CreditCoin: A Privacy-Preserving Blockchain-based Incentive Announcement Network for Communications of Smart Vehicles

Lun Li, Jiqiang Liu, Lichen Cheng, Shuo Qiu, Wei Wang*, Xiangliang Zhang and Zonghua Zhang

***Abstract*—The vehicular announcement network is one of the most promising utilities in the communications of smart vehicles and in the smart transportation systems. In general, there are two major issues in building an effective vehicular announcement network. First, it is difficult to forward reliable announcements without revealing users' identities. Second, users usually lack the motivation to forward announcements. In this work, we endeavor to resolve these two issues through proposing an effective announcement network called *CreditCoin*, a novel privacy-preserving incentive announcement network based on Blockchain via an efficient anonymous vehicular announcement aggregation protocol. On the one hand, *CreditCoin* allows non-deterministic different signers (i.e., users) to generate the signatures and to send announcements anonymously in the non-fully trusted environment. On the other hand, with Blockchain, *CreditCoin* motivates users with incentives to share traffic information. In addition, transactions and account information in *CreditCoin* are tamper-resistant. *CreditCoin* also achieves conditional privacy since Trace manager (TM) in *CreditCoin* traces malicious users' identities in anonymous announcements with related transactions. *CreditCoin* thus is able to motivate users to forward announcements anonymously and reliably. Extensive experimental results show that *CreditCoin* is efficient and practical in simulations of smart transportation.**

***Index Terms* — Smart transportation, Blockchain, vehicular communication, incentive mechanism, threshold authentication, privacy**

I. Introduction

SMART cities have drawn much attention due to the rapid growth of urbanization and the resulting pollution from traffic, in both academia and industry. Vehicular announcement networks in VANETs (Vehicular *ad hoc* networks) have become one of the most promising vehicular communication applications, as it leads to a much safer vehicle-driving experience. Additionally, it is also eco-friendly while decreasing the expenditure of many public resources by reducing the frequency of traffic jams and accidents.

Blockchain is a novel decentralized ledger-based storage method. Satoshi firstly applied Blockchain into *Bitcoin* [3], which is a peer to peer e-cash system. Later, Blockchain gets more and more attention in e-commerce. Particularly, it has become a hot topic since Blockchain-based *Bitcoin* became popular. Moreover, in Blockchain-based networks, each node manages a copy of the whole or part of a database from the system. Thus, Blockchain-based networks are promising in recording credit data with the good properties of tamper-resistance and decentralization, which is useful in VANETs.

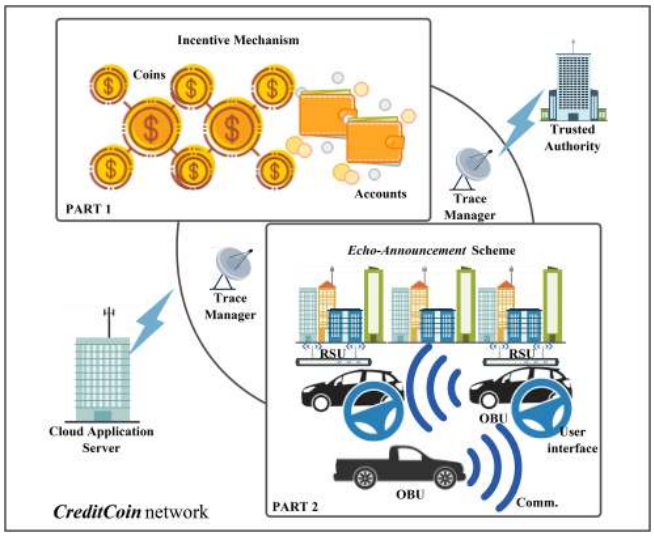


Fig. 1. The general idea of ***CreditCoin***.

With the increasing privacy concerns of data [4-7], there exist two major issues in building an effective vehicular announcement network. First, ideally, all messages must be forwarded anonymously in VANETs since they usually contain sensitive information of users, such as vehicle numbers, driving preferences and customer identities. However, forwarding messages anonymously does not assure the reliability of the messages, thus decreasing the credit of vehicular announcements.

Second, users usually lack enthusiasm to forward any messages in VANETs if there is a risk that their privacy will be breached. In addition, users do not benefit from forwarding announcements, which also makes them lack motivation to

* Corresponding author
This work was supported in part by National Natural Science Foundation of China, under Grant 61672092.
L. Li, J. Liu, L. Cheng, S. Qiu and W. Wang are with Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing 100044, China (e-mail: Lun.Li@bjtu.edu.cn; jqliu@bjtu.edu.cn; 16112090@bjtu.edu.cn; qiushuo@bjtu.edu.cn; wangwei1@bjtu.edu.cn).
X. Zhang is with the King Abdullah University of Science and Technology, Thuwal 23955-6900, Saudi Arabia (e-mail: xiangliang.zhang@kaust.edu.sa).
Z. Zhang is with IMT Lille Douai, Institut Mines-Telecom, France (e-mail: zonghua.zhang@imt-lille-douai.fr).

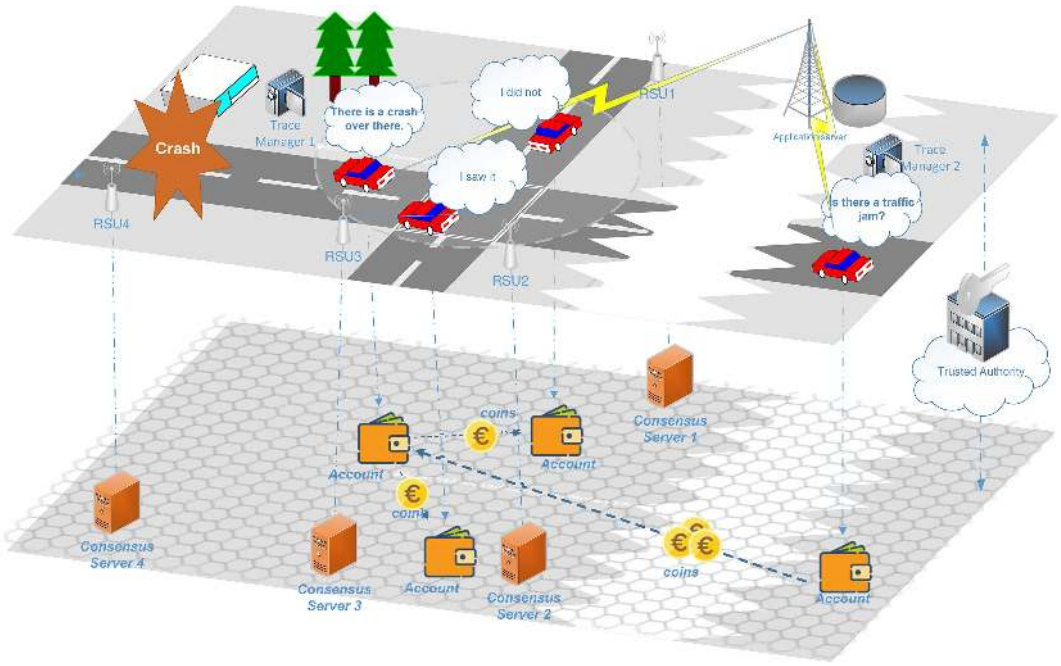


Fig. 2. Overview of ***CreditCoin***.

respond to messages. For instance, we consider the following example:

Example 1: Alice would like to go to an unfamiliar place, and she needs traffic information from the destination without revealing her privacy. Meanwhile, Bob is a resident. He finds an accident on the road. Now, Bob wants to inform other drivers with an announcement of the accident and hopes to get some rewards. However, for other drivers, such as Alice, only one witness's message is unable to make the announcement reliable. Therefore, Bob contributes some rewards as incentives to encourage other witnesses to forward the same announcement together.

Existing work [1, 2, 8, 9] leveraged threshold authentication and group signatures to tackle the challenges we stated in Example 1. However, these solutions suffer from having a heavy workload and lack incentives to forward messages.

In this work, in order to resolve these two issues, we build an effective network for communication of smart vehicles. In particular, we propose a novel privacy-preserving incentive announcement network based on Blockchain, named ***CreditCoin***, which contains **two parts**, the announcement protocol and incentive mechanism as illustrated in Fig. 1.

In summary, we make the following contributions:

- To the best of our knowledge, ***CreditCoin*** is the first privacy-preserving Blockchain-based incentive network in VANETs. It is able to build trust in communications of smart vehicles.
- We propose a vehicular announcement protocol *Echo-Announcement* in ***CreditCoin***. It achieves efficiency and privacy-preserving for the practical usage in forwarding announcements.
- We design an incentive mechanism based on Blockchain in ***CreditCoin***. Users manage reputation points while they earn or spend ***coins*** as incentives. Meanwhile, ***CreditCoin*** still preserves privacy and achieves anonymity. Moreover, based on Blockchain, ***CreditCoin*** prevents many security attacks and achieves conditional privacy because Trace manager will trace malicious nodes when an unexpected event occurs.
- We implement ***CreditCoin*** systematically in the simulation of smart transportation in *Network Simulator 2* and *Java Runtime Environment 1.8*. The test results show that ***CreditCoin*** is efficient and practical in the simulations of the smart transportation and smart vehicles.

The remainder of this paper is organized as follows. Section II describes the overview and design goals of ***CreditCoin***. Section III introduces the related work. Section IV reviews preliminaries used in this paper. Section V presents the announcement protocol *Echo-Announcement* and its security analysis in ***CreditCoin***. Section VI defines the incentive mechanism in ***CreditCoin*** and its security analysis and then illustrates that how ***CreditCoin*** works effectively with incentive mechanism based on Blockchain. Section VII provides the detailed analysis of our performance and simulation results. Section VIII concludes this paper.

II. Overview and Design Goals

A. Overview of CreditCoin

Our system consists of five entities: the Trusted authority (i.e., TA), the Trace manager (i.e., TM), users (i.e., On-board units (OBUs)), RSUs (Roadside units), and a cloud application server as shown in Fig. 2. *1) Trusted authority* is responsible for managing the key, generating public parameters and managing the users' identities; *2) Trace manager* (i.e., TM) is a role that traces malicious users. TMs are distributed in

various regions for tracing malicious users rapidly in the given region. We comprehensively analyzed the tracing process in Section V-D-§5; *3) Users* are general vehicles. In ***CreditCoin***, Users are the main components of network services to send messages or transactions; *4) RSUs* are distributed along the road and designed to manage a group of OBUs within their communication range. In particular, in ***CreditCoin***, the RSU also participates in the consensus voting algorithm when constructing the block chain; *5) Cloud application server* stores and exchanges some non-cryptographic information in the network.

In order to build an effective vehicular announcement network, there are two parts in ***CreditCoin***. The first part is announcement protocol, namely *Echo-Announcement*. This protocol provides threshold authentication and a certain privacy level to guarantee that anonymous announcements are reliable in ***CreditCoin***. Users set their roles as follows: An *Initiator* invites other witnesses as *Repliers* to agree with his/her announcement with corresponding signatures and generates an announcement with traffic information and responses signed by *Repliers*. Since there is a larger group of users concealing all of the participants in the protocol, the receivers of the announcement knows the number of participants but cannot figure out their identities.

The second part is Blockchain-based incentive mechanism that works together with *Echo-Announcement*. Every user in ***CreditCoin*** owns a credit account at several addresses. The account contains reputation points called the ***coins***. Users reward traffic announcements from a certain area by paying some ***coins*** as incentives. They can also spend some ***coins*** to make an announcement for hunting others' reward missions. Thus, in ***CreditCoin***, a user gets a small amount of ***coins*** from replying to the aggregation request for an announcement of others. Meanwhile, he/she also has a chance to hunt a large amount of ***coins*** by making an announcement to someone in particular, as someone else needs it.

In ***CreditCoin***, the traffic missions are managed by a cloud application server, and the transactions among users are forwarded based on Blockchain. After constructing transactions, users forward the transactions to RSUs nearby, and then RSUs vote the validity of transactions. Later, the valid transactions are confirmed by the consensus server. Finally, the valid transactions are added to the blocks on the chain.

