Threshold Anonymous Announcement in VANETs

Liqun Chen *Member, IEEE*, Siaw-Lynn Ng, and Guilin Wang

***Abstract*—Vehicular *ad hoc* networks (VANETs) allow wireless communications between vehicles without the aid of a central server. Reliable exchanges of information about road and traffic conditions allow a safer and more comfortable travelling environment. However, such profusion of information may allow unscrupulous parties to violate user privacy. On the other hand, a degree of auditability is desired for law enforcement and maintenance purposes. In this paper we propose a Threshold Anonymous Announcement service using direct anonymous attestation and one-time anonymous authentication to simultaneously achieve the seemingly contradictory goals of reliability, privacy and auditability.**

***Index Terms*—Vehicular communication, threshold verification, reliability, privacy, auditability.**

I. Introduction

VEHICULAR *ad hoc* networks (VANETs) allow wireless communications between vehicles and roadside infrastructures. By using a VANET, a vehicle can broadcast a piece of information to other vehicles without going through a central server. Let us consider the following scenario:

Scenario 1.1: Suppose that a car driver Bob receives a message from another vehicle reporting some traffic jam a few miles away. He has no idea whether the message is true or false. He attempts to ignore it. But shortly after that he receives several (say n) messages reporting the same traffic jam. If n is a reasonably large number and these n messages are sent by n different vehicles, then it is likely that this information is true, because it seems unlikely that any n vehicles would collude to lie. However, all these messages are sent anonymously, since nobody wants to disclose their identities. How can Bob find out whether n received messages are sent by n different legitimate vehicles without discovering the identities of these vehicles?

In order to resolve Bob's problem (illustrated in Figure 1), here we propose a novel solution, called "Threshold Anonymous Announcement (TAA)" service.

Clearly, exchanges of information about road and traffic conditions, if reliable, would enable a safer and more conducive travelling environment. On the other hand, such profusion of information may allow unscrupulous parties to track vehicles for profiling or more invidious purposes. The protection of user privacy is thus a matter of great importance.

Manuscript received 5 January 2010; revised 7 May and 12 July 2010.

L. Chen is with Hewlett Packard Labs, Bristol BS34 8QZ, United Kingdom (e-mail: liqun.chen@hp.com).

S.-L. Ng is with the Department of Mathematics, Royal Holloway, University of London, Egham, Surrey TW20 0EX, United Kingdom (e-mail: s.ng@rhul.ac.uk).

G. Wang is with the School of Computer Science and Software Engineering, University of Wollongong, NSW 2522, Australia. (e-mail: guilin@uow.edu.au)

Digital Object Identifier 10.1109/JSAC.2011.110310.

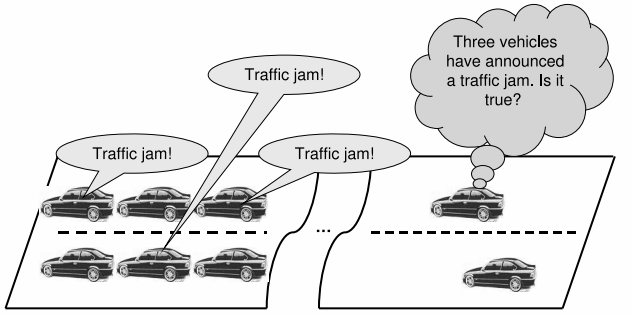

Fig. 1. A scenario of the TAA service.

However, it may also be desirable that rogue or malfunctioning vehicles can be identified for repair or for law enforcement purposes. There are thus seemingly contradictory requirements of privacy and auditability. Our aim is to create a TAA solution which satisfies the following three features simultaneously:

1) **Reliability**: If a vehicle accepts a reported event, the announcement was issued by a legitimate and present source (entity authentication) without unauthorized modification (data integrity). Furthermore, with a high probability, the event has actually happened (message truthfulness), and the service must also be able to tolerate a small fraction of internal adversaries (system robustness).
2) **Privacy**: A broadcast message cannot be bound to its source (anonymity), and different messages from one source cannot be linked to each other (unlinkability), conditional upon the source behaving properly.
3) **Auditability**: If the source is defective or malicious, it can be identified and rejected (revocation). A source cannot deny having sent a message (non-repudiation), and all transactions of a misbehaving source can be collated and tied to the source (traceability).

The rest of this paper is organized as follows. We first introduce related and prior work in this research topic in Section II, which will emphasize the motivation of this work. We then describe an overview of our contributions in Section III, and the proposed TAA scheme in details in Section IV. Then, we give an example to illustrate how the scheme can be used in Section V, analyse the security of our TAA scheme in Section VI, and present a comparison of performance between this scheme and a number of the existing schemes in Section VII. Finally, we conclude the paper in Section VIII.

