td>
<td>The set of non-revoked nodes</td>
</tr>
<tr>
<td>gCert_i</td>
<td>A group certificate of OBU_{i} issued by an RSU</td>
</tr>
<tr>
<td>tag_i</td>
<td>A tag used by an OBU_{i} for obtaining token_{i,t}</td>
</tr>
<tr>
<td>M</td>
<td>A message sent by OBU</td>
</tr>
<tr>
<td>σ</td>
<td>A group signature</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>A hash function H : {0, 1}^* \rightarrow Z_p</td>
</tr>
</tbody>
</table>

1) System Initialization

In this phase, every appliance in VANETs performs the initialization.

**Setup(\lambda):** For TM, it takes as inputs a security parameter \lambda \in \mathbb{N} and outputs the global public key gpk, the secret tracing key of the tracing manager tkTM, the non-revoked token secret key rkTM. After the execution of Setup, TM publishes the global public key gpk. For RSU, taking the gpk as input, it outputs the secret key skGM and group public key pkGM of the group manager. For OBU, taking the gpk as input, it creates its secret signing key skOBU and corresponding personal public key pkOBU.

2) Member Registration

In this phase, each RSU and OBU interacts with TM to get the public key certificate cert_{RSU} and cert_{OBU} respectively.

**Register_{RSU}(skGM,pkGM):** Each RSU interacts with the TM to prove the knowledge of private key skGM by using zero-knowledge proof protocol. After that, the TM will output the public key certificate cert_{RSU} on pkGM.

**Register_{OBU}(skOBU,pkOBU):** Each OBU interacts with the TM to prove the knowledge of private key skOBU by using zero-knowledge proof protocol. After that, the TM will assign a tag_i to the OBU and output the public key certificate cert_{OBU} on pkOBU.

3) Member Revocation

**Revoke\((gpk, rkTM, t, RC_t)\):** It is an algorithm allowing TM to generate an updated revocation list RL_t = \{t, RC_t, \Phi = \{token_{i,t}\}_{i=0}^m\} for the new revocation epoch t, where the token set \Phi contains all tokens of non-revoked users. It takes as input the set RC_t of public key certificate of revoked OBUs for gpk, rkTM, revocation epoch t, and outputs the revocation list RL_t for epoch t.

4) Member Joining

**Join\((OBU: (cert_{OBU}, pkOBU), RSU: (cert_{RSU}, pkGM, RL_i))\):** It is an interactive protocol between the group manager RSU and an user OBU when the latter becomes a group member. If it is the first time for the user to join the group, the execution of the protocol terminates with user OBU i obtaining a group membership certificate gCert_i. To be noted, the revoked user in the set RC_t cannot be allowed to join any group.

5) Message Signing

**Sign\((gpk, t, token_{i,t}, gCert_i, skOBU, M)\):** Given an epoch t with an updated token_{i,t}, a group membership certificate gCert_i, a secret signing key skOBU, and a message M, this algorithm outputs a group signature \sigma as the anonymous authentication information.

6) Signature Verification

**Verify\(σ, M, t, gpk\):** Given a signature \sigma, a revocation epoch t, a message M and global public key gpk, this deterministic algorithm returns either 0 or 1.

7) Signature Tracing

**Trace\((M, t, RL_i, σ, tkTM, gpk)\):** The TM can use this algorithm to reveal the identity corresponding to a valid signature \sigma on a false message M and sends the updated revocation list to each RSU. Take as input a message M, a valid signature \sigma for the indicated revocation in epoch t, and the secret key tkTM, this deterministic algorithm returns the public key certificate cert_{OBU,i}, where i is the identity of an OBU. After revealing the identity i, the TM adds cert_{OBU,i} to RC_t which is included in RL_t and sends the updated revocation list to each RSU via authenticated channels.

8) Token Updating

At each revocation epoch, the OBU needs to update its non-revoked token by interacting with the RSU. We assume that the RSU can obtain the up-to-date revocation list RL_i.

**Update\((OBU: (cert_{OBU}, pkOBU, tag), RSU: (cert_{RSU}, RL_i))\):** At a revocation epoch t, the OBU i sends a request to the RSU for obtaining a new non-revoked token. Upon receiving the request, the RSU checks whether the cert_{OBU,i} exists in RC_t. If not, the RSU determines a token token_{i,t} from the set \Phi in RL_t according to the OBU’s tag_i, then returns the token to the OBU.

8. ANONYMOUS AUTHENTICATION PROTOCOLS FOR VANETS

Now, we present the details of our secure anonymous authentication scheme, which consists of eight parts: system initialization (Initialization), party registration (Registration), OBU join (Join), message signing (Sign), signature verification (Verify), identity reveal (Trace) and OBU revocation (Revoke), non-revoked token updating (Update).
1) System Initialization Stage

In this stage, the TM generates parameters of the whole system and its own private key. In addition, RSU and OBU also generate their own private key pair. Details are as follows:

- The TM performs the following steps by running the algorithm Setup(λ):
  - Choose an asymmetric bilinear group pair \((G_1 = \langle g_1 \rangle, G_2 = \langle g_2 \rangle)\) of prime order \(p \in \{0, 1\}^\lambda\) and a pairing function \(\varepsilon: G_1 \times G_1 \rightarrow G_T\).
  - Select \(\tilde{g}_1, \tilde{g}_1 \leftarrow G_1\) randomly and a hash function \(H\) which is a secure cryptographic hash function \(H(\cdot): \{0, 1\}^* \rightarrow Z_p\).
  - Select a secret key \(\gamma \leftarrow Z_p\) randomly, and output \((rk_{TM}, pk_{TM}) = (\gamma, g_2^\gamma)\).
  - Select \(x'_1, y'_1, y'_2 \leftarrow Z_p\) randomly, and compute \(\varphi_1 = g_1^{x'_1} g_1^{y'_1}, \varphi_2 = g_1^{y'_1} g_1^{y'_2}\).
  - Select a secret key \(\nu \leftarrow Z_p\) randomly, and compute \(u = g_2^{\varphi_1}\). Let \(tk_{TM} = \nu\) as the tracing key of the TM.
  - Keep \(rk_{TM}\) and \(tk_{TM}\) secret.
  - Publish global public system parameters \(gpk = (p, G_1, G_2, G_T, \varepsilon, g_1, g_2, \tilde{g}_1, \tilde{g}_1, \varphi_1, \varphi_2, u, H, pk_{TM})\).

- Each RSU does the following steps:
  - Select secret key \(\omega \leftarrow Z_p\) randomly, and compute \((sk_{GM}, pk_{GM}) = (\omega, g_2^{\omega})\), \(pk_{GM}\) is the public key of RSU, while \(sk_{GM}\) is the private key of RSU.
  - Keep \(sk_{GM}\) secret.

- Each OBU does the following steps:
  - Select secret key \(\chi \leftarrow Z_p\) randomly, and compute \((sk_{OBU}, pk_{OBU}) = (\chi, g_2^\chi)\), \(\chi\) is the private key of OBU.
  - Keep \(sk_{OBU}\) secret.