B. Design Goals

The goal of our work is to design an effective vehicular announcement network for VANETs. Based on the proposed incentive mechanism and *Echo-Announcement*, ***CreditCoin*** has the following properties:

Enthusiasm: ***CreditCoin*** motivates users with incentives to share traffic information via announcements. It is the vehicular *incentive* announcement network in VANETs.

Privacy: The requests, announcements, and the transactions do not leak any information about their sources (*anonymity*). Two messages in ***CreditCoin*** cannot be linked to the same sources (*unlinkability*). Only the *TM* reveals the user's identity when a un-expectancy occurs (*traceability in conditional privacy*).

Reliability: The announcements are signed by several honest witnesses (*truthfulness*). According to threshold authentication and Blockchain, every user could manage a copy of the whole block chains of transactions, and each transaction is related to the phases of announcement aggregation. Therefore, a source is unable to deny sending messages (*non-reputation*). Additionally, announcements and transactions cannot be modified without authorization (*tamper-resistance*).

III. Related Work

The existing work consists of threshold authentication that is related to our proposed announcement protocol *Echo-Announcement* and Credit network that is related to our proposed incentive mechanism.

A. Threshold Authentication

Threshold authentication [10] is a standard method to prove messages reliability in VANETs. In general, the vehicles in VANETs communicate with each other in the non-fully-trusted environment. In threshold authentication protocols, the receiver only accepts a message when the message is confirmed by the threshold number of vehicles in VANETs. Thus, messages aggregation in VANETs is an effective way to realize threshold authentication and reduce the network overhead. Some existing work proposed different types of communication protocols in VANETs without considering users' privacy.

With the increasing privacy concerns in VANETs, since the messages should be forwarded anonymously in VANETs, several attacks[11] [12-15] (e.g., the *Sybil* attack) have drawn a lot of attention. These attacks lead to a trade-off between users' privacy and message's reliability. Thus, some issues of privacy, such as anonymity, reliability, linkability (i.e., two signatures on the same message by one signer could be linkable) and traceability have become the main topics to be studied. Kounga et al. [16] proposed a secure hardware mechanism to control the generation of pseudonyms for preventing *Sybil* attacks. Wu et al. [17] used one-time authentication and message-linkable group signatures to identify malicious users. However, the trace phase requires expensive pairing operations so that it is inefficient to trace doubtable messages. Chen et al. [1] proposed a threshold anonymous announcement (i.e., TA-Announcement) scheme with direct anonymous attestation and one-time anonymous authentication. In their scheme, the credentials of the malicious users cannot be revoked efficiently, and thus frequent attacks from malicious vehicles would decrease the efficiency of the scheme. Qin et al. [18] adopted a secure RSU management to achieve pseudonyms control, and Xia et al. [19] proposed a protocol on forwarding adaptive multimedia data with attribute-based encryption. However, it is an arduous task to design these protocols [18, 19] so that the hundreds of original messages are concealed or encrypted by RSUs since the RSU is assumed as a light-hardware with a low-security level in traditional VANETs. Zhang et al. [20] leveraged

group signature to solve linkability in vehicular announcement networks in VANETs. However, a group of users in this scheme shares the same private key, which is considered unsafe in a privacy scheme. Later, Lin et al. [21] proposed an RSU-aided protocol, which reduced the impact of malicious users and supports the local detection with efficient traceability of malicious nodes. Unfortunately, it is unable to work effectively in areas with sparse RSUs, and the utilization of trusted RSUs cannot bootstrap such a system. In the most recent related work, Shao et al. [2] proposed a new threshold anonymous authentication protocol (i.e., TA-authentication) based on a decentralized group model. It realized anonymity, traceability and solved the problem of message linkability simultaneously via increasing the complexity of the decentralized group. Azees et al. [22] proposed an efficient anonymous authentication scheme with an efficient tracing method. However, without incentives, it still suffers from the enthusiasm issue while forwarding messages.

B. Credit Network and Blockchain

Credit network [23] [24] is a common method to describe the credit relations among users in the network. In Credit networks, each node has points related to reputation, and it is easy to identify whether a node is honest or malicious by judging the reputation points. Therefore, it is widely applied [25, 26] in the digital currency of decentralized networks and *Sybil*-tolerant systems. Recently, Kate et al. [27] considered building a Blockchain-based Credit network in anonymous and *Sybil*-tolerant networks. Blockchain is currently widely studied on cryptocurrency in recent years. Nakamoto [3] proposed *Bitcoin*, which is a decentralized cryptocurrency based on Blockchain. *Bitcoin* is popular and claimed as a kind of anonymity currency. However, due to the property of de-centralization, it can obtain the relations between different addresses by tracing a series of transactions. Therefore, there exists related work focusing on studying Blockchain-based networks in a privacy-preserving manner, such as *Zerocash* [28] and extended *Zerocash* [29]. Particularly, in [29], the coins can be traced selectively via a public key encryption scheme in the output of coins. However, the transactions cannot be traced in the network, which is considered not reliable in a Credit network.

Additionally, after transactions are constructed, the verification of the valid transactions in the agreement is also necessary for Blockchain-based networks. To solve this problem, currently, *Proof of work* [3] and *Consensus Algorithm* [30] are two agreement algorithms used in Blockchain. *Consensus Algorithm* is developed from the notion of interactive consistency [31] and *Byzantine General Problem* [32]. We design our own consensus algorithm in ***CreditCoin***.

C. Our Solution

Having the risk of privacy leaking without benefits, users usually lack the enthusiasm to respond [12], which makes the previous vehicular announcement protocols somewhat impractical. In ***CreditCoin,*** we firstly design a privacy-preserving vehicular announcement protocol on a Blockchain-based network. It maintains the reliability and anonymity of the messages simultaneously. Then, Our ***CreditCoin*** meets the requirement of incentives in vehicular announcement protocol. It increases the users' enthusiasm and achieves reliability and anonymity simultaneously without leaking any extra private information. To the best of our knowledge, ***CreditCoin****,* as an interdisciplinary work, representing the novel work realizes forwarding announcements with incentives in a Credit vehicular network based on Blockchain.

IV. Preliminaries

A. Threshold Ring Signature

The cryptography on the threshold was firstly proposed by Shamir et al. [33]. In threshold sharing protocol, several people want to share and regain a secret only if there are more than t people. In a traditional scenario, users are known to each other in the first instance. Thus, it is not suitable in the non-fully-trusted environment. Bresson et al. [8] proposed the signature of threshold ring. A ring signature is a message authentication method with the threshold t and the member size r. It is a message-sharing protocol that a message is not valid unless at least t participants sign it. Meanwhile, the identities of the real signers are still covered in r people. However, in this traditional protocol, the identities of members are fixed. Moreover, when we generate a $(t, r) - threshold$ signature σ, all of private keys of members are needed, as they suppose that the members trust each other from the beginning.

Therefore, existing work [20, 34, 35] does not satisfy the privacy requirements of VANETs due to the above problems above. In the scenarios of VANETs, firstly, the users are non-deterministic. We do not know who will join in advance. Secondly, users are not in trust for a long period. Thus, they cannot share private keys with each other. Finally, the algorithm must be efficient as an *ad hoc* network changes frequently. Thus, these problems must be solved in the announcement network in VANETs.

B. Combined-Public Keys (CPK)

The idea of Combined-Public Keys was developed from the IBC [36] (Identity-Based Cryptography). It was proposed [37] as a method of key management and identity authentication at first. Liu et al. [38] improved CPK to construct cryptographic preliminaries implementing IBC for efficiency since it relieves the burdens of key management apartments such as Trusted authority. Zhang et al. [39] designed an encryption scheme based on ECC (i.e., Elliptic Curves Cryptography) and CPK. In this paper, we leverage CPK to simplify certificates and reduce the cryptographic time consumptions of the ring signature.

C. Merkle Hash Tree

Tree Signature is widely used in public key cryptosystems [40]. Merkle et al. [41] firstly proposed *Tree Signature* as a digital signature authentication. Because of the lower storage cost and the efficient verification, *Merkle Hash Tree* was used in the construction of cryptography in [3], in order to

create the hash index of transaction records and the verification of the block's locations on Blockchain. *Merkle Hash Tree* is a binary tree, and each leaf node is related to a fixed hash value calculated from a small fixed fragment. In other words, each leaf node represents a unique and fixed fragment. The union set of all fragments is made up by raw data waiting to be verified. The hash value of a parent node is computed by the hash value of its child node. As the raw data and the fragment are fixed, the root of the hash tree is also fixed. Thus, the verification of a fragment is the proof of the existence of a leaf node. This is proceeded by finding a path from the fragment to the root.

D. Byzantine Faults Tolerate Algorithm

Byzantine Faults Tolerate Algorithm is used for reaching agreements in the presence of faults by repeating several rounds of voting. Pease et al. [31] discussed the probability of reaching agreements in the presence of faults. They proposed the notion of interactive consistency in fault tolerant system to resolve the issue on inconsistency numbers of faults in a system. They proved that it was impossible to reach an expected agreement when faulty nodes were not fewer than one-third of all nodes. Later, they discussed the issues again in detail using the description of Byzantine Generals Problem [32]. Byzantine Generals Problem is described as the process of planning to attack an enemy city: Before attacking, a group of generals in each portion of the army should decide upon an identical plan of action. Meanwhile, a small number of traitorous generals among them try to disturb the decision by spreading fraudulent votes. For instance, if there are nine generals voting in the system. Four of them support attacking while others support retreat. The ninth general may send a vote of retreat to those generals in favor of retreat, and a vote of attack to the rest. Thus, those who receive a retreat vote from the ninth general will retreat, while the rest will attack. This problem will be even more complicated if the generals are physically separated and vote via messengers since they may fail to deliver votes or may forge false votes to each other. In a Byzantine Fault Tolerant System, the case that servers make an unexpected decision for any kind of reasons is called a Byzantine Fault. A server with at least one Byzantine Fault is called a Byzantine server.

V. Proposed Announcement Protocol

In this section, we propose an announcement protocol called *Echo-Announcement*. Although threshold authentication is a common method to send messages in the network [9], according to Section I, our protocol is under a non-fully-trusted environment. Thus, the signers who generate the signatures in *Echo-Announcement* are non-deterministic.

A. Basic Idea

Echo-Announcement focuses on the application of vehicular announcements in VANETs. Firstly, we assume that the number of malicious vehicles is relatively small than the number of honest vehicles. Secondly, the areas in VANETs are large with a long distance of disseminated messages. Finally, there is a black-box in each OBU to store vital information independently such as keys and other security accessories. The cryptographic computations and system parameters should be kept in the black box to prevent tampering. These assumptions are widely accepted in most of the privacy-preserving protocols in VANETs. Then, we design *Echo-Announcement* as the following example:

TABLE I
NOTATIONS

Param.	Explanation	Param.	Explanation
$\mathcal{T}$	Trusted authority	t	Threshold of announcement
$\mathcal{R}$	Replier	H	Hash functions
$\mathcal{I}$	Initiator	f	A polynomial over $\mathrm{GF}(2^l)$
$\mathcal{V}$	Verifier	r	Ring size of signatures
RQP	Request Packet	S	The set of IDs
RPP	Reply Packet	γ	Random Index for a user
AGP	Aggregated Packet	E_k	A symmetric encryption scheme with key k
$\mathbb{G}/\ q$	An addition group/ the order of group.	k, sk, pk	Symmetric/private/public keys
$\mathcal{X}, \mathcal{Y}$	Master key vectors of sk / pk	r	The size of the ring
msg	Messages in announcement		

Algorithm 1: Request Reply

```
Input: RQPArray P, LastRQP L
Sort(P,TIME);    //Sort RQPs about the same event by time
for i=0; i<P. length; i++ do
  if P[i].threshold > L.threshold then
      Reply(P[i]);
      L = P[i]; //Record the last
  end
  else
      P[i].delete(); //Ignore packet P[i]
  end
```

Example 2: Bob witnesses an accident and would like to let other drivers know by sending an announcement. To make such an announcement message trustworthy, Bob needs to cooperate with other witnesses. To do that, Bob firstly initiates a request to the surrounding witnesses for confirming his announcement message. After getting $t-1$ replies, Bob forwards the announcement with t confirmations (including himself) to other drivers heading to this place. Suppose Alice receives the announcement, who then checks its validity and re-plans the travailing route.