II. Related and prior work

Many schemes have been proposed to address the issues of reliability, privacy and auditability with a variety of mechanisms, with differing emphasis and with varying degrees of success. Here we will focus only on research that are particularly relevant to providing reliability, privacy and auditability within a smiliar framework, and discuss the issues that arise. We refer the reader to the papers of Papadimitratos *et al*. ([17]) and Kounga *et al*. ([14]) for their excellent surveys and overviews on the wider topic.

The use of a threshold method to acheive reliability is a common approach in the literature ([11], [13], [14], [19]). Normally vehicles communicating with each other do not have a strong trust relation. In order to achieve some assurance that an announcement of a traffic event is true, the receiver only accepts a message which has been confirmed by a threshold number of vehicles. This threshold may be fixed system-wide ([11], [19]), or user-controlled ([13], [14]). It must be selected carefully, since too high a threshold may result in a vehicle not being able to collect a sufficient number of endorsements and therefore not being able to utilise the information, while too low a threshold would allow a small number of adversaries present to influence the decision. Various voting methods ([16]) may be adopted for decision-making by a verifier receiving many, possibly conflicting, messages.

Associated with the reliability of an announcement are the questions of integrity of the announcement and the authenticity of the announcer. These properties are often achieved by using a trusted third party (TTP) such as a certification authority (CA), who distributes credentials (public key certificates in most cases) to legitimate vehicles. These credentials are used to somehow sign an announcement ([8], [11], [14], [15], [17], [19]), which guarantees authentication and data integrity. In addition, timestamps can be used to assure freshness.

Another issue that arises from adopting a threshold method is the issue of distinguishability of origin: are two messages purporting to be from different sources indeed from different sources? If a single vehicle is able to send multiple messages pretending to come from different sources (the Sybil attack of [12]), then it could influence the acceptance of announcements, especially if the threshold is set low. In many publications (such as [11], [14], [17], [19]), digital signatures attached to announcements allow a receiver to check that they are from different vehicles. Such a method, if used naively, would compromise user privacy and allow linking of user activities. On the other hand, schemes achieving anonymity (such as [8], [15], using group signatures) tend to have difficulty allowing distinguishing of sources without the use of an online group manager.

User privacy generally includes two aspects, anonymity and unlinkability. The property of unlinkability prevents user tracking and profiling. A common mechanism used to provide this is the regular updating of a vehicle's credentials ([8], [14], [15], [17]). However, this does not altogether solve the problem. A vehicle may still be tracked within the validity period of a credential. A long validity period allows an adversary to gather substantial information, while a short one allows a rogue vehicle to send multiple messages without being detected. Moreover, the efficiency of updating credentials is also a concern.

Most schemes allow a CA or a group manager to reveal a signer's identity and hence allow linking in some circumstances. Otherwise, complete unlinkability may mean that there is no possibility for auditing and revocation. It is not clear that such a proposal would be acceptable to law enforcement authorities. Interestingly, in [18], it is said that US carmakers favour anonymity over liability.

Anonymity of vehicles can be provided by using pseudonyms which do not indicate their owners' identities. These pseudonyms may be individual signing keys ([11], [17], [14], although part of [14] has been shown to be insecure ([9])), or they may be group signing keys ([8], [15]), the group being all vehicles registered with the CA. Since the schemes based on group signatures are of the most interest to us in this work, we will examine further the schemes proposed in [8] and [15]. In both solutions, each vehicle is given an individual membership signing key and the group public key. The main difference is that in [15], a group signature (based on [3]) is used to sign an announcement directly, while in [8], a vehicle will instead generate a set of pseudonyms (public keys) and use its group membership signing key (based on [4]) to sign each of the pseudonyms. These new pseudonyms and associated secret keys are then used to sign announcements. Both solutions achieve anonymity within a group and only the group manager can reveal the signer's identity.

However, two main issues are not addressed in these two solutions: that of non-repudiation, and that of distinguishability of origin. In both solutions, the group signature schemes let the issuer create the private keys of group members. Thus the schemes cannot provide non-repudiation, since the signer is not the sole holder of the signing key. Group signature schemes intrinsically do not provide distinguishability of origin. Even though two signatures signed under the same short-term key is linkable in [8], a malicious user may still abuse the scheme by generating multiple such short-term keys simultaneously in order to pretend to be multiple vehicles. As a result, it is a challenge to achieve threshold verification of anonymous signatures. While the group manager is able to reveal the identities of the message senders in a dispute, it is most likely off-line in the normal run of events, and honest verifiers may not notice such a system abuse.

Our main goal is to provide a new TAA scheme which will provide distinguishability of origin without the intervention of any third party, and signer non-repudiation. Distinguishability of origin is a prerequisite for flexible threshold verification, by which we mean that each verifier can set up their own thresholds and make their own judgement case by case. Our TAA solution will achieve all the goals of reliability, privacy and auditability simultaneously.