2) Registration Stage

In this stage, the TM will issue public key certificates for each RSU and OBU and create non-revoked tokens for each OBU. The details are stated below.

- Each RSU interacts with the TM by using some zero-knowledge proof protocol, such as that in [24], to prove the knowledge of the secret key \(sk_{GM}\). After that, the TM sends the RSU the public key certificate \(cert_{RSU}\) on \(pk_{GM}\).

- Each OBU also interacts with the TM by using the zero-knowledge proof protocol to obtain the public key certificate. Except that, according to the CS method in NNL framework, the TM will assign to OBU \(i\) an available leaf node \(v_i\) of the binary tree and a path \(\rho_i := (\eta = \varepsilon, n_1, \ldots, n_t = v_i)\) connecting the leaf \(v_i\) to the root \(\varepsilon\) of the tree. Then the TM sends a public key certificate \(cert_{OBU}\) on \(pk_{OBU}\) and a path \(\rho\) to OBU via an authenticated and secure channel. In our scheme, we call the path \(\rho_i\) as the tag \(tag_i\) for an OBU \(i\) to request the non-revoked token from the group manager RSU.

- After issuing a public key certificate to an OBU, the TM will perform the Revoke algorithm to update the set of non-revoked tokens and restore \(\{i, cert_{OBU}, i, pk_{OBU}, i, tag_i\}\) into the corresponding database, such as DB\_cert.

3) Joining Stage

This stage happens when an OBU travels within the communication range of one RSU, which is an interactive phase between the OBU and the RSU.

- When one OBU enters the communication range of an RSU, the OBU will issue a request message to obtain the RSU’s group public key.

- After receiving the request message, the RSU returns its own group public key \(pk_{GM}\) and certificate \(cert_{RSU}\).

- The OBU receives the RSU’s group public key \(pk_{GM}\) and certificate \(cert_{RSU}\), then checks its validity. If valid, the OBU encrypts its own certificate \(cert_{OBU}\) and \(pk_{OBU}\) and a node \(\varepsilon \in tag\) with the public key of RSU, then sends the ciphertext \(c_{OBU}\) to the RSU (The node \(\varepsilon\) was determined in the Update algorithm).

- Upon receiving \(c_{OBU}\), the RSU performs the following steps:
  - Decrypt the ciphertext \(c_{OBU}\) to obtain \((cert_{OBU}, pk_{OBU}, \varepsilon \in tag)\) by using \(sk_{GM}\).
  - Check whether \(cert_{OBU}\) exists in the RC\_1 and the validity of \((cert_{OBU}, pk_{OBU})\). If the answer is yes, go to the next step; otherwise, abort this stage.
  - Compute the group membership certificate \(gCert = (g_1\tilde{g}_1^\varepsilon \cdot pk_{OBU})^{1/(sk_{GM} \cdot \eta)}\), where \(\eta \in R_{Z_p}\).
  - Send \((gCert, \eta)\) if the OBU is a new group member.
  - Send a copy of the OBU’s \((cert_{OBU}, pk_{OBU}, gCert)\) with group identity ID\_Group to the TM who will restore the copy to the corresponding database, such as DB\_gCert.
  - Add the OBU’s \((tag, cert_{OBU}, pk_{OBU}, gCert)\) to the local group member list.

- Upon receiving the response \((gCert, \eta)\) from the RSU, the OBU does the following steps:
  - Check whether \(e(gCert, pk_{GM}^\eta) = e(pk_{OBU}, g_2) \cdot e(g_1, \tilde{g}_1, g_2)\). If yes, the OBU accepts its group certificate \(gCert = (g_1\tilde{g}_1^\varepsilon \cdot g_2)\).

4) Signing Stage

![FIGURE 3: The message format.](http://www.ieee.org/publications_standards/publications/rights/index.html for more information.)
RSU issues the group certificate to the OBU. The signature part is the OBU signature of the first five parts.

When entering the message $M \in \{0, 1\}^*$, the OBU signs the message with the SIGN algorithm and broadcasts the signature together with a message $M$. The details are as follows:

- Select $\zeta \leftarrow Z_p$ randomly, and output: $\psi_1 = gCert \cdot n^\zeta$, $\psi_2 = \text{token}_{i,t} \cdot g_1^\zeta$, $\psi_3 = g_1^\zeta$, $\psi_4 = \sigma$, $\psi_5 = (\varphi_{1} \varphi_{2})^\zeta$, where $h = H(\psi_1 || \psi_2 || \psi_3 || \psi_4)$.
- Let $\alpha = \zeta \cdot \eta$ and $\beta = \zeta \cdot \eta'$.
- Prove that the value $(\zeta, \alpha, \beta, \varepsilon, \chi, \eta, \eta')$ satisfies the following relation.
  \[
  e(\psi_1, g_2)^{-\gamma} e(\sigma, g_2)^{\gamma} e(\varepsilon, g_2)^{\gamma} e(u, g_2)^{\gamma} e(\psi_1, pk_{GM})^{\gamma} e(g_1, g_2)^{\gamma} e(\psi_3, pk_{TM})^{\gamma} e(\varepsilon, g_2)^{\gamma} = e(\psi_1, pk_{GM}) e(g_1, g_2) e(\psi_3, pk_{TM}) e(\varepsilon, g_2)^{\gamma} = e(\psi_3, pk_{TM}) e(\varepsilon, g_2)^{\gamma} \]

5) Verification Stage

In this phase, the OBU runs the Verify algorithm to check the validity of the received message $(M, \sigma)$. The details are as follows:

- Compute the following values.
  \[
  R_{gCert} \leftarrow e(\hat{g}_1, g_2)^{\gamma} e(\psi_1, g_2)^{-\gamma} e(\hat{g}_1, g_2)^{\gamma} e(u, pk_{GM})^{\gamma} e(g_1, g_2)^{\gamma} e(\psi_1, pk_{GM})^{\gamma} e(g_1, g_2)^{\gamma} e(\psi_3, pk_{TM})^{\gamma} \]

6) Tracing Stage

In this stage, by using the tracing key $tk_{TM}$, the TM can reveal the identity corresponding to a valid group signature $\sigma$ on a false message. The details are as follows.

- Compute the group certificate $gCert = \psi_1 \psi_3^{tk_{TM}}$.
- Search $gCert$ in the database $DB_{Gcert}$ and get the corresponding public key certificate $cert_{OBU}$.
- Add $cert_{OBU}$ into $RC_t$.
- Run the Revoke algorithm to update the revocation list.

7) Revocation Stage

The TM updates the revocation list $RL_t$ periodically or when a user is revoked. In this stage, the TM will update the non-revoked token set $\Phi$, and the revoked public key certificate set $RC_t$ at an epoch $t$. The details are as follows.