B. Settings

Roles Settings: The Trusted authority is $\mathcal{T}$, the *Initiator* is $\mathcal{I}$, the *Replier* is $\mathcal{R}$, and the *Verifier* is $\mathcal{V}$ shown in Table 1.

Packet Settings: To accomplish Echo-Announcement, three types of packets are generated by the vehicles, depending on their roles.

a) **Request Packet** (**RQP**) is a type of packets that are

from an *Initiator* to a group of witnesses. The purpose is to ask the witnesses to agree on the announcement and sign it. In particular, RQP contains three information: the message reports, the threshold value t and the large group of r value.

b) **Reply Packet** (**RPP**) is a type of packets from *Repliers* to the *Initiator*. If a witness is willing to join the announcement, the identity information for the generation of the ring will be sent back to *Initiator*. Particularly, if most of the RPPs are sent to one witness (*Replier*), the witness replies as Algorithm 1 to avoid congestion.

c) **Announcements-Aggregated Packet (AGP)** is a type of packets sent from *Initiator* to other *Verifiers*. An AGP includes an announcement and a threshold ring signature of this announcement.

C. Description

There are five phases in our *Echo-Announcement*:

a) **Setup:** $\mathcal{T}$ generates some public parameters to all. Then, $\mathcal{T}$ generates keys for users in *Echo-Announcement*.

b) **Request:** A user finds an accident and becomes $\mathcal{I}$ with his/her willingness. Then, $\mathcal{I}$ selects some parameters for the announcement and forwards RQPs to other witnesses for inviting them to join the announcement.

c) **Reply:** A witness will forward an RPP back to $\mathcal{I}$, becoming $\mathcal{R}$, if he/she agrees with the announcement of $\mathcal{I}$. This RPP includes a fraction of the ring signature.

d) **Announcement:** $\mathcal{I}$ forwards the AGP to others after $\mathcal{I}$ receives more than threshold t RPPs.

e) **Verification:** Any user receiving the AGP can become $\mathcal{V}$ to verify the validity of the AGP.

Note that, let $x \xleftarrow{R} X$ denote selecting element x from the set X at random. We also explain parameters in Table 1. The construction of *Echo-Announcement* is as follows:

1) Setup

Let $\mathbb{G}$ be an addition group of points on an elliptic curve. Let $q \in \mathbb{G}$ be an order of $\mathbb{G}$. P is the generator of $\mathbb{G}$. Let $E_k(x)$ be a symmetric encryption protocol using secret key k to decrypt x. Let H be a hash function.

$\mathcal{T}$ does the following steps:

a) Generates $(x_1, \dots, x_n) \xleftarrow{R} \mathbb{Z}_q^*$, and computes $\mathcal{Y}_i = x_i \cdot P$, where $x_i \epsilon (x_1, \dots, x_n)$.

b) Selects $H_0 : \{0,1\}^* \to \{0,1\}^n, H_1 : \mathbb{G} \to \mathbb{Z}_q, H_2\ H_3 : \{0,1\}^* \to \{0,1\}^l$.

c) Chooses $E \leftarrow \mathrm{GF}(2^l)$.

d) Defines $\mathcal{X} = (x_1,\ x_2, \dots, x_n)$ as the master private key vector.

e) Defines $\mathcal{Y} = (\mathcal{Y}_1,\ \mathcal{Y}_2,\ \mathcal{Y}_3, \dots, \mathcal{Y}_n)$ as the master public key vector.

f) Sets the system public parameters to be $(\mathbb{G}, q, \mathcal{Y}, H, E_k)$. Makes them public.

The private keys of the users are also managed by $\mathcal{T}$. The identities ID of the users are the *VIN* (Vehicle Identification Number) of the vehicles. For every user with identity ID, let the keys be:

- $sk_{ID} = \sum_{i=1}^{n} h_i x_i \bmod q$
- $pk_{ID} = \sum_{i=1}^{n} h_i \mathcal{Y}_i$

where h_i is the ith bit of $H_0(ID),\ i = 1, \dots, n$.

2) Request

$\mathcal{I}$ does the following steps:

a) Produces an accident description msg according to the specific context.

b) Chooses values t and r. Their values play import roles in the protocol, and we will discuss them in Section VII.

c) Selects $(ID_1, ID_2, \dots, ID_{r-t}) \xleftarrow{R} ID$ list sets, Let $\overline{\mathrm{S}} = \{ID_1, ID_2, \dots, ID_{r-t}\}$,

d) For each $ID_i \in \overline{\mathrm{S}}$, computes $PK_i = \sum_{j=1}^{n} h_j \mathcal{Y}_j$ where h_j is the jth bit of $H_0(\mathrm{ID})$ and $\mathcal{Y}_j$ is the jth value in the master public key vector, $\mathrm{j} = 1, \dots, n$.

e) For each $ID_i \in \overline{\mathrm{S}}$, generates a random number γ_i as an index number of each user for each $ID_i \in \overline{\mathrm{S}}$.

f) For each $ID_i \in \overline{\mathrm{S}}$, creates forgery identities as confusion with PK_i. Selects $a_i, b_i \xleftarrow{R} \mathbb{Z}_q^*$ arbitrarily, and computes:

- $\alpha_i = a_i P + b_i \cdot PK_i$
- $\beta_i = -b_i^{-1} H_1(\alpha_i)$
- $m_i = a_i \beta_i$

Where (α_i, β_i) is a valid EC-Elgamal signature of m_i, because $m_i \cdot P = H_1(\alpha_i) \cdot PK_i + \beta_i \alpha_i$.

g) Defines Ω together with the event description msg, threshold value $\boldsymbol{t}$, and ring size r as an RQP.

$$\Omega = \begin{pmatrix} \{ID_1,\ ID_2, \dots, ID_{r-t}\}, \{\gamma_1,\ \gamma_2, \dots, \gamma_{r-t}\}, \\ \{m_1,\ m_2, \dots, m_{r-t}\},\ \{\beta_1,\ \beta_2, \dots, \beta_{r-t}\} \end{pmatrix}$$

h) Broadcasts this RQP to invite other witnesses.

3) Reply

If $\mathcal{R}$ wants to join the announcement, $\mathcal{R}$ does the following steps:

a) Gets (msg, t, r, Ω) from the receiving RQP.

b) Computes $k = H_2(msg)$, where k is a symmetric key and the size of the key k is l.

c) Constructs a polynomial f over $\mathrm{GF}(2^l)$ that satisfies following statement:

- $deg(f) = r - t$
- $f(0) = H_3(t||r)$
- $f(\gamma_i) = E_k(m_i), i\ = 1, \dots, r - t$

d) Chooses random index $\gamma \xleftarrow{R} \mathbb{Z}_q^*$, where $\gamma \notin \{\gamma_1, \gamma_2, \dots, \gamma_{r-t}\}$, and computes $m\ = E_k^{-1}(f(\gamma))$.

e) Generates $\mathrm{c} \xleftarrow{R} \mathbb{Z}_\mathrm{q}$, and computes the EC-Elgamal signature (α, β) of m, where $\alpha = cP,\ \beta = (m - skH_1(\alpha))c^{-1}$.

f) Wraps $(\gamma, \mathrm{m}, (\alpha, \beta), ID)$ as the RPP. Finally, $\mathcal{R}$ forwards the RPP back to the $\mathcal{I}$.

4) Announcement

$\mathcal{I}$ is waiting for RPPs after forwarding a RQP. Once $\mathcal{I}$ receives over t RPPs, we assume $S = \{ID_{r-t+1}, ID_{r-t+2}, \dots, ID_r\}$. $\mathcal{I}$ conducts as follows:

a) Combines the signatures in the RPPs with the forgery signatures in the RQP to produce a threshold ring signature in AGP, as following:

$$AGP = (< \gamma_1, m_1, \alpha_1, \beta_1 >, \dots, < \gamma_r, m_r, \alpha_r, \beta_r >, msg, t, S \cup \overline{S})$$

5) Verification

When $\mathcal{V}$ receives an AGP, he/she verifies as following:

a) Gets AGP $(< ID_1, \gamma_1, m_1, \alpha_1, \beta_1 >, \dots, < ID_r, \gamma_r, m_r, \alpha_r, \beta_r >, msg, t)$.

b) Computes $k = H_2(msg)$, where k is a symmetric key.

c) For each ID_i, computes $PK_i = \sum_{j=1}^{n} h_j \mathcal{Y}_j$, where $i = 1, \dots, r$ and h_j is the jth bit of $H_0(ID)$ and $\mathcal{Y}_j$ is the jth value in the master public key vector and $j = 1, \dots, n$.

d) For each ID_i, verifies the equation $m_i \cdot P = H_1(\alpha_i) \cdot PK_i + \beta_i \alpha_i$. If any one of the tuples $< m_i, \alpha_i, \beta_i >$does not satisfy the equation, $\mathcal{V}$ rejects.

e) Selects a pair $< (0), H_3(t||r) >$ and $r - t$ pairs of $< \gamma_i, E_k(m_i) >$ randomly in AGP, and reconstructs the polynomial f :

- $deg(f) = r - t$
- $f(0) = H_3(t||r)$
- $f(\gamma_i) = E_k(m_i), i = 1, \dots, r - t$

f) Checks the rest of the pairs$< \gamma_j, E_k(m_j) >$ in the AGP. If anyone of them does not satisfy $f(\gamma_j) = E_k(m_j)$, $\mathcal{V}$ rejects the signature. Otherwise, $\mathcal{V}$ accepts the signature. The AGP is valid.

D. Security Analysis

***Sybil*-resistance:** The threshold cryptographic technique used in the protocol is based on *Lagrange Interpolation*, and a signature is only generated by a fixed number of different private keys. This makes our protocol achieve ***Sybil*-resistance** as follows:

Specifically, if a malicious $\mathcal{I}$ forges more than $r - t$ users to forward request, $\mathcal{V}$ will detect this dishonest AGP in *Verification Phase* because an adversary $\mathcal{I}$ could not select more than $r - t$ random $< \gamma_i, E_k(m_i) >$ while keeping $deg(f) = r - t$. If he/she fixes the number r and t and forwards more than $r - t < \gamma_i, E_k(m_i) >$ tuples in RQPs, $\mathcal{R}$ could discover this attacks in the step of constructing f. Moreover, the $r - t$ tuples in each RQP must be the same in one announcement request. Otherwise, all the f' constructed by $\mathcal{R}$s are different. Thus, it will be detected in *Verification Phase*.

Furthermore, if a malicious $\mathcal{R}$ replies more than once in one announcement, this could be simply detected by $\mathcal{V}$, since $\mathcal{V}$ verifies the equation $m_i \cdot P = H_1(\alpha_i) \cdot PK_i + \beta_i \alpha_i$. PK_i is related to the ID of $\mathcal{R}$. Thus, $\mathcal{R}$ could not sign more than once without others' sk.

Privacy-leakage: We take into account the privacy of the AGPs, since only the AGPs will be multi-hop disseminated (i.e., broadcasted) and received by multiple vehicles. The privacy of participants generating the AGPs is protected by the threshold ring signature. The threshold ring signatures provide indistinguishability among all ring members, but only part of members are the actual signers of the signature. Specifically, it means the group of participants of the aggregation will be concealed in a larger group of possible signers. Therefore, our protocol protects users' privacy.

Specifically, in *Echo-Announcmenet*, if the verification finishes successfully, the *Verifier* would know that this announcement is valid. He/she also knows that over t participants claim the correctness of the aggregated-announcement with a ring signature. However, the *Verifier* does not know the identities of the participants, since the identities of the t participants will be covered in the large numbers r of group users in this area.

Reliability: In *Echo-Annoucnemet*, digital signatures are used to protect AGPs. Thus, if an adversary wants to tamper with an announcement, it is necessary for him to construct fraudulent signatures. The symmetric encryption in our protocol ensures indistinguishability in the phase of verification. Without changing the verification polynomial, the only way that an adversary creates forgeries successfully in the phase of request is to break the one-way functions. Specifically, the adversary has to forge valid *ElGamal* signatures. However, the probability of doing that is considered to be negligible. Moreover, if the adversary wants to report a dishonest event report with high trust level in a legitimate way, he has to get enough RQPs to generate an AGP. However, in *Echo-Annoucement*, we assume the number of malicious vehicles is relatively smaller. Thus, the adversary gets few replies to generate a valid AGP.

Prevention of upgraded attack: In *Echo-Announcement*, an AGP's trust level is related to the threshold value of amounts of participants. The more participants an AGP contains, the higher trusted level an AGP will be. If an adversary is willing to launch an upgraded attack, he has to modify the threshold value of an existing legitimate AGP. In other words, the adversary needs to persuade the receivers to believe the modified AGP and the modified trust level. However, for each receiver, the verification of the signature will fail, because they cannot calculate a desirable verification result from a wrong threshold value. Thus, any modification will lead to the failure of verification.

Prevention of replay attack: If an adversary obstructs others in traffic information by replaying an existing legitimate message received before. However, the description msg of the event is generated with a description of time in detail. Receivers will check the current time and the event time when they receive an AGP. The adversary will fail reply attack unless the adversary tampers the content of the message and forges a valid signature. However, the latter is prevented due to the analysis above.

Prevention of usurpation and forgery: In *Echo-announcement*, the key is identity-based and generated by ID. According to the assumption, ID is difficult to be tampered. pk is also not distributed in the protocols. Thus, in the process of signature verification, risk of usurpation and forgery is reduced. Moreover, Sybil attack is a typical attack of usurpation and forgery[7]. According to the analysis above,

TABLE II
A TYPICAL SCENARIO OF CREDITCOIN

Phases	Initiator $\mathcal{I}$	Replier $\mathcal{R}$	Verifier $\mathcal{V}$ With Post-mission
Phase 0	--	--	**Post a mission with reward;** valid credit info by cloud application server
Phase 1	Hunt a mission for reward; valid credit info by cloud application server; **start forwarding RQPs**	--	Waiting
Phase 2	Waiting	**Receive a RQP; send a RPP for rewarding**	Waiting
Phase 3	Get RPPs over *t* threshold; generate an announcement; **forward an AGP**	Receive a transaction from public address; wait to be confirmed by RSUs	Waiting
Phase 4	Waiting	A consensus is reached by RSUs; transaction confirmed; get $\boldsymbol{coins_0}$	Receive AGP; verify ring signatures successfully**, closing the mission**; pay $coins_1$, waiting to be confirmed by RSUs
Phase 5	**Mission is completed;** Receive a transaction from V, waiting to be confirmed by RSUs	Get balance; reset role	A consensus is reached by RSUs; pay $\boldsymbol{coins_1}$; transaction confirmed
Phase 6	A consensus is reached by RSUs; transaction confirmed; get reward $\boldsymbol{coins_1}$ as an incentive	--	Get balance; reset role
Phase 7	Get balance; reset role	--	--

Echo-Announcement is ***Sybil*-resistance**. Thus, to some extent, *Echo-Announcement* prevents usurpation and forgery beforehand.

VI. PROPOSED BLOCKCHAIN-BASED INCENTIVE MECHANISM IN CREDITCOIN

In this section, we propose the incentive mechanism used by ***CreditCoin*** network. The mechanism works with the proposed announcement protocol *Echo-announcement*, with an objective to encourage the users to honestly forward the true announcements. We first introduce the network, and then we present the definitions and protocols in detail.

A. Basic Idea

As we proposed in Section II, ***CreditCoin*** consists of five entities: the Trusted authority, the Trace manager, users (i.e., OBUs), RSUs, and a cloud application server. Each user is given a credit account, storing reputation points, i.e., ***coins***. The users are encouraged to forward and receive packets with the incentive to increase their ***coins***. Existing work [42-45] has proved this mechanism is effective in crowdsourcing tasks and ad hoc networks. However, they are not suitable for the privacy-preserving requirements of vehicular announcement networks.

We use Blockchain-based network to build accounts and record transactions so that users' behavior is in privacy-preserving without loss of reliability.

B. Collaboration with Echo-announcement

We show a typical scenario of ***CreditCoin*** in Table II. A user behaves as $\mathcal{R}$, $\mathcal{I}$ or $\mathcal{H}$ in the network as we proposed in Section V. Specifically, if the user is an new user, he/she has few ***coins*** in initial amount. This leaves him/her with no choice but to behave as $\mathcal{R}$, replying to others' RQPs. He/she will remain active if he/she hopes to post a rewarding mission for traffic information from a certain area someday.

After saving enough ***coins***, he/she could change the role to $\mathcal{I}$ and forwards RQPs later for two reasons: one is that he/she wants to get more ***coins*** from a rewarding mission; the other one is that he/she volunteers to tell new information to others. Since sending a request costs ***coins***, it protects honest users by reducing fraudulent or meaningless requests in the network.

When the amount of ***coins*** is enough for posting a mission, he/she can create a mission to get traffic information. The mission containing more ***coins*** will be finished sooner than those with fewer rewarding ***coins***. Thus, users are encouraged to reward more ***coins*** in missions. This kind of incentives briefly encourages the flow of ***coins*** and remains the ***CreditCoin*** network active. Furthermore, ***CreditCoin*** enhances the availability and non-repudiation of the vehicular announcements effectively. Adversaries can hardly modify ***coins*** because of the process of voting and the records on Blockchain. Also, for any adversaries forwarding dishonest messages, TM is able to associate their identities with their addresses.

We also design a credit expiration mechanism to protect users from coin-reserving attacks. We set a settlement day to each address. If the ***coins*** in the address are not spent until the day set, some of the ***coins*** will be sent back to the *public*.

We divide the operations of users into several trading propositions in ***CreditCoin***. The details are described as follows; V-C describes the roles and propositions. V-D describes detailed descriptions. V-E describes security analysis and some discussions.

C. Roles and Trading Propositions

1. **Consensus server:** The consensus server is an entity that receives transactions and participates in the consensus phase. RSUs or official public vehicles are the consensus servers in our ***CreditCoin***. There are l servers in ***CreditCoin***. Users are connected directly to at least one server. For each server $s_z, z = 1,2, \cdots l$, there is a *Unique Node List* (*UNL*), called UNL_{s_z}. The list records the identities of multiple servers, each of which is directly connected to the list. In the consensus phase, s_z only believes the vote

results sent by the server listed on its UNL_{s_z}. According to the proof of David et al. in[25], when the probability that any of the servers in UNL_{s_z} attempts to initiate a collusion with other servers in the same list is less than 20%, with the increasing number of servers, the probability of making an undesirable consensus approaches to none quickly. As Armknecht et al. [30] proved, in order to avoid bifurcation, the repetition rate of servers in two diffident *UNLs* equals to or is greater than $\rho / 2$, where ρ is a threshold of a voting rate about *yes*.

2. **Cloud application server:** Cloud application server manages and stores non-privacy information in the VANETs, such as msg consisting in AGPs. For security reasons, they are separated from the encrypted information to help the entire network operate safely. Application server spreads public information, such as missions and announcements. Cloud application server works as a *watcher* in ***CreditCoin***.
3. **User (OBU):** The user is an entity that trades in ***CreditCoin*** network. He/she creates or receives transactions. A user behaves in varieties of roles, such as *Hunter*, *Replier*, *Initiator,* and *Verifier*. We will elaborate these roles later in the following part.
4. **Public role:** Public role is defined similarly to the user. However, it is more privileged than the user. It receives and sends transactions and creates ***coins*** as well.
5. **Trusted authority**: Trusted authority takes charge of the generation and delivery of public keys. It creates d addresses for each user, and records the relationship between users and addresses.
6. **Trace manager:** Trace manager is the role that traces malicious users. If a fraudulent transaction is reported to Trace manager, Trace manager will trace the malicious users with the help of Trusted authority and send a report to cloud application server.

In the following part, *Trusted authority* is denoted as $\mathcal{T}$. *Cloud application server* is denoted as $\mathbf{S_{app}}$. *Consensus server* is denoted as $\boldsymbol{s}$. The *Hunter* that requires traffic information is denoted as $\mathcal{H}$. The *Initiator* in the aggregation process is denoted as $\mathcal{I}$. The *Replier* in aggregation process is denoted as $\mathcal{R}$. The *Verifier* of an announcement is denoted as $\mathcal{V}$. The public role is denoted as $public$. Trace manager is denoted as TM.

CreditCoin obeys several trading propositions to build an incentive announcement network with coin balances. The proposition rules are as follows:

Proposition 1: {Reply an RQP} When $\mathcal{T}$ broadcasts a RQP, if $\mathcal{R}$ replies to the RQP, $\mathcal{R}$ will get several ***coins***. Particularly, in order to avoid the abuse of replies, the frequency of a user's daily reply is limited, which we set it up to 3 times in our simulation.

Proposition 2: {Post a rewarding task} $\mathcal{H}$ will get information about an area only if an announcement about the area is forwarded. Thus, $\mathcal{H}$ constructs a rewarding mission with the attractive award of ***coins***. Then, the mission is sent to $\mathbf{S_{app}}$ and posted online to users.

Proposition 3: {Finish a rewarding mission} When a user in a particular area becomes $\mathcal{I}$ and forwards the AGPs successfully, $\mathcal{I}$ is eager to get a reward as an incentive. If $\mathcal{H}$ approves the AGP, the $\mathbf{S_{app}}$ will give $\mathcal{I}$ a number of ***coins*** as *bounty*. We also recommended traffic management apartment to become $\mathcal{H}$.

Algorithm 2: Address Generation

Input: public key $\left(sk_{u_j}, pk_{u_j}\right)$, USER_LIST, d // $\mathcal{T}$ generated key pairs.
for *j*=0; *j*<USER_LIST. length; *j*++ do
  for i=0; i< d; i++ // d could change by users' intentions
  $sk_{u_j}^{(i)} = sk_{sig}$; $pk_{u_j}^{(i)} = pk_{sig}$;
  $addr_{u_j}^{(i)} = pk_{u_j}^{(i)}$;
  end
end
output: d key pairs for one user u_j,

Proposition 4: {Initiate of an announcement} According to the description in Section V, if $\mathcal{I}$ hopes to forward an announcement, RQPs should be sent to other users. ***Coins*** should be spent before $\mathcal{I}$ sends RQPs. If $\mathcal{H}$ approves the announcement, $\mathcal{I}$ will get rewards according to Proposition 3. In general, the amount of reward is usually higher than the cost of sending RQPs. According to our hypothesis, if the request sent by $\mathcal{I}$ is not honest, few people will respond to him/her. This feature reduces ***coins*** of $\mathcal{I}$. Thus, malicious users cannot continue to send messages in ***CreditCoin***. Therefore, this feature also increases the influence of honest users with high prestige.

Proposition 5: {Mechanism of reputation expiration} The unspent ***coins*** will be halved in a certain period. This mechanism simply prevents the accumulation of ***coins*** that could be used to attack. This rule is used in many incentive mechanisms.

D. Descriptions

We have given the details of announcement protocol in ***CreditCoin*** in Section V. Thus, in this section, we describe the construction of network and incentive mechanism in ***CreditCoin***.