- Using the CS covering algorithm, determine the non-revoked node set $\Theta = \{n_0, n_1, \ldots, n_m\}$, where $m$ is the number of the non-revoked users.
- For $i = 1$ to $m$, select $\eta'_i \leftarrow Z_p$ randomly, and output the non-revoked token ($\text{token}_{i,t} = (g_1^{\eta'_i} \cdot g_2^{\eta'_i})^{1/(rk_{TM} + \eta'_i)}$).
- Send the updated $RL_t = \{t, RC_t, \text{token}_{i,t}\}_{i=0}^m, \Theta$ to each RSU via an authenticated and secure channel.

8) Updating Stage

In this stage, the OBU updates its non-revoked token periodically from the RSU. Processes of the interaction between the OBU and the RSU are as follows.

- The OBU encrypts its own certificate $cert_{OBU} \cdot pk_{OBU}$ and tag with the public key of the RSU, then sends the resulting $c'_{OBU}$ to the RSU.
• Upon receiving $c_{OBU}$, the RSU performs the following steps:
  - Check whether $cert_{OBU}$ exists in the RC_i and the validity of $(cert_{OBU}, p_{OBU})$. If all answers are yes, go to the next step; otherwise, abort this stage.
  - Select a node $\varepsilon$ from the intersection of the tag and the node set $\Theta$, as described in Section 3.
  - Look up the non-revoked token set $\Phi = \{token\}_i$ in revocation list RL_i to get the token corresponding to the node $\varepsilon$.
  - Forward the selected token $\varepsilon_i$ and $\varepsilon$ to the OBU.

V. SECURITY PROOFS

The proposed anonymous scheme has some secure properties, including forward security, anonymity, non-frameability, unforgeability where the forward security is straightforward, because it can be seen from the process of proving the knowledge of the non-revoked token at an epoch $t$.

When a user was revoked at the previous revocation epoch, he cannot obtain the newest token from any OBU. Therefore, he cannot establish any proof to prove the following relation at the newest epoch $t$.

$$e(g_2, g_2)^{-\eta} = e(g_1^t, g_2)^{\beta} e(g_1, g_2)^{\xi} e(\hat{g}_1, g_2)^{\zeta}$$

In other words, unless the revoked user or attacker can forge a valid token, he will not generate a valid proof for the coming epoch. The unforgeability of token is given in Theorem 3. So, the forward security can be guaranteed.

In the following paragraphs, we will give the security proof that our scheme has anonymity, non-frameability, traceability and unforgeability.

Theorem 1. The proposed scheme has IND-CCA2 anonymity in the random oracle model under the XDH assumption, where $H$ is modeled as a random oracle.

Proof. Anonymity means that the signer cannot be identified without the private key of the opener, so the attack on the anonymity of the attacker is equal to the attack on the ciphertext $(\psi_1, \psi_2, \psi_3, \psi_4, \psi_5)$, that is, the anonymity of the scheme is reduced to CCA2 security under the XDH assumption. The idea of the proof is: Suppose an adversary $A$ can break the anonymity of our scheme with a non-negligible probability and hash/signature queries, then there is a polynomial-time algorithm $S$ that solves the XDH problem with a non-negligible probability. The concrete proof is given below. This proof proceeds with a sequence of games. First, we define the games. In the following context, we denote by $G_i$ the event in Game i whether the adversary successfully guesses the bit picked by the challenger.

Game 0. The initial game is the same as the game defined in the definition of anonymity. First we assume that the challenger responds to the hash inquiry. For this purpose, the challenger maintains a hash list which contains tuples of the form $(M, \psi_1, \psi_2, \psi_3, \psi_4, \psi_5, R_1, R_2, R_3, R_{Cert}, R_{token}, c)$ for the hash function $H$.

Game 1. In this game, a simulated proof is used instead of a zero-knowledge proof of the challenge signature. When the adversary requests a challenge signature $(\psi_1, \ldots, \psi_5, c^*, R_1, \ldots, R_{token})$ by sending $i_0, i_1, M$, the challenger makes the following calculation.

The adversary selects a bit $b \in \{0, 1\}$ randomly, computes $(\psi_1, \ldots, \psi_5)$ by signature private signing key $(g_{Cert_{i_0}}, s_{i_0})$, generates $(\xi, \eta, \zeta, \eta^*, \zeta^*)$, $c^*, c_{\xi}^*, c_{\zeta}^*$, $\cdot \cdot \cdot$ (the specific method is not given here). The challenger outputs $e(\hat{g}_1, g_2)^{\zeta^*} (\eta^*)$, then the challenger outputs $(\hat{g}_1, g_2)^{\zeta^*}$ to the adversary as the challenge.

Game 2. In this game, we modify the encryption in the challenge to be "invalid". Specifically, in order to compute the challenge $(\psi_1, \ldots, \psi_5, c_i, s_1, \ldots, s_5)$, the challenger selects random integers $a, b, c \in Z_p$, where $c \neq a, b$, computes $u_1 = \hat{g}_1^t, u_2 = \hat{g}_2^t$, let $\tilde{g} = \tilde{g}_1^t, u = \tilde{g}_1^t \tilde{g}_2^t$ in the global public key $gpk$, and then outputs:

$$\begin{align*}
\psi_1 &= g_{Cert_{i_0}} \cdot u_{1}^{z_1} u_{2}^{z_2} \\
\psi_2 &= \text{token}_{i_0} \cdot u_{1}^{z_1} u_{2}^{z_2} \\
\psi_3 &= u_{1} \cdot \psi_4 = u_{2} \\
\psi_5 &= u_{1}^{x_1+y_1} u_{2}^{x_2+y_2} h
\end{align*}$$

where $b$ is selected in $\{0, 1\}$ randomly, $g_{Cert_{i_0}}$ is a part of the group member certificate of member $i_0$, $\text{token}_{i_0}$ is the signature of the TM corresponding to member $i_0$ in the non-revoked token set $\Phi$. The challenger computes the challenge value by open private key $s_{k_{i_0}, M}$ (namely $(z_1, z_2)$). Other challenge values are computed in the same way as Game 1. Based on the XDH assumption, the effect of this change on the adversary’s probability of winning is negligible.
(\tilde{R}_1, \ldots, \tilde{R}_{g\text{Cert}}), \text{ where } (\tilde{R}_1, \ldots, \tilde{R}_{\text{token}}) \text{ are the group elements reproduced in the verification process. This change does not affect the adversary’s advantage non-negligibly.}

**Game 4.** In this game the tracing oracle should reject a signature which contains an invalid ciphertext tuple \((\psi_1, \psi_3, u_4)\) where \(c \neq ab\). All other queries are treated as before. This modification does not affect the behavior of the adversary, as the adversary can issue such an invalid query under a proof with negligible probability only.