1) Setup

Let ρ be the percentage of servers voting *yes* in the phase of consensus. Let l be the total number of servers in a UNL. λ is the length of random inputs. First, a public parameter set PP is chosen at the beginning of the scheme:

$$PP = \{\rho, l, \lambda\}$$

Then, a collision-resistance hash function is chosen as below:

$$H_{pubK}: \{0,1\}^* \rightarrow \{0,1\}^\tau$$

Finally, we choose an unforgeable digital signature algorithm to ensure the ownership of ***coins***:

$$\left(G_{sig}, K_{sig}, S_{sig}, V_{sig}\right):$$

- $G_{sig}\left(1^\lambda\right) \rightarrow pp_{sig}$, by using a random number of length λ, $G_{sig}\left(1^\lambda\right)$ works as a public parameter generation algorithm and generates a public parameter pp_{sig} for signature scheme.

- $K_{sig}(pp_{sig}) \rightarrow (pk_{sig}, sk_{sig})$, by using public parameter pp_{sig} as the input, $K_{sig}(pp_{sig})$ works as a public key generation algorithm and generates public key pair (pk_{sig}, sk_{sig}).
- $S_{sig}(sk_{sig}, msg) \rightarrow \sigma$, by using private key sk_{sig} and message msg as the input, S_{sig} works as a signature algorithm and generates a signature σ of message msg.
- $V_{sig}(pk_{sig}, msg, \sigma) \rightarrow 0/1$, by using the input public key pk_{sig}, message msg and signature σ, V_{sig} works as a signature verification algorithm. If the correctness of the signature is verified, V_{sig} outputs 1, otherwise outputs 0.

Assuming that d is the amount of addresses for each user, and q is the number of users in network, the addresses $addr_{u_j}^{(i)}, i = 1,2,\cdots,d$ in ***CreditCoin*** are a set of hash strings generated by $\mathcal{T}$, and $\mathcal{T}$ distributedly forwards to user $u_j, j = 1,2,\cdots q$. The Algorithm 2 is *Address Generation*.

Each user has d addresses as his/her willing. In order to generate the ith address $addr_{u_j}^{(i)}$ for a user u_j, $\mathcal{T}$ generates a pair of keys $\left(sk_{u_j}^{(i)}, pk_{u_j}^{(i)}\right)$ by using $G_{sig}(1^{\lambda}) \rightarrow pp_{sig}$ and $K_{sig}(pp_{sig}) \rightarrow (pk_{sig}, sk_{sig})$, where $sk_{u_j}^{(i)} = sk_{sig}$ and $pk_{u_j}^{(i)} = pk_{sig}$. Then, $\mathcal{T}$ distributes d pairs of keys to u_j. $pk_{u_j}^{(i)}$ is recognized as one of the addresses for a user, recognized as $addr_{u_j}^{(i)}$. Then, $\mathcal{T}$ destroys private keys but updates the list that records the relationship between addresses and the identifications of users for tracing.

Moreover, *Pay-to-Public-Key-Hash* (P2PKH) script [46] is applied to realize locking and unlocking scripts in ***CreditCoin***. The method of $ScriptSig$ is a kind of *Forth-like Reverse-Polish* notation stack-based execution language. It consists of $scriptsig$ and $scriptPubKey$. The form of scripts is described as below:

- $\boldsymbol{scriptsig:}$

$$\langle sig \rangle \langle PubK \rangle$$

Where:

$$S_{sig}(sk_{sig}, vout_1 \| vout_2 \| \cdots \| vout_{\beta} \| t_{create}) \rightarrow \sigma$$
$$\langle sig \rangle \coloneqq \sigma$$
$$\langle PubK \rangle \coloneqq pk_{sig}$$

- $\boldsymbol{scriptPubKey:}$

DUP HASH $\langle PubKHash \rangle$ EQUALVERIFY CHECKSIG

Where:

$$\langle PubKHash \rangle \coloneqq H_{PubK}(pk_{sig})$$

- $\boldsymbol{ScriptSig:}$

$\langle sig \rangle$ $\langle PubK \rangle$ DUP HASH ‖
$\langle PubKHash \rangle$ EQUALVERIFY CHECKSIG

Only if a $ScriptSig$ is verified, the ownership of this transaction output in $tr_{n'}$ will be verified.

2) Transaction

In ***CreditCoin***, the identification of each user is bound with a series of addresses recorded by $\mathcal{T}$. Thus, a transaction flow is from addresses to addresses. Let tr be a transaction. Let $txid$ be the index of transaction tr. Let $addr_{\beta}$ be the address of a receiver. Let $\{vin_1, vin_2, \cdots, vin_{\alpha}\}$ be the set of transaction inputs. Let $\{vout_1, vout_2, \cdots, vout_{\beta}\}$ be the set of transaction outputs. Let t_{create} be the created time. Based on [3], we describe a simple transaction format for ***CreditCoin***, as follows:

$$tr \coloneqq (txid, \{vin_1, \cdots, vin_{\alpha}\}, \{vout_1, \cdots, vout_{\beta}\}, t_{create})$$
$$where$$

- $vin_{\alpha} \coloneqq (txid', scriptsig', addr_{\alpha})$
- $vout_{\beta} \coloneqq (value_{\beta}, scriptPubKey_n, addr_{\beta})$

Assume that tr' is a valid historical transaction that has been recorded on the block chain. An input vin_{α} is related to a fixed and unique $vout'$ that has been recorded in the output set of tr'. Thus, every index $txid'$ in every input vin_{α} on tr is identical with the $txid'$ on a valid historical transaction tr'.

$scriptsig'$ is a script that is used to unlock the corresponding locking script $scriptPubKey'$ in transaction tr'. A transaction output $vout_{\beta}$ is $value_{\beta}$ ***coins*** received in address $addr_{\beta}$. On the contrary the of unlocking script, $scriptPubKey_n$ is a locking script.

In the following, for simplicity, let vin_* be the set of vin. Let $vout_*$ be the set of $vout$. We omit the detail definition of vin, as it has been defined above.

Five kinds of propositions (more detail in VI-C) are defined in ***CreditCoin***, as below:

- **Proposition 1**: {Reply to an RQP} tr_{reply} is a transaction from $public$ to $\mathcal{R}$, including a change to public addresses itself:

$$tr_{reply} \coloneqq (txid, vin_*, vout_*, t_{create})$$
$$where$$
$$vin_* \coloneqq \left(vin_{Public,1}^{(i)}, vin_{Public,2}^{(i)}, \cdots, vin_{Public,\alpha}^{(i)}\right)$$
$$vout_* \coloneqq \left(vout_{\mathcal{R}}^{(i')}, vout_{Public,1}^{(i'')}, \cdots, vout_{Public,\beta}^{(i'')}\right)$$

- **Proposition 2**: {Posting a rewarding task} tr_{post} is a transaction from $\mathcal{H}$ to $public$, including a change to $\mathcal{H}$ or others.

$$tr_{post} \coloneqq (txid, vin_*, vout_*, t_{create})$$
$$where$$
$$vin_* \coloneqq \left(vin_{\mathcal{H},1}^{(i)}, vin_{\mathcal{H},1}^{(i)}, \cdots, vin_{\mathcal{H},\alpha}^{(i)}\right)$$
$$vout_* \coloneqq \left(vout_{public}^{(i')}, vout_{\mathcal{H},1}^{(i'')}, \cdots, vout_{\mathcal{H},\beta}^{(i'')}\right)$$

- **Proposition 3:** {Finishing a rewarding task} tr_{reward} is a transaction from $public$ to $\mathcal{I}$, including a change to $\mathcal{I}$ or others.

$$tr_{reward} \coloneqq (txid, vin_*, vout_*, t_{create})$$
$$where$$
$$vin_* \coloneqq \left(vin_{public,1}^{(i)}, vin_{public,2}^{(i)}, \cdots, vin_{public,\alpha}^{(i)}\right)$$
$$vout_* \coloneqq \left(vout_{\mathcal{I}}^{(i')}, vout_{Public,1}^{(i'')}, \cdots, vout_{Public,\beta}^{(i'')}\right)$$

- **Proposition 4:** {Initiation of an announcement} $tr_{announce}$ is a transaction from user $\mathcal{I}$ to $public$, including a change to $\mathcal{I}$ or others. When RQPs are proposed, the related transactions must be verified at first. In other words, the person who requests an aggregated-announcement must provide a *Merkle Tree* root

and the corresponding path of that transaction to provide a proof of the aggregated-announcement related transaction to others.

$$tr_{announce} \coloneqq (txid, vin_*, vout_*, t_{create})$$
$$where$$
$$vin_* \coloneqq \left(vin_{J,1}^{(i)}, vin_{J,2}^{(i)}, \cdots, vin_{J,\alpha}^{(i)}\right)$$
$$vout_* \coloneqq \left(vout_{public}^{(i')}, vout_{J,1}^{(i'')}, \cdots, vout_{I,\beta}^{(i'')}\right)$$

- **Proposition 5:** {Reputation expiration} If there is an unspent transaction in unspent transaction pool and the creation time of the unspent transaction reaches the settlement day, a transaction from user u_j to $public$ will be created and sent to $\boldsymbol{s_z}$. By this transaction, some of the ***coins*** are sent to public addresses, and the rest of ***coins*** are sent back to user's addresses. This method is called *coin damping*. In the process of verifying transactions, created time $t_{create}{}'$ of each input transaction is verified by checking the corresponding transaction $txid_{n'}$ on the chain. If a user denies doing the *coin damping*, the ***coins*** in this address will be banned from spending. *Coin damping* is described as below:

$$tr_{expire} \coloneqq (txid, vin_*, vout_*, t_{create})$$
$$where$$
$$vin_* \coloneqq \left(vin_{u_j,1}^{(i)}, vin_{u_j,2}^{(i)}, \cdots, vin_{u_j,\alpha}^{(i)}\right)$$
$$vout_* \coloneqq \left(vout_{public}^{(i')}, vout_{u_j,1}^{(i'')}, \cdots, vout_{u_j,\beta}^{(i'')}\right)$$

3) Voting Consensus

In ***CreditCoin***, RSUs or official public vehicles undertake the work of consensus. Since vehicles move fast and respond rapidly in VANETs, the consensus algorithm with a short period in the process of reaching an agreement is better than a computational-based algorithm (i.e., *Proof of Work* algorithm). Thus, we design our consensus phases based on *Byzantine Fault Tolerates* algorithm [32] to satisfy the requirements of efficiency is the scenario of VANETs.

In the consensus phase, a server $\boldsymbol{s_z}$ has a candidate set of transactions. Each server $\boldsymbol{s_z}$ tries its best to receive transactions that have been already broadcasted in the consensus network in a determined period. Then, servers initiate a new consensus round.

For each tr, server $\boldsymbol{s_z}$ verifies the validation of addresses, input transactions and $ScriptSig$. The verified transaction tr_v is acceptable to candidate set cs_{s_z}.

At the beginning of each consensus round, a union set for the candidates is needed. To get the union set, each $\boldsymbol{s_z}$ implements the Algorithm 3. $\boldsymbol{s_z}$ merges its candidate set cs_{s_z} with other cs_{s_i} from $\boldsymbol{s_i}$ listed in its UNL_{s_z}. Then, $\boldsymbol{s_z}$ gets the candidate union set $\boldsymbol{CS'_{s_z}} = cs_{s_z} \cup cs_{s_1} \cup cs_{s_2} \cup \cdots \cup cs_{s_l}$. Phases of transaction verifications and votes are initiated.

In the mth phase, based on the validation of each transaction, $\boldsymbol{s_i}$ listed in their UNL_{s_z} votes to transactions in candidate set $\boldsymbol{CS'_{s_z}}$. If there is a server $\boldsymbol{s'_z}$ that does not vote in time, it will be deleted from UNL_{s_z} and will not be considered in the other rounds any more. For each transaction, if the rate of affirmative votes is no more than $\rho_m = (5 + \mathrm{m} - 1)/10$, this transaction will put back into the candidate set $\boldsymbol{cs_{s_z}}$ for a new round. Otherwise, it is still considerable in another new phase. A consensus round is ended at the end of mth consensus phase, where $\rho = \rho_m$.