**Lemma 1.1.** \(\Pr[\mathcal{G}_0] - \Pr[\mathcal{G}_1]\) is negligible.

**Proof.** We claim that the distribution (of the challenge) in Game 1 is identical to that in Game 0 except for cases in which the challenger outputs \(\bot\). This follows from a standard discussion of the simulation of zero-knowledge proof. To see this, we can observe that \(s^* - c\alpha\) in Game 1 corresponding to \(r_a\) in Game 0, and similar correspondence holds for all other \(s^*\)’s and \(r\)’s. We can also see that both \(s^* - c\alpha\) and \(r_a\) are uniformly distributed over \(Z_p\). We will then see that the challenger in Game 1 outputs \(\bot\) only with negligible probability. It can be obtained from the fact that \((\tilde{R}_1, \ldots, \tilde{R}_{\text{token}}\)} are distributed uniformly over a set with cardinality (at least) \(p\), that is, the oracle queries to \(H\) issued before the challenge phase contain \((M, \psi_1^*, \ldots, \psi_p^*, g\text{Cert}^*, \ldots, R_3, \tilde{c})\) with probability (at most) \(q_1/p\) where \(q_1\) denotes the number of oracle queries to \(H\) issued by the adversary.

**Lemma 1.2.** \(\Pr[\mathcal{G}_1] - \Pr[\mathcal{G}_2]\) is negligible, provided that the XDH assumption holds in \(G_1\).

**Proof.** We will describe a distinguishing algorithm \(S\) of the XDH problem to bound the absolute difference \(\Pr[\mathcal{G}_1] - \Pr[\mathcal{G}_2]\). The algorithm receives a tuple \((\tilde{g}_1, \tilde{g}_2, \tilde{g}_3, \tilde{g}_4)\), where \(c\) is either \(a\) or \(b\), together with the description \((p, G_1, G_2, G_T, c, g_1, g_2)\) of the asymmetric bilinear groups. The distinguisher sets up the scheme by choosing \(x_1, x_2, y_1, y_2, z, \omega, Z_2, Z_3, z_1, z_2, r_1, r_2 \leftarrow Z_p(1 \leq i \leq n), g_1 \leftarrow G_1\) setting \(\tilde{g}_1 = g_1^{x_1}, \tilde{g}_2 = g_1^{x_2}, \tilde{g}_3 = g_1^{r_1}, \tilde{g}_4 = g_1^{r_2}, u = g_2^{z_1}, g_{\text{PKTM}} = g_2^{z_2}, g_{\text{PKGM}} = g_2^{z_3}\) and \(t_{\text{KTM}} = (z_1, z_2)\), and creating group certificates \(g\text{Cert} = (g_1^{\tilde{g}_1^{x_1}}, g_1^{\tilde{g}_2^{x_2}})^{1/(\omega + \eta_n)}\), non-revoked tokens \(t_{\text{KTM}} = (g_1^{\tilde{g}_1^{x_1}}, g_1^{\tilde{g}_2^{x_2}})^{(1/(\omega + \eta_n))}\) as usual. Queries from the adversary \(A\) to the random oracle \(H\) are responded in the ordinary manner, that is, all fresh queries are responded with a random hash value and are recorded together with the hash value, while previously issued queries are responded in the same way as in the previous query. Opening queries are responded as specified in the scheme, that is, the distinguisher first verifies the NIZK proof and if the proof passes the verification, the distinguisher decrypts the linear encryption part \((\psi_1, \psi_2, \psi_3, \psi_4)\) using \((z_1, z_2, z_1, z_2)\), otherwise return \(\bot\). When the adversary requests a challenge regarding \((i_0, i_1, M)\), the distinguisher proceeds as follows:

To compute the challenge \((\psi_1^*, \ldots, \psi_p^*, c^*, s_1^*, \ldots, s_3^*)\), the distinguisher flips a bit \(b\) and sets \((\psi_1^* = g\text{Cert}_b)\), \(u_1^* \cdot u_2^* \cdot u_3^* = (\psi_1 \cdot \psi_3 \cdot \psi_4 \cdot \psi_5) = u_1^* \cdot u_2^* \cdot u_3^* \cdot u_4^* \cdot u_5^* = u_{11}^{x_1 + y_1} u_{12}^{y_1} u_{22}^{y_2} u_{23}^{y_3} u_{24}^{y_4} u_{25}^{y_5} u_{33}^{y_5} u_{34}^{y_4} u_{35}^{y_3} u_{44}^{y_4} u_{45}^{y_5} u_{55}^{y_5}\) where \(h = H(\psi_1^* \ldots | \psi_s^*\)} the zero-knowledge proof \((c^*, s_1^*, \ldots, s_3^*)\) is computed with the simulation algorithm as in Game 1. The distinguisher sends the challenge computed as above to the adversary. After receiving the challenge, the adversary further makes queries to the random oracle and the opening oracle, which are responded as before by the distinguisher. Finally, the adversary outputs the guess \(b^*\). The distinguishing outputs 1 if \(b^* = b\), outputs 0 otherwise.

We observe that when the distinguisher receives a random tuple \((c \neq ab)\), the adversary’s view is equivalent to that of Game 2. In contrast, when the distinguisher receives a valid tuple \((c = ab)\), we can see that the view is identical to that of Game 1, as the equation \(u_1^* \cdot u_2^* = u_1^* + \alpha \cdot z_2 = u_1^*\) holds. Finally, the lemma follows from the inequality \(\Pr[\mathcal{G}_1] - \Pr[\mathcal{G}_2] = |\Pr[\mathcal{G}_1] - \Pr[\mathcal{G}_2]| = |\Pr[\mathcal{G}_1, g_1^*, g_1^*, g_4^*]| - |\Pr[\mathcal{G}_1, g_1^*, g_1^*, g_4^*]| c \neq ab| = \text{Adv}_{\text{SDH}}(\lambda).\)

**Lemma 1.3.** \(\Pr[\mathcal{G}_2] - \Pr[\mathcal{G}_3] \leq 1/p\)

**Proof.** If the adversary can produce queries satisfying the condition \((R_1, \ldots, R_{\text{token}}) = (R_1^*, \ldots, R_{g\text{Cert}}^*)\) and \((\psi_1, \psi_3, \psi_5) = (\psi_1^*, \psi_3^*, \psi_5^*)\), the difference \(\Pr[\mathcal{G}_2] - \Pr[\mathcal{G}_3]\) is bounded by the probability that the mapping \(\phi : \{\psi_1, \psi_3, \psi_5\} \rightarrow (R_1, \ldots, R_{\text{token}})\) is injective. The probability that the mapping is injective is \(1/p\), since random values that appear in the equation are distributed uniformly over \(Z_p\) and independently.

**Lemma 1.4.** \(\Pr[\mathcal{G}_3] - \Pr[\mathcal{G}_4]\) is negligible.

**Proof.** Game 4 differs from Game 3 when the adversary queries the tracing oracle with a signature where \((\psi_1, \psi_3, \psi_4, \psi_5)\) does not contain a valid ciphertext tuple. If the adversary issues such a query to the tracing oracle, there should be the query \((M, \psi_1, \psi_3, \psi_5, R_1, \ldots, R_{\text{token}})\) in \(H\), and the hash value \(H(M, \psi_1, \psi_3, \psi_5, R_1, \ldots, R_{\text{token}})\) coincides with the unique challenge \(c\) that is determined from the problem instance \((\psi_1, \psi_3, \psi_5)\) and the commitment \((R_1, \ldots, R_{\text{token}})\). Hence for concluding the proof it is enough to bound the probability of this event. Notice that in this case any query \((M, \psi_1, \psi_3, \psi_5, R_1, \ldots, R_{\text{token}})\) to \(H\) in question is different from \((\psi_1^*, \psi_3^*, \psi_5^*, R_1, \ldots, R_{\text{token}})\) which is used for backpatching, the output of \(H\) is chosen from \(Z_p\) uniformly, and thus the probability that a query to \(H\) described as above exists with probability less than \(\varepsilon_p + q_{\text{trace}}/p\) in which \(q_H\) and \(q_{\text{trace}}\) respectively denote the upper bounds of the number of queries issued by the adversary to \(H\) and the opening oracle, therefore \(q_H + q_{\text{trace}}/p\) is negligible.

**Lemma 1.5.** \(\Pr[\mathcal{G}_4] = 1/2\)

**Proof.** In this lemma, we prove that the value \(u_1^* \cdot u_2^*\) and \(u_1^* \cdot u_2^*\) are uniformly random even when conditioned on the adversary’s view. To this end we examine the distribution of the adversary’s view related to the randomness \(Z_1, Z_2, Z_1, Z_2\) under the condition where all the other randomness involved in the game are fixed. The adversary obtains
information related $z_1, z_2, z'_1, z'_2$ from the part of the group public key $g_1g_1$ and the responses from the tracing oracle. As for the responses from the tracing oracle, any query whose $(ψ_1, ψ_3, ψ_4, ψ_5)$ components does not constitute the ciphertext tuple will be rejected by the tracing oracle, thus the adversary gains no information on $z_1, z_2, z'_1, z'_2$ from such queries. A query with a linear tuple also gives no information to the adversary. When the adversary issues a signature $c, s_α, s_β, s_γ, s_δ, s_ε, s_ν, ψ_1, ψ_2, ψ_3, ψ_4, ψ_5$, the tracing oracle computes group elements $u_1^z_1u_2^z_2$ and $u_1^{z'_1}u_2^{z'_2}$, which is what the adversary learns from this query. In fact does not increase the information the adversary knows, since the above equation can be rewritten as $u_1^z_1u_2^z_2 = u_1^{z_1z_2} = u^b$ and $u_1^{z'_1}u_2^{z'_2} = u_1^{z'_1z'_2} = g_1^m$, where $r = b(z'_1 + α·z_2)$ and $k(g_1 = g_1^m)$ is uniformly random when we write $ψ_3 = u_1 = g_1^m$, $ψ_4 = u_2 = g_1^b$. Even if the adversary asks the updating oracle for a special token token* at an epoch $t$, the right-hand side of the equation shows that the response of the tracing oracle gives no information to the adversary, since all the values that appears in the right-hand side are already known to the adversary.

The above discussion shows that the responses of the opening oracles do not leak any information on tracing key and the values $u_1^z_1u_2^z_2$ and $u_1^{z'_1}u_2^{z'_2}$ are distributed uniformly. This shows that the challenge signature is independent of gCert in and token in.

**Theorem 2.** The proposed scheme has non-frameability in the random oracle model under the DL assumption, where $H$ is modeled as a random oracle.

**Proof.** We first give an overview of the proof. S is given as input of a DL pair $g_1$, $y = g_1^x$, and $S$ creates a special OBU signer whose secret key $x = log g_1, y$, however, $S$ does not know the secret key $pk_{OBU}$. Then $S$ computes gCert and the attestation $σ$ using $(y, g_1)$ pair. In the output phase, if $A$ can output an attestation $σ'$ of the special signer, $S$ can solve the DL problem. Therefore, $S$ interacts with $A$ as follows.

**Setup.** $S$ generates public system parameters $(p, G_1, G_2, G_T, e, g_1, g_2, g_{1T}, g_{12}, H)$ as usual, chooses a random $x, y, σ \leftarrow Z_p$, then computes $(rkTM, pk_{TM}) = (γ, g_2^γ)$, $(tkY, u) = (ν, g_{2ν})$, $(sk_{GM}, pk_{GM}) = (ω, g_{2ω})$ and sets the global system parameters $gpk = (p, G_1, G_2, G_T, e, g_1, g_2, g_{1T}, g_{12}, g_{1T}, g_{12}, H, pk_{TM})$. All values are known to $A$.

**Hash Queries.** At any time, $A$ can query the hash function $H$. $S$ chooses hash value uniformly at random from $Z_p$ while ensuring consistency.

**Update Queries.** At the revocation epoch $t$, $A$ acts as TM to issue the updated revocation list. After updating the non-revoked token list, $S$ interacts with $A$ to obtain the last token. Actually, this oracle does not help $A$ to forge the target signature, since the token is independent of the group certificate and the $sk_{OBU}$.

**Register Queries.** In the registration stage, the OBU proves the knowledge of its secret key by using a zero-knowledge proof protocol between OBU and TM. Here, we give an implementation of the ZK protocol as below.

A zero-knowledge proof protocol between OBU and TM

1) OBU chooses at random $r_x \leftarrow Z_p$ and computes $R_x = g_1^{r_x}$.
2) OBU computes $c = H(pk_{OBU}∥R_x)$ and $s_x = r_x + c·x$.
3) OBU sends the proof $(pk_{OBU}, c, s_x)$ to the RSU.
4) TM checks $pk_{OBU}$ against the revocation list. If valid, compute $\tilde{R}_x = g_1^{s_x}pk_{OBU}$, otherwise abort.
5) TM verifies that $c = H(pk_{OBU}∥\tilde{R}_x)$. If the verification is successful, the proof is valid that denotes the OBU knows the knowledge of the security key $x$.

In this query, $A$ requests for creating a new OBU $i$. Let $q_*$ be the expected number of register requests from $A$. $S$ chooses a random $i^* \in R_1\{1, q_*\}$. There are two cases for $S$ to respond:

**Case of** $i = i^*$, $S$ forges the above protocol as follows: it sets $pk_{OBU} = y$ and picks $c_x, s_x \in R_1\{1, q_*\}$, and computes $R := g_1^{s_x}pk_{OBU}$. $S$ needs to patch the opening oracle by setting $H(pk_{OBU}∥R) = c_x$. If the backpatch is a failure, $S$ aborts and outputs the failure. Then $S$ runs the rest of the register protocol as the OBU with $A$ as TM, and obtains a public key certificate $cert_{OBU}$ on $y$, then stores $(i^*, cert_{OBU}, pk_{OBU}, tag_{i^*})$ in its log.

**Case of** $i \neq i^*$, $S$ selects $s_{OBU,i} \in R_1\{1, q_*\}$ and runs the Register protocol with $A$ as usual, and receives a property certificate $cert_{OBU,i}$ from $A$. $S$ stores $(i, cert_{OBU,i}, s_{OBU,i}, sk_{OBU,i}, pk_{OBU,i}, tag_{i})$ in its log.

**Join Queries.** $A$ (as group manager RSU) requests to run Join protocol for a new honest user $i$ in the group, $S$ behaves as follows.