Algorithm 3: Candidate Union Set

Input: cs_{s_z} , UNL_{s_z}
$cs'_{s_z} = cs_{s_z}$;
for *i*=0; *i*<UNL_{s_z}.size(); *i*++ do
  $\boldsymbol{s_i} = UNL_{s_z}.get(i)$;
  $\boldsymbol{CS'_{s_z}} = cs'_{s_z} \cup cs_{s_i}$
end
output: candidate set $\boldsymbol{CS'_{s_z}}$ of $\boldsymbol{s_z}$

Algorithm 4: Consensus Phase

Input: UNL_{s_z} , $list_{vote}^{self}$
$list_{vote}^{z} = list_{vote}^{self}$;
for *i* = 0; *i* < UNL_{s_z}.size(); *i*++ do
  $s_i = UNL_{s_z}.get(i)$;
  $list_{vote}^{i} = request(s_i)$;
end
$list_{decision} = decision(list_{vote})$;
for *j* = 0; *j* < $list_{decision}$.size(); *j*++ do
  if $list_{decision}$.get(*j*) == *accept*
    $\boldsymbol{CS''}.add(\, tr.get(i)\,)$;
  else
    continue;
end
output: consented transactions' candidate set $\boldsymbol{CS''}$

At the end of each consensus round, the states of all servers' transaction records are the same. In other words, when a consensus round is ended, an agreement is reached for all transactions recorded in candidate set $\boldsymbol{CS''}$. All of the transactions in $\boldsymbol{CS''}$ are valid. More detail is shown in Algorithm 4. Furthermore, at the end of each consensus phase, the relevance is built between the present transaction set and its previous set to protect the voting result.

Based on Blockchain, a block consists of a hash value of pre-block, a transaction set, a timestamp and other information that are significant to record. Each block has a unique and fixed hash value. Because of the good features of hash function, any modification of content causes the change of hash value. Therefore, if an adversary attacks, he/she not only needs to tamper the content of a block but also needs to re-calculate each hash value of blocks after it.

4) Construction of Blockchain

Assuming that there are θ blocks on the chain, the depth of the block chain is θ. The construction of block chain is as below:

$$BlockChain \coloneqq Block_1 \| Block_2 \| \cdots \| Block_\theta$$

Arranging the transactions of $\boldsymbol{CS''}$ in an ascending sort, constituting a *Merkle Hash Tree* [3, 41, 47], and calculating a hash value of block, the new block's structure is described as below:

$$Block_{new} \coloneqq (rt, num, info, hash_{pre}, hash_{new}, hash_{next})$$

In $Block_{new}$, $hash_{pre}$ is a hash value of a recent block. rt is a merkle tree root of present block. $hash_{new}$ is a hash value of present block where $hash_{new} = Hash\,(rt, num, info, hash_{pre})$. $hash_{next}$ is a hash value of the next block. num is the number of transactions in $\boldsymbol{CS''}$. $info$ is a series of transactions.

After reaching the consensus, the new block will be added to the end of the block chain, as follows:

$$BlockChain \coloneqq Block_1 \| Block_2 \| \cdots \| Block_\theta \| Block_{new}$$

5) Transaction Tracing

A transaction with fraudulent behaviors is found via the record of relationships between messages and transactions. Then, the sender's address $addr_{u_j}^{(i)}$ is retrieved from the block chain. Due to the list recording the relations between addresses and users is stored in $\mathcal{T}$, it can be found easily.

Assuming that $txid^\circ$ forwarded by an adversary carries out some fraudulent behaviors in VANETs, an honest user reports this transaction index to TM. Then, TM makes a request to consensus server $\boldsymbol{s_z}$ and requests information about the corresponding transaction record. Then, TM analyzes the transaction record, and separates the user address $addr^\circ$ from vin_*. Finally, TM makes a request to $\mathcal{T}$ and requests the identity information of the corresponding user address $addr^\circ$. $\mathcal{T}$ responds the request to TM. In this way, TM gets the identity of the adversary u° .

E. Security Analysis

Privacy-preserving: In the view of users, the owners of the addresses are unaware. Even the number of addresses for a certain user is unaware. Furthermore, a user chooses different addresses for different transactions to forward or receive ***coins***. There are hundreds of thousands of addresses in ***CreditCoin***. Thus, it is difficult to analyze the total transactions of a user or the balance from Blockchain. The relationship between users and transactions are covered, due to the busy traffic in practice. The sender is usually hidden in the network. The same idea is proposed in Section V as well.

Transaction tampering resistance: In ***CreditCoin***, the transactions recorded on Blockchain already has got the agreements from all servers. Moreover, Blockchain maintains interactive consistency of servers. According to the description of ***CreditCoin***, a hash chain is used to protect the order and the information of blocks. These hash values are unique for each block. Modifying any content of any block will cause a change to the hash values of the other blocks. Depending on the properties of the hash function, if an adversary performs a perfect tampering, he/she not only needs to modify the contents of the block but also needs to modify and recalculate the hash values of all blocks after the modified block. Therefore, if there are hundreds of thousands of blocks regardless of workload, the longer the block chain is, the better security will be.

Prevention of forging a new transaction: In the consensus phase, servers verify each transaction waiting to be added to the block chain. Specifically, it verifies the inputs/outputs of transactions as well as their related transactions. Each server needs to check whether a transaction input corresponds to a valid, unused transaction's output, which is already recorded on Blockchain. If a user tries to forge a transaction that does not exist or is not valid, the consensus server will easily find it out by checking the transaction history on the chains; if an adversary attempts to forge a false address, the server will find such a spoofing by verifying the existence of the addresses. Therefore, forging transactions is difficult in our solution.

Prevention of thieving addresses (usurpation): If an adversary attempts to use the ***coins*** in others' addresses maliciously, the adversary needs to provide a valid unlocking script for a transaction input. The unlocking script contains the user's signature, which is signed by the private key of address. In our hypothesis, each user only knows the private keys of his addresses but has no knowledge of others' private keys. Therefore, it is difficult for an adversary to construct an unlocking script for locking script of others unless the private key is compromised.

Prevention of modifying coins: In ***CreditCoin***, the account is only a concept in the address set. The balance is calculated from the records of transaction outputs. If an adversary attempts to modify the balance of ***coins***, he/she must create a new transaction sent back to him/her. The fraudulent transaction must get approvals from the majority of servers. The balance is modified only if the related fraudulent transaction was added to the block chain. An adversary could also try to achieve his goal by tampering an existing transaction sent to him. However, according to the analysis above, it is difficult to modify the transaction records. The difficulty is equivalent to breaking the one-way hash. Thus, it is not feasible for an adversary to modify the balance of any accounts.

Prevention of replay attack: Each transaction has a unique identifier $txid$. Therefore, transactions with the same identifier will be rejected by the consensus servers. In addition, the transaction with an invalid input is rejected in the phases of consensus, and thus replay attacks are prevented.

Prevention of man-in-the-middle attack: First of all, since the VANETs is usually based on SSL and 802.11p, some traditional man-in-middle attacks will not succeed. We now assume that an adversary, called Malice, tries to launch a man-in-the-middle attack on the transaction delivered from user Alice to consensus server Bob (i.e., the transaction hijack attack). If Malice modifies any contents of the transaction, such as the values of ***coins*** or addresses, the transactions will be rejected by Bob through the verification of the consensus phase. This case does not lead to any loss in Alice's account.

In ***CreditCoin***, consensus servers do not initiate a dialogue with a user directly, unless the user requests information to consensus servers. If a user requests some information to servers located in the same area, the user will receive many replies from more than one consensus servers. Moreover, assuming that Malice also tries to launch a hijack attack on the reply delivered from Bob to Alice and other users. These replies contain the results of a consensus round. If there is no adversary, all of these replies will

be correct because the state of the block chain is stable and remains correct. If Malice wants to respond to Alice's request, he must intercept and modify most of these replies, which is very difficult. Even if Malice succeeds, Alice's message will be still recognized and rejected by the consensus servers if she uses the information of these modified replies in future transactions. This case causes no loss of Alice's ***coins***. After several transactions, Alice is able to detect the problem and prosecute the attack.

Prevention of denial of service attack: In ***CreditCoin***, if the adversary initiates a denial of service attack. Since ***CreditCoin*** is a Credit network, which means replying and other operations need to spend ***coins***, this process effectively prevents the adversary from launching too many service requests. Besides, as discussed in Section VI-E, tempering account balance is difficult because the adversary can hardly increase their ***coins*** illegally in Blockchain. Moreover, we set up the Trace manager so that the identities can be traced through transactions. Therefore, the adversary is difficult to carry out long-period denial of service attacks.

VII. Simulations and Performance Analysis

In this section, we analyze the performance of ***CreditCoin*** through extensive simulations. In *Echo-Announcement* protocol, we implement our work with the library PolarSSL [48] and math library GMP [49]. We simulate systematically with a modern PC. The configuration is shown in Table III. The simulation strictly follows the protocols and patterns that may be used by the actual scenario. We are also looking for more cooperation in IoT (Internet of Things). In future work, we will strengthen the data collection of real applications.

A. Evaluation of Announcements Protocol

We use the *Curves NIST* [50] in our ring signature phase. Then, we implement the *Echo-Announcement* protocol and incentive mechanism in ***CreditCoin*** in VANETs simulation. We improve the simulator on the basis of the project [51] that is a simulation project for VANETs security, and we develop our own network from the open source code in [51] to achieve all of the proposed contributions ourselves. The details are shown as follows:

The map we used is generated from [52]. We set over 1000 vehicles nodes, and 40 RSUs shown in Fig. 3. Simulation parameter settings of our simulation scenario are shown in Table IV.

Fig. 4 gives the average of 1000 computation time of three phases proposed in Section V, in which $r \in \{20, 30, 40, 50\}$ while $t = 10$. Fig. 5 shows the average computation time of different threshold values while the ring size is fixed in 30. According to the simulation above, the average computation time meets the practical requirements of VANETs.

Specifically, the time from request to announcing is usually under 550ms up to the ring size of 50 in total as shown in Fig. 4. The computation time of request phase and reply phase decreases slightly with the increment of threshold value while the ring size is fixed, as shown in Fig. 5. Especially, the computation time of announcement verification is

TABLE III
Hardware Configuration

Hardware	Settings
CPU	Intel(R) Core(TM) i7-6700k quad-core CPU at 4.00GHz
RAM	64GB
Operation System	Ubuntu(R) 16.04 ARM64

Fig. 3. Simulation scenario map.

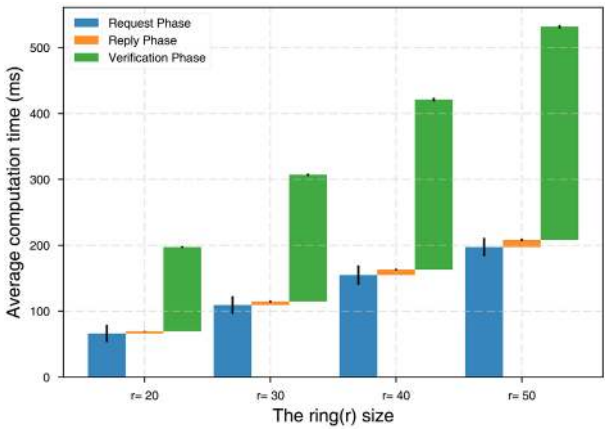


Fig. 4. The average computation time of three phases in *Echo-Announcement* proposed in Section V.

not related of the threshold value; it is mainly related to the size of the ring. The reason is that the most expensive computation in our protocol is related to the number of the forged signers. As the threshold is fixed, when the ring size decreases, the fewer forgery-signers are forged, the less computation time will cost. However, in the verification phase, the *Verifier* treats all signatures similarly. Therefore, verification is only related to the size of the ring. In addition, replying a request is very fast according to our simulation. This consequence is great for the responses of *Repliers*, as it only takes a slight effort to give a reply to the request.