**Case of** $i = i^*$, $S$ sends $(cert_{OBU}, pk_{OBU,i}, i^*)$. In subsequent steps of the Join protocol, $S$ proceeds as the real OBU $i^*$. When Join terminates, $S$ obtains a membership certificate $cert^* = (g_1g_1^{s_x}pk_{OBU}^i)/(sk_{GM}+γ^*)$, and restores into the log.

**Case of** $i \neq i^*$, $S$ follows the Join protocol as usual.

**Sign Queries.** $A$ queries a group signature at OBU $i$. If $i \neq i^*$, $S$ finds the corresponding secret key $sk_i$ and $(cert_{OBU,i}, token_{i}, ω, tag_{i})$ associated with OBU $i$, $S$ runs the Sign protocol, and return $σ_i$ to $A$. And if $i = i^*$, $S$ forges a signature as follows.

(a) $S$ takes out $(gCert, token_{i^*}, i)$ from its log, and computes $ψ_1', \ldots, ψ_5'$ as usual.
(b) $S$ chooses $γ_i, \tilde{s}_{OBU,i}, \tilde{s}_i, s_1', s_1', s_2', s_1, s_1, s_1, s_1' \in R_1\{1, q_*\}$, and computes:

$$\tilde{R}_{gCert} \leftarrow e(g_1, g_2)^{s_x}e(ψ_1, g_2)^{s_x}e(ψ_2, g_2)^{s_x}e(ψ_1, g_2)^{s_x}$$
$$e(u, pk_{GM})^{s_x}e(ψ_1, g_2)^{s_x}e(ψ_2, g_2)^{s_x}e(ψ_1, pk_{GM})^{s_x}$$
$$\tilde{R}_{token} \leftarrow e(g_1, g_2)^{s_x}e(ψ_1, g_2)^{s_x}e(ψ_2, g_2)^{s_x}e(ψ_1, g_2)^{s_x}$$
$$e(u, pk_{TM})^{s_x}e(ψ_1, g_2)^{s_x}e(ψ_2, g_2)^{s_x}e(ψ_1, pk_{TM})^{s_x}$$
The proposed scheme has traceability in the
Theorem 3.

Corruption Queries. A requests the secret key $sk_{OBU,i}$ at
OBU $i$. If $i \neq i^*$, $S$ responds with the key secret
corresponding to $i$. Otherwise, $S$ aborts and outputs the
failure.

Output. $A$ outputs a $\sigma' = (\psi_1^*, \ldots, \psi_5^*, c', s_0^*, \ldots, s_5^*)$, $S$ rewinds $A$ to extract the secret key $sk'$. By
using the forking lemma [22] to rewind $A$, $S$ controls the
result of hash $H$ by choosing two different $c'$ values $c'$, $c''$ to the same $(\psi_1^*, \ldots, \psi_5^*, R_{1}^*, \ldots, \tilde{R}_{\text{token}}^*)$. $A$ responds with
$(s_0^*, s_2^*, \ldots, s_5^*)$ and $(s_0^*, s_2^*, \ldots, s_5^*)$. Then $S$ computes $c'' - c'$ and extracts $sk'' = |s_2'' - s_2'|/|c'' - c'|$. If $sk'' \in \{sk_{OBU,i}\}$ for all $i$ except $i^*$ and $sk'' \neq y$, $S$ aborts and outputs
failure. Otherwise, if $\tilde{y}_1'' = y$, it means that $S$ obtains the
$OBU$ for all $i$ except $i^*$. $S$ aborts and outputs the
difference if $A$ wins the game with a non-negligible probability.
and \((s'_1, s'_2, \ldots, s'_s)\). \(S\) computes \(c = c' - c\), and extracts
\(s'k = |s'_1 - s'_i|/c, \eta' = |s'_1 - s'_i|/c, \) and \(g\)Cert' = \(ψ_1/w_{ik}^{t_{KM}}\) by using tracing key \(t_{KM} = t\). The gCert' must be outside the adversary coalition \(L_ε\). As mentioned in our security goals, there are two types of forgers to consider.

**Type-1 forger:** If \(n' \notin \{1, \ldots, n - 1, n\} \) \(\forall \) \(e' \notin \{1, \ldots, e - 1, e\}\), it means that \(A\) can forge a valid signature with an invalid pair \((n', e')\). Then \(S\) can obtain an extra valid modified-SDH tuple as follows:

\[
\begin{align*}
A' = &\text{gCert'} \cdot (g_1g'_1)^{s'k(b/a)} \cdot (g_1g_2)^{\nu} \cdot (g_1g_2)^{\eta'} = (g_1g'_1)^{\nu} \cdot (g_1g_2)^{\eta'}
\end{align*}
\]

**Type-2 forger:** If \(g\)Cert' \(\in \{\text{gCert}\}\) for all \(n\), and \(\eta' \notin \eta \land s'k' = a \land e' \notin \epsilon, S\) aborts and outputs failure with the probability of \(1/q_s\), otherwise \(\eta' = \eta \land s'k' = a \land e' = e, S\) can compute:

\[
\begin{align*}
A' = &\text{gCert'} \cdot (g_1g'_1)^{s'k(b/a)} \cdot (g_1g_2)^{\nu} \cdot (g_1g_2)^{\eta} = (g_1g'_1)^{\nu} \cdot (g_1g_2)^{\eta}
\end{align*}
\]

In either of the above two cases, \(S\) can solve the \(q\)-modified SDH problem with a non-negligible probability if \(A\) can win the game with a non-negligible probability. Thus, we can conclude that an adversary cannot forge a valid signature with invalid \((n', e')\) pairs or extra \((n, e)\) pairs, unless it can solve the \(q\)-modified SDH problem.

Now, we prove the \(q\)-modified SDH problem can be reduced to the \(q\)-SDH problem as below.

**Lemma 3.1.** If there is an adversary \(A\) who can solve the \(q\)-modified SDH problem, there exist a polynomial time simulator \(S\) can solve the \(q\)-SDH problem.

**Proof.** \(S\) is given \(G_1 = \langle \alpha_1 \rangle, G_2 = \langle \alpha_2 \rangle \) where \(\alpha_1 = \tau(\alpha_2)\), \(τ\) is a homomorphism from \(G_2\) onto \(G_1\), and \((q + 2)\)-tuple \(\langle u_1, u_2, u_3, u_4, \ldots, u_q \rangle \) \(\in G_1 \times G_2^{q+1}\) as inputs, \(S\) interacts with \(A\) as follows.

1) \(S\) selects \(\alpha, b \in Z_\phi\) and \(c_1, d_1 \in Z_\phi\) randomly, for \(i = 1, \ldots, q - 1\), so that \(c_1, d_1\) are different from each other. Then by using the \(q\)-SDH input tuples, \(S\) can set \(h_1 = \psi(h_2) = \psi(u_2) = u_2^\epsilon\) where \(Z = \alpha \prod_{i=1}^{q-1}(\gamma + c_1), g_1 = \psi(g_2) = \psi(g'_2) = h_1^\gamma, \Omega = h_2^\gamma\).

2) When \(A\) asks for a modified-SDH (MSDH) tuple with index \(i, S\) computes \(Y_i = \alpha \cdot (\beta + d) \prod_{j=1}^{q-1}(\gamma + c_1)\) and \(d_1 = \psi_1(Y_i)\), and forwards \((A_i, c_1, d_1)\) to \(A\). Because of \(A_i(\gamma + c_1) = h_1^{(\beta + d)\gamma}, A\) can have \(e(A_1, \Omega, h_2^\gamma) = e(A_1, h_2, h_2^\beta) = e(g_1, h_2^\gamma, h_2^\beta) = e(g_1, h_2, h_2^\beta) = e(h_1, h_2)\).

3) After \(q - 1\) queries, \(A\) outputs the extra modified SDH tuple \((A = (g_1 \cdot h_1^\gamma)^{\gamma+c_1}, c, d)\) with \((c, d) \neq (c_1, d_1)\) \((i = 1, \ldots, q - 1)\).

4) \(S\) then set \((A = (g_1 \cdot h_1^\gamma)^{\gamma+c_1} = h_1^{(\beta + d)\gamma}, \) and \(Z = (\beta + d)/(\gamma + c)\) can be transformed as follows:

\[
\begin{align*}
\alpha(\beta + d) &\prod_{i=1}^{q-1}(\gamma + c_1) \\
= &\alpha(\beta + d) \prod_{i=1}^{q-1}(\gamma + c) + (c_1 - c) \\
= &\alpha(\beta + d) \prod_{i=1}^{q-1}(c_1 - c) + (c_1 - c) \\
= &\alpha(\beta + d) \prod_{i=1}^{q-1}(c_1 - c) + (c_1 - c)
\end{align*}
\]

where \(f\) is polynomial of degree at most \(q - 2\), and \(f(\gamma + c)\) can be easily computed with the \(q\)-SDH input tuples. Then, \(S\) can get \(A = u_1^\gamma\cdot u_1^f(\gamma + c)\), where \(X = \alpha(\beta + d)\prod_{i=1}^{q-1}(c_1 - c)\).

In summary, \(S\) can compute \(A' = (A/u_1^f(\gamma + c))^1/X = \alpha(\beta + d)/(\gamma + c\cdot f(\gamma + c))\) as a solution to the \(q\)-SDH problem.

**Theorem 4.** The proposed scheme has unforgeability in the random oracle model under the \(q\)-SDH assumption, where \(H\) is modeled as a random oracle.

**Proof.** The attack on unforgeability is to fake a BBS+ signature as a member certificate or a non-revoked token. Therefore, the security against forgeable attacks can be simplified as the unforgeability of the BBS+ signature scheme, which was proved in [20]. Consider two types of forgers: (1) Forge certificates belonging to the current group. (2) Forge tokens of non-revoked users. Because of cracking the un-forgeability of the BBS+ signature scheme can help us construct an algorithm that breaks the \(q\)-SDH assumption, the theorem holds.

**VI. PERFORMANCE ANALYSIS**

Comparisons of security properties and the efficiency of our scheme to several other schemes are provided in this section.

**A. SECURITY ANALYSIS**

**TABLE 2** compares the security properties of our scheme to that of schemes introduced in [15][17][26][28]. According to **TABLE 2**, none of these counterparts protocols can satisfy all six security requirements. The schemes [17][26][28] did not consider the revocation mechanism, which is a very important secure requirement in VANETs. In [15], the scheme has a revocation mechanism based on public key certificates. However, in this scheme, if a user was revoked in a group, such as group A, the scheme cannot prevent the revoked user from broadcasting some illegal messages in group A, since the revoked user’s group certificate is still valid. Hence, the scheme does not satisfy the forward security. In our scheme, we construct a double revocation mechanism to prevent that:

- **Revocation List**: If a user is revoked, the TM will put its public key certificates into the RL. So that, when the revoked user enters a new group, the RSU will not issue a group certificate to it.

- **Non-revoked Token**: When a user is revoked, the TM will run the Revoke algorithm to update the non-revoked
token set. At a new revocation epoch, the revoked user is not able to obtain a new token; therefore, even if it still stays in original group, a valid signature will not be generated.

With the revocation mechanism, our scheme can ensure the forward security. According to the above analysis, our solution is more secure than existing schemes.

B. COMPUTATION ANALYSIS

Next, we analyze our scheme in terms of the computation overhead. We define and compute the time of the cryptographic operations required in our proposed scheme and the compared schemes. Let $T_{e,G_1}$, $T_{e,G_2}$, $T_{e,G_T}$ denotes the time to perform one exponentiation in $G_1$, $G_2$ and $G_T$, $T_{par}$ is the time to perform one pairing operation. Since these operations determine the speed of signature generation and verification, we only consider the influence of these four operations.

Firstly, we optimize the computation overhead of the Sign and Verify algorithms.

- **Sign algorithm**
  - In Formula (1), by precomputing $e(\hat{g}_1, g_2)$, $e(\hat{g}_1, g_2)$, $e(u, g_2)$ and $e(u, p_{\text{PK}_{GM}})$, the computation overhead of Formula (1) contains one $T_{par}$ operation, one $T_{e,G_1}$ operation and four $T_{e,G_T}$ operations.
  - In Formula (2), by precomputing $e(\hat{g}_1, g_2)$, $e(\hat{g}_1, g_2)$, $e(g_1, g_2)$ and $e(g_1, p_{\text{PK}_{GM}})$, the computation overhead of Formula (2) also contains one $T_{par}$ operation, one $T_{e,G_1}$ operation and four $T_{e,G_T}$ operations.

- **Verify algorithm**
  - The Formula (3) can be transformed as follows: $e'(\hat{g}_1, g_2) = e(\psi_1, g_1)^{-s_\psi} e(g_1, g_2)^{-s_\gamma} e(\hat{g}_1, g_2)^{-s_\gamma} e(u, g_2)^{s_\eta} \cdot e(u, p_{\text{PK}_{GM}})^{s_\gamma} (e(g_1, g_2)/e(\psi_1, p_{\text{PK}_{GM}}))^{s_\gamma}$
    - In theory $1T_{e,G_1}$
    - In simulation $0.031ms$
  - The computation overhead of Formula (3) contains one $T_{par}$ operation, two $T_{e,G_1}$ operation and five $T_{e,G_T}$ operations.
  - The Formula (4) also can be transformed as the Formula (3), and have the same computation overhead.

The proposed protocol was implemented using the PBC library [23] with type-D curve which implements the Tate pairing on an MNT curve [36]. It is simulated on a workstation with specifications listed in TABLE 3. TABLE 4–6 present the detailed computation overhead.

<table>
<thead>
<tr>
<th>TABLE 2: Comparisons of security properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Properties</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>Anonymity</td>
</tr>
<tr>
<td>CCA2-Anonymity</td>
</tr>
<tr>
<td>Traceability</td>
</tr>
<tr>
<td>Non-frameability</td>
</tr>
<tr>
<td>Unforgeability</td>
</tr>
<tr>
<td>Forward security</td>
</tr>
<tr>
<td>Revocation*</td>
</tr>
</tbody>
</table>

*: The requirement is satisfied. ✗: The requirement is not satisfied. N/A: It does not have such property. *: It has revocation mechanism.