Moreover, as we have analyzed in Section V, there is a certain probability of exposing anonymous information. There are two ways, and both are related to ring size and

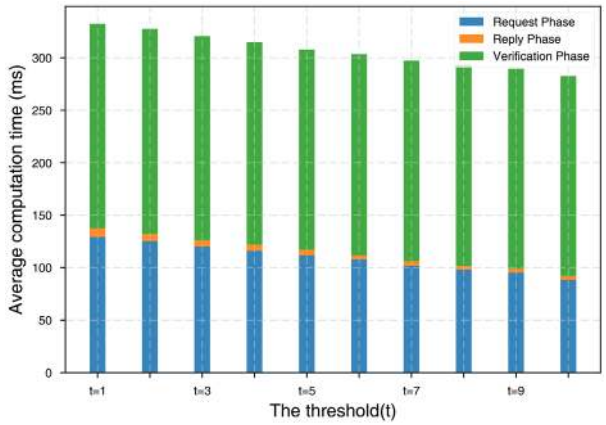


Fig. 5. The average computation time of different threshold values while the ring size is fixed in 30.

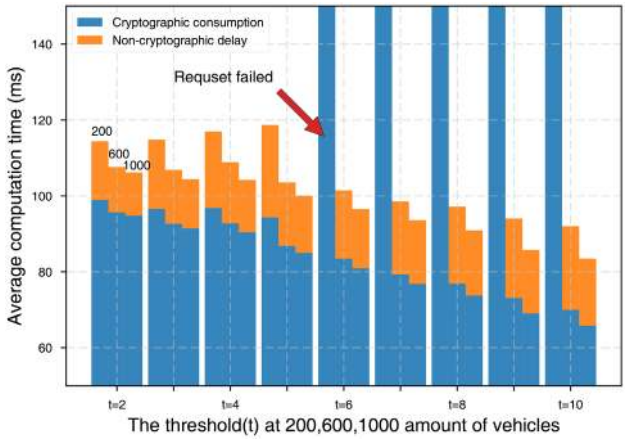


Fig. 6. Average communication time in cryptographic time consumption and non-cryptographic delay at different threshold values and vehicle amounts 200, 600, 1000.

threshold. The first is that adversaries could judge whether a member of ring is an actual signer. The second is that adversaries could expose at least a certain number of actual signers from an announcement. Furthermore, these two values are also related to the practical traffic condition. We will discuss the relations in the following part. Therefore, we recommend setting these two values by system advice, because of the importance of these two values in security.

Now we discuss the average communication time containing the cryptographic time consumption and non-cryptographic delay in different vehicle amounts in Fig.6.

Since the verification is local operations of the *Verifier*, it has few effects on the network condition. Thus, we do not consider the verification phase here. Fig. 6 gives the simulation results at ring size 30. The result shows that producing an announcement is also related to vehicle amount. The non-cryptographic delay in these network factors is much smaller than cryptographic computations. It should be noticed that in the low density at the vehicle amount of 200, the request phase failed due to time-out. In fact, we do not recommend vehicles producing the AGPs with low density and high threshold values. In summary, the biggest impact factor of the *Echo-Announcement* is undoubtedly cryptographic time consumption.

Furthermore, we now compare other typical solutions we followed, TA-Announcement [1] scheme and TA-Authentication [2] scheme. Table V shows the comparison of cryptographic consumption time. We assume that 10 vehicles are willing to report the same traffic jam on a certain road. Specifically, in TA-Announcement, the *Verifier* verifies the announcement one by one, which *Echo-Announcement* only authenticates once because of the aggregations. Also, we also compare the simulation time in TA-Authentication. We only compare the time of generation and authentication. In summary, the total result shows our advantages in algorithm efficiency. This efficiency owes to the CPK and *EC-Elgamal* signature protocol in the construction of our signatures. The cost of both verification and generation reduces a lot while the privacy is preserved adequately in our ***CreditCoin***.

TABLE IV
PARAMETER SETTINGS IN SIMULATIONS

Parameters	Settings
Time (one time)	6000s
Size of the Area	5000m×5000m
Number of Vehicles	200/600/1000
Average Speed of Vehicles	40km/h
Sending Range of OBU/RSU	200m/500m
Protocol	Local/802.11p

TABLE V
COMPARISONS OF COMPUTATION TIME (MS)

Phases	TA Announce-ment[1]	TA Authentica-tion[2]	***CreditCoin***
Sign	24.5	499	I:46.4 R:1.6
Verify	74.3x10	1233*	126.1
Total	767.4	1732*	174.1*

*It does not exactly match our phases.

B. *Evaluation of Incentive Mechanism in Network*

We focus on Blockchain-based incentive mechanism in network in this subsection. Based on the *Bitcoinj* develop-library [53], we develop our work from the open sources code of this library and run it in the *Regression Test Mode*, and that means running the Blockchain-based network nodes in scenario locally in *JRE 1.8 (Java Runtime Environment)*. We set the detail values of our propositions shown in Table VI.

In ***CreditCoin***, each node represents a vehicle (i.e., OBU) or an RSU. The posted missions are managed in the cloud application server. Vehicles aggregate to forwarded announcements and exchange ***coins*** as incentives. The transactions in the network are sent to RSUs within the vehicles communication range. Then, the transactions wait to be voted by RSUs in the consensus phase. We record both the construction time of transactions and the transmission delay between vehicles and RSUs. We also find that relationship between vehicle density and transaction is not vitally significant unless the announcement is difficult to construct due to the insufficiency of witnesses.

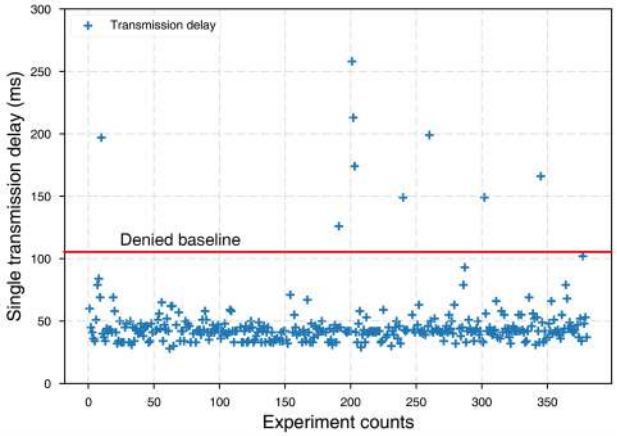


Fig. 7. Part of detailed experiment data in transmission delay of a single transaction.

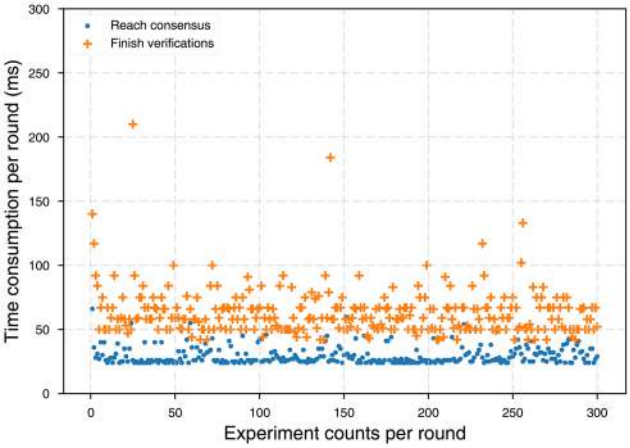


Fig. 8. Part of detailed experiment data in reaching a consensus and finishing verifications of transactions in a single round.

According to our simulation, the average construction time for a transaction is **45.57ms**. In addition, the average delay in transmission is **84.29ms** as shown in Table VII. We show the part of our detailed experimental data in Fig. 7. Particularly, it indicates that some of the transactions take rather longer time than average to transmit. These transactions tend to be denied in the consensus phase without causing any coin loss to users.

Finally, we implement the consensus phase in the same simulation, and RSUs are our consensus servers. When the transaction of vehicles is transferred to RSUs, the RSUs merge their candidate sets of transactions with others RSUs in their trusted list. For simplicity, we only defined four $UNLs$ for 40 RSUs without intersection. In each UNL, RSUs votes for 100 transactions of $\boldsymbol{CS'_{s_z}}$ in one consensus round. The criteria of votes are shown in Table VIII. The result is shown in Table VII; we give some experiment detailed results in Fig. 8, which shows good stability and efficiency running the consensus phase.

VIII. Conclusion

In this paper, we have proposed ***CreditCoin***, a novel privacy-preserving Blockchain-based incentive announcement network with our vehicular announcement protocol *Echo-Announcement* in VANETs.

Our announcement protocol maintains the reliability of announcements without revealing users' privacy and is reliable and efficient in the non-fully-trusted environment in VANETs. Through our simulations, the total time of announcements for a user only is 174ms in our assumptions, which is much more efficient than other protocols. Furthermore, the designed incentive mechanism encourages users to be active in responding. With Blockchain, the security is also enhanced since announcements and transactions are traced only by Trace manager in ***CreditCoin***. Through our simulations, the total time of transaction part for users is around 130ms per transaction, and the total time of consensus part for RSUs is around 92.4ms per 100 transactions. To conclude, ***CreditCoin*** is practical in the scenario of smart vehicles and smart transportation.

In future work, we plan to improve the key management and the coin balance in ***CreditCoin***. Designing more effective trading propositions is also being investigated.

TABLE VI
THE REWARD SETTINGS IN CREDITCOIN (COINS)

Reward types	Amount
Request Packet (RQP)	-10/per packet -50 at least
Reply Packet (RPP)	+5
Mission reward	100-500
Statement Date	Cut half

TABLE VII
THE AVERAGE SIMULATION TIME IN CREDITCOIN

Average time in	Amount
Construction per Transaction	45.57
Transmission per Transaction	84.29
Verification per consensus round	29.50
Vote per consensus round	62.9

TABLE VIII
THE CRITERIA OF VERIFICATION

No.	Criteria
1	Input and output validity check.
2	Serialized size is no larger than the max block size.
3	Outputs do not sum to larger than the max allowed quantity of coin in the system.

References

[1] L. Chen, S.-L. Ng, and G. Wang, "Threshold anonymous announcement in VANETs," *IEEE J. Sel. Areas Commun.,* vol. 29, no. 3, pp. 605-615, Feb. 2011.

[2] J. Shao, X. Lin, R. Lu, and C. Zuo, "A threshold anonymous authentication protocol for VANETs," *IEEE Trans. Veh. Technol.,* vol. 65, no. 3, pp. 1711-1720, Feb. 2016.

[3] S. Nakamoto. (2008, Bitcoin: A peer-to-peer electronic cash system. Available: https://bitcoin.org/bitcoin.pdf

[4] H. Hartenstein and L. Laberteaux, "A tutorial survey on vehicular ad hoc networks," *IEEE Commun. Mag.,* vol. 46, no. 6, June 2008.

[5] B. Parno and A. Perrig, "Challenges in securing vehicular networks," in *Proc. Workshop on hot topics in networks (HotNets-IV)*, Maryland, USA, Nov. 2005, pp. 1-6.

[6] F. Dötzer, "Privacy issues in vehicular ad hoc networks," in *Proc. Int. Workshop on Privacy Enhancing Technologies*, May 2005, pp. 197-209.

[7] J. R. Douceur, "The sybil attack," in *Proc. Int. Workshop on Peer-to-Peer Syst.* 2002, pp. 251-260.

[8] E. Bresson, J. Stern, and M. Szydlo, "Threshold ring signatures and applications to ad-hoc groups," in *Annu. Int. Cryptology Conference*, Aug. 2002, pp. 465-480.

[9] J. Ren and L. Harn, "An efficient threshold anonymous authentication scheme for privacy-preserving communications," *IEEE Trans. Wireless Commun.,* vol. 12, no. 3, pp. 1018-1025, Jan. 2013.

[10] M. Raya, A. Aziz, and J.-P. Hubaux, "Efficient secure aggregation in VANETs," in *Proc. of the 3rd Int. workshop on Veh. ad hoc networks*, Sep. 2006, pp. 67-75.

[11] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," in *Proc. Black Hat USA*, Aug. 2015.

[12] R. G. Engoulou, M. Bellaïche, S. Pierre, and A. Quintero, "VANET security surveys," *Comput. Commun.,* vol. 44, pp. 1-13, May 2014.

[13] B. Yu, C.-Z. Xu, and B. Xiao, "Detecting sybil attacks in VANETs," *J. of Parallel and Distributed Computing,* vol. 73, no. 6, pp. 746-756, June 2013.

[14] J. Petit and S. E. Shladover, "Potential Cyberattacks on Automated Vehicles," *IEEE Trans. Intell. Transp. Syst.,* vol. 16, no. 2, pp. 546-556, Apr. 2015.

[15] W. B. Jaballah, M. Conti, M. Mosbah, and C. E. Palazzi, "Fast and Secure Multihop Broadcast Solutions for Intervehicular Communication," *IEEE Trans. Intell. Transp. Syst.,* vol. 15, no. 1, pp. 433-450, Feb. 2014.

[16] G. Kounga, T. Walter, and S. Lachmund, "Proving reliability of anonymous information in VANETs," *IEEE Trans. Veh. Technol.,* vol. 58, no. 6, pp. 2977-2989, July 2009.

[17] Q. Wu, J. Domingo-Ferrer, and U. González-Nicolás, "Balanced trustworthiness, safety, and privacy in vehicle-to-vehicle communications," *IEEE Trans. Veh. Technol.,* vol. 59, no. 2, pp. 559-573, Oct. 2009.

[18] B. Qin, Q. Wu, J. Domingo-Ferrer, and W. Susilo. (2012). *Robust distributed privacy-preserving secure aggregation in vehicular communication*. Available: http://ro.uow.edu.au/cgi/viewcontent.cgi?article=1309&context=eispapers

[19] Y. Xia, W. Chen, X. Liu, L. Zhang, X. Li, and Y. Xiang, "Adaptive Multimedia Data Forwarding for Privacy Preservation in Vehicular Ad-Hoc Networks," *IEEE Trans. Intell. Transp. Syst.,* vol. PP, no. 99, pp. 1-13, Jan. 2017.

[20] L. Zhang, Q. Wu, B. Qin, and J. Domingo-Ferrer, "APPA: Aggregate privacy-preserving authentication in vehicular ad hoc networks," in *Proc. Int. Conference on Inform. Security*, Oct. 2011, pp. 293-308.

[21] X. Lin, "LSR: mitigating zero-day sybil vulnerability in privacy-preserving vehicular peer-to-peer networks," *IEEE J. Sel. Areas Commun.,* vol. 31, no. 9, pp. 237-246, Sep. 2013.

[22] M. Azees, P. Vijayakumar, and L. J. Deboarh, "EAAP: Efficient Anonymous Authentication With Conditional Privacy-Preserving Scheme for Vehicular Ad Hoc Networks," *IEEE Trans. Intell. Transp. Syst.,* vol. PP, no. 99, pp. 1-10, Feb. 2017.

[23] D. B. DeFigueiredo and E. T. Barr, "Trustdavis: A non-exploitable online reputation system," in *Proc. IEEE Int. Conference on E-Commerce Technology, 2005. CEC 2005* 2005, pp. 274-283.

[24] A. Ghosh, M. Mahdian, D. M. Reeves, D. M. Pennock, and R. Fugger, "Mechanism design on trust networks," in *Proc. Int. Workshop on Web and Internet Econ.* 2007, pp. 257-268.

[25] D. Schwartz, N. Youngs, and A. Britto. (2014, The Ripple protocol consensus algorithm. *Ripple Labs Inc White Paper 5*. Available: http://www.the-blockchain.com/docs/Ripple%20Consensus%20Whitepaper.pdf

[26] *Stellar Network*. Available: https://www.stellar.org/.

[27] A. Kate, "Introduction to Credit Networks: Security, Privacy, and Applications," in *Proc. of the 2016 ACM SIGSAC Conference on Comput. and Commun. Security*, Oct. 2016, pp. 1859-1860.

[28] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: Decentralized anonymous payments from bitcoin," in *Proc. IEEE Symp. on Security and Privacy*, May 2014, pp. 459-474.

[29] C. Garman, M. Green, and I. Miers. (2016, Accountable Privacy for Decentralized Anonymous Payments. *IACR Cryptology ePrint Archive 2016,* 61. Available: http://eprint.iacr.org/2016/061

[30] F. Armknecht, G. O. Karame, A. Mandal, F. Youssef, and E. Zenner, "Ripple: Overview and outlook," in *Proc. Int. Conf. on Trust and Trustworthy Computing*, Aug. 2015, pp. 163-180.

[31] M. Pease, R. Shostak, and L. Lamport, "Reaching agreement in the presence of faults," *J. of the ACM (JACM),* vol. 27, no. 2, pp. 228-234, Apr. 1980.

[32] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Trans. on Programming Languages and Syst. (TOPLAS),* vol. 4, no. 3, pp. 382-401, July 1982.

[33] A. Shamir, "How to share a secret," *Commun. of the ACM,* vol. 22, no. 11, pp. 612-613, Nov. 1979.

[34] A. Viejo, F. Sebé, and J. Domingo-Ferrer, "Aggregation of trustworthy announcement messages in vehicular ad hoc networks," in *Proc. 69th IEEE Veh. Technology Conference, 2009*, Apr. 2009, pp. 1-5.

[35] V. Daza, J. Domingo-Ferrer, F. Sebé, and A. Viejo, "Trustworthy privacy-preserving car-generated announcements in vehicular ad hoc networks," *IEEE Trans. Veh. Technol.,* vol. 58, no. 4, pp. 1876-1886, May 2009.

[36] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Proc. Workshop on the Theory and Applicat. of Cryptographic Techniques*, Apr. 1984, pp. 47-53.

[37] X.Nan, "Identity authentication based on combined public keys," in *Proc. Nat. Defense Ind. Press*, Beijing 2006.

[38] J. Liu and S. Zhong. (2009, Oct.). Fast Identity-Based Encryption Using Combined Public Keys. Available: http://www.paper.edu.cn/advanced_search/resultQuickSearch?type=0&judge=0&filename=Fast+Identity-Based+Encryption+Using+Combined+Public+Keys

[39] R. Zhang, J. Liu, Z. Han, and L. Zheng, "An IBE scheme using ECC combined public key," *Comput. & Elect. Eng.,* vol. 36, no. 6, pp. 1046-1054, Nov. 2010.

[40] S. Brands, "Untraceable off-line cash in wallet with observers," in *Annu. Int. Cryptology Conference*, Aug. 1993, pp. 302-318.

[41] R. Merkle, "A certified digital signature," in *Proc. Conference on the Theory and Application of Cryptology, CRYPTO '89* 1990, pp. 218-238.

[42] Z. Ning, L. Liu, F. Xia, B. Jedari, I. Lee, and W. Zhang, "CAIS: A copy adjustable incentive scheme in community-based socially-aware networking," *IEEE Trans. Veh. Technol.,* Apr. 2016.

[43] Z. Su, Q. Xu, M. Fei, and M. Dong, "Game theoretic resource allocation in media cloud with mobile social users," *IEEE Trans. Multimedia,* vol. 18, no. 8, pp. 1650-1660, May 2016.

[44] Y. Wang, M.-C. Chuah, and Y. Chen, "Incentive based data sharing in delay tolerant mobile networks," *IEEE Trans. Wireless Commun.,* vol. 13, no. 1, pp. 370-381, Dec. 2014.

[45] Q. Xu, Z. Su, and S. Guo, "A game theoretical incentive scheme for relay selection services in mobile social networks," *IEEE Trans. Veh. Technol.,* vol. 65, no. 8, pp. 6692-6702, Aug. 2016.

[46] A. M. Antonopoulos, *Mastering Bitcoin: unlocking digital cryptocurrencies*: " O'Reilly Media, Inc.", 2014.

[47] R. C. Merkle, "Protocols for public key cryptosystems," in *Proc. IEEE Symp. on Security and Privacy*, Nov. 1980, pp. 122-122.

[48] *PolarSSL Library*. Available: https://polarssl.org/

[49] *PBC Library*. Available: http://crypto.stanford.edu/pbc/

[50] D. H. Johnson, A. Menezes, and S. A. Vanstone, "The Elliptic Curve Digital Signature Algorithm (ECDSA)," *Int. J. of Inform. Security,* vol. 1, no. 1, pp. 36-63, Jan. 2001.

[51] A. Tomandl, D. Herrmann, K.-P. Fuchs, H. Federrath, and F. Scheuer, "VANETsim: An open source simulator for security and privacy concepts in VANETs," in *Proc. IEEE Int. Conference on High Performance Computing & Simulation (HPCS)*, July 2014, pp. 543-550.

[52] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Comput.,* vol. 7, no. 4, pp. 12-18, Oct. 2008.

[53] *bitcoinj: An open source Bitcoin client library built using Java and implements the Bitcoin network protocol.* Available: https://en.bitcoin.it/wiki/Bitcoinj

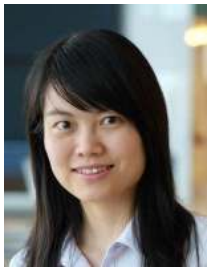
Lun Li received the B.E. degree from the School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China, in 2015, where he is currently pursuing the Ph.D. degree in information security at the Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, School of Computer and Information Technology. His current research interests are in the field of privacy-preserving and smart transportation.

Jiqiang Liu received his B.S. (1994) and Ph.D. (1999) degree from Beijing Normal University. He is currently a Professor at the School of Computer and Information Technology, Beijing Jiaotong University. He has published over 80 scientific papers in various journals and international conferences. His main research interests are trusted computing, cryptographic protocols, privacy-preserving and network security.

Lichen Cheng received the B.E. degree from the School of Computer and Information Technology, Beijing Jiaotong University, Beijing, China, in 2016, where she is currently pursuing the Ph.D. degree in Computer Science and Technology at the School of Computer and Information Technology. Her current research interests are in the field of information security and Blockchain technology.

Shuo Qiu received her B.S. degree in computer science from Anhui Normal University in 2011.She is a Ph.D. student in the School of Computer and Information Technology at Beijing Jiaotong University. She has worked in the Computer Science Department at Utah State University as a visiting scholar during Oct. 2014-Aug. 2015, and she has been a visiting scholar in the Department of Electrical and Computer Engineering at the University of Arizona from Sept.2015-Nov.2015. Her research interests are in cryptographic protocols, data security and privacy in cloud computing.

Wei Wang is currently associate professor in the Department of Information Security, Beijing Jiaotong University, China. He earned his Ph.D. degree in control science and engineering from Xi'an Jiaotong University, in 2006. He was a postdoctoral researcher in University of Trento, Italy, during 2005-2006. He was a postdoctoral researcher in TELECOM Bretagne and in INRIA, France, during 2007-2008. He was a European ERCIM Fellow in Norwegian University of Science and Technology (NTNU), Norway, and in Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, during 2009-2011. He visited INRIA, ETH, NTNU, CNR, and New York University Polytechnic. He has authored or co-authored over 60 peer-reviewed papers in various journals and international conferences. His main research interests include mobile, computer and network security.



Xiangliang Zhang is currently an associate professor and directs the Machine Intelligence and kNowledge Engineering (MINE) Laboratory (http://mine.kaust.edu.sa) in the Division of Computer, Electrical and Mathematical Sciences & Engineering, King Abdullah University of Science and Technology (KAUST). She earned her Ph.D. degree in computer science from INRIA-University Paris-Sud 11, France, in July 2010. She has authored or co-authored over 80 refereed papers in various journals and conferences. Her main research interests and experiences are in diverse areas of machine intelligence, knowledge engineering and their applications, such as information security and privacy.

Zonghua Zhang received the Ph.D. degree in information science from the Japan Advanced Institute of Science and Technology, in 2006. He is currently an Associate Professor with the Institut Mines-Télécom/TELECOM Lille, France. His research interests include anomaly detection, network forensics, and attacks mitigation in different types of computer and communication networks and service.