<table>
<thead>
<tr>
<th>TABLE 3: Configurations of the simulation platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor</td>
</tr>
<tr>
<td>Operating frequency</td>
</tr>
<tr>
<td>Memory capacity</td>
</tr>
<tr>
<td>OS</td>
</tr>
<tr>
<td>Programming environment</td>
</tr>
<tr>
<td>Crypto-library</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE 4: Computation overhead on the TM</th>
</tr>
</thead>
<tbody>
<tr>
<td>TM</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>In theory</td>
</tr>
<tr>
<td>In simulation</td>
</tr>
</tbody>
</table>

$m$: the number of non-revoked users

<table>
<thead>
<tr>
<th>TABLE 5: Computation overhead on the RSU</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSU</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>In theory</td>
</tr>
<tr>
<td>In simulation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE 6: Computation overhead on the OBU</th>
</tr>
</thead>
<tbody>
<tr>
<td>OBU</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>In theory</td>
</tr>
<tr>
<td>In simulation</td>
</tr>
</tbody>
</table>

In addition, the OBU proposed in our scheme can verify the validity of the signature without downloading the revocation list, and the verification cost remains unchanged with the revocation of the user.

Furthermore, a comparative summary for the computation overhead is presented in TABLE 7. In this table, we only compare with the schemes based on group signatures.

Our scheme is less efficient than the other schemes in the Sign algorithm, but the Verify algorithm from us is efficient; moreover, our scheme is more secure than other schemes.

C. ROBUSTNESS ANALYSIS

In this subsection, we evaluate the robustness of the proposed scheme under VANETs. In our proposal, RSUs periodically broadcast public key certificates from themselves and their neighbors while also providing the following two services for vehicles within their range:

1) Executing the Join protocol to issue the group certificate to vehicles.
2) Executing the Update protocol to help vehicles renew the non-revoked token.
TABLE 7: Comparisons of computation overheads

<table>
<thead>
<tr>
<th>Schemes</th>
<th>Sign Algorithm</th>
<th>Verify Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shao et al. [13]</td>
<td>$12T_{e,G_1} \approx 0.38\text{ms}$</td>
<td>$10T_{\text{par}} + 4T_{e,G_1} \approx 15.83\text{ms}$</td>
</tr>
<tr>
<td>Gao et al. [28]</td>
<td>$5T_{e,G_1} \approx 0.16\text{ms}$</td>
<td>$4T_{\text{par}} + 6T_{e,G_1} \approx 0.47\text{ms}$</td>
</tr>
<tr>
<td>Proposed solution</td>
<td>$2T_{\text{par}} + 5T_{e,G_1} + 8T_{e,G_T} \approx 5.78\text{ms}$</td>
<td>$2T_{\text{par}} + 6T_{e,G_1} + 4T_{e,G_2} + 8T_{e,G_T} \approx 6.23\text{ms}$</td>
</tr>
</tbody>
</table>

When a vehicle passes by an RSU, if this is the first time it communicates with the RSU or its non-revoked token is expired, then the RSU will execute the above two protocols for this vehicle. After receiving the group member certificate or non-revoked token, vehicles can anonymously sign some important messages during its stay within the range of the RSU. These signed messages can be verified by other vehicles in areas that are covered by the current and neighboring RSUs. This local processing of messages by the RSUs within the range results in increased efficiency and robustness.

As for robustness, if an RSU, for some reason such as hardware problems, high vehicle density, high mobility of vehicles, high computation overhead or limited bandwidth, cannot provide services to vehicles within its range, there are still some effective solutions to sustain our proposal.

VII. CONCLUSIONS

In this paper, to solve the problem of missing secure properties and low efficiency of anonymous authentication for privacy protection in VANETs, an efficient and secure anonymous authentication protocol is proposed based on the framework of group signatures. The proposed anonymous authentication does not only have efficient verification and revocation features, but also has forward security, CCA2-anonymous, non-frameablility, unforgeability and traceability, which cannot be completely satisfied in existing schemes. And this scheme adopts the decentralized group model to release the trusted authority from the heavy work burden from generating group certificates for OBUs, and to free the OBU from retrieving the revocation list from the trusted authority.

FIGURE 5: The organization of a VBN system.

VANETs is a subclass of MANETs. The organization of a VBN is shown in Fig. 5.

In VBN, particular vehicles are dynamically elected as Relay Nodes (RN) to act as base stations. These relay nodes are used as backbone forwarding nodes for a specific period of time. In case if a relay node detects a path or link failure (loss of signals), then the relay node can choose an alternate path with a different set of intermediate nodes for emergency warning message dissemination [42] [43]. A vehicle is permitted to act as a relay node when it is within a specific distance limit from the road side unit. Once a vehicle goes away from the coverage of road side unit, a new vehicle closer to RSU will be reelected as RN. The RN selection procedures are explained in [41] [45].

With the method mentioned above, we can extend the coverage and load capacity of RSU A and RSU C so that the non-revoked token can still be obtained even if RSU B collapsed. As a consequence, the robustness of our scheme is can also be ensured through the idea of VBN. On the other hand, if the robustness of our scheme is affected by the computation overhead, hardware accelerators [48] [49] can be employed as a relief.
In the future, a prototype of the proposed solution for real word evaluation is going to be implemented. This will allow us to evaluate and re-define the proposed scheme, in order to be more practical and efficient.

REFERENCES


XIAOHAN YUE received B.E. in electronic information and technology, M.E. and Ph.D. in computer application technology from Northeastern University, China, in 2005, 2009, and 2013, respectively. He joined the School of Information Science and Engineering, Shenyang University of Technology, in 2013, as a Lecturer. He authored over 40 research publications. His primary research interests are trusted computing, wireless mesh network security, security and privacy in ubiquitous computing, and cryptography. He is a member of the CCF.

BING CHEN received B.E. in computer science and technology from Binzhou College, China, in 2016. In the same year, he started his graduate study at the School of Information Science and Engineering, Shenyang University of Technology. His research direction is information security.

MINGCHAO GAO is an Associate Research Fellow in Jilin Academy of Agricultural Sciences. After graduating with his Ph.D. from Shenyang Agricultural University, he mainly engages in the study of digital agriculture and agricultural internet of things, including directing over 20 scientific research projects, publishing more than 20 articles, and winning 4 provincial awards. He is mainly interested in agricultural applications of internet of things and satellite highlight spectrum.

YUAN HE (SM’07–M’14) is an Associate Professor at the School of Information Science and Engineering, Shenyang University of Technology. He received his Ph.D. from The University of Tokyo in 2014, ME(Hons) and BSc from the University of Auckland in 2009 and 2005, respectively. His research interests include computer architecture, energy-efficient computing at all scales and post-Moore performance scaling. He is a member of the ACM, IEEE and IEICE.