III. Overview of the TAA solution

A. Context

We focus on the environment of vehicle-to-vehicle communication and will not assume the availability of roadside infrastructure. We assume that a vehicle is equipped with a tamper-resistant black box. This is a common approach

in VANET protocol design ([14], [17]). This black box has protected secure storage for secrets and a component which can perform cryptographic operations securely, as well as a secure time base. We assume that it will always operate correctly according to protocol, and will never disclose its secrets, even when its inputs are controlled by an adversary. Associated with each black box is an individual cryptographic key and its credential. This credential is a form of membership certificate, and differs from an ordinary public key certificate in that it is not made available to a signature verifier. It is used to testify that the black box is indeed in possession of the secret key and has been accredited by the system. It is created under a system secret key owned by a system issuer, and the corresponding system public key is available to all vehicles. Each announcement is generated by using the vehicle's key and credential and is verified by using the system public key.

The black box may also be preloaded with certain information (such as the issuer's public key), some of which may be downloaded to or updated by a trusted party at intervals. Trusted parties may be national transport authorities and vehicle manufacturers. This situation of having relatively few centralised authorities lend itself to public key infrastructures (see [21] for an overview and further references). Hence public keys and other public parameters may be securely distributed at manufacture or when vehicles are registered. Interactions between vehicles and trusted parties may be required at intervals for updating of public keys and credentials for revocation purposes or for tracing purposes. These may also be performed at maintenance time, or at designated service points. In the normal run of events the trusted parties would be offline.

B. Adversaries

In our paper we will adhere to the model in which adversaries constitute a relatively small fraction of the active vehicles at any one time ([13], [17]). They may be internal - essentially they may have in their possession some legitimate black boxes. However, while they may make vehicles and black boxes form and broadcast any messages they choose, the black boxes will always perform their internal operations correctly. For example, an adversary may input an invalid message to the black box, and the black box will generate random numbers and perform other necessary operations correctly and return a correct signature on that invalid message. The aim of an adversary would be to make announcements that would be accepted by other vehicles and thereby mislead them, or to track other vehicles.

We will also assume that the authorities or any part of the VANET infrastructure apart from the vehicles are "honest-but-curious", that is, they are passive and would eavesdrop and gather information but would not launch any active attacks. They would also execute all protocols correctly. The privacy of a well-behaved user should also be protected against these parties, in the sense that its legitimate activities should not be traceable by them.

C. Our solution

Based on techniques used in direct anonymous attestation (DAA) ([10]) and k-time anonymous authentication ([22]), we propose a new solution to address the issues of non-repudiation and distinguishability of origin in the context of user privacy.

The original DAA scheme ([5]) was adopted in 2003 by the Trusted Computing Group ([23]) as a mechanism for a user to anonymously convince a verifier that the user's computer platform is accredited by a trusted authority. Recently a number of more efficient DAA schemes based on bilinear maps have been proposed (for example, [6], [7], [10]). DAA schemes can also be seen as group signature schemes without the functionality of tracing a signature back to its signer, and signer privacy is thus enhanced. The property of non-repudiation is achieved since a user's secret key is not disclosed to any authority. In addition, DAA schemes allow user-controlled linkability of signatures. We make use of this feature to provide distinguishability of origin by allowing the linking of signatures if a signer signs a message with the same content twice.

A k-time anonymous signature scheme (such as [22]) allows a signer's identity to be revealed if the signer signs the same message more than k times. For the purpose of the TAA solution, we need an anonymous signature scheme which can optionally achieve one of the two properties: when a signer signs the same message twice, either the two signatures are linked while the signer remains anonymous, or the signer's identity is revealed. We will adapt the k-time anonymous authentication scheme of [22] to be used in conjunction with the DAA scheme of [10] to create our TAA scheme. We keep these two optional properties as two separate versions in the TAA scheme. Other issues arising in the use of anonymous signature schemes are those of revocation and auditability. The DAA scheme of [10] allows for an efficient revocation scheme, which we will describe in Section IV-I.

We note that neither the direct anonymous attestation scheme of [10] nor the one-time anonymous signature scheme of [22] would satisfy all the goals of a TAA service on its own. In the next section we will present an adaptation and amalgamation of both schemes, consolidated by the addition of three new algorithms, to provide a comprehensive service that achieves all the desired goals of reliability, privacy and auditability.

IV. Description of the TAA scheme

We now give a detailed description of our new TAA scheme. The TAA scheme includes the Setup, Sign, Verify, ThresholdCheck, Link and Trace algorithms and the Join and Disavow protocols, where Setup creates system parameters and long-term keys, the Join protocol allows each legitimate user to get its credential, Sign/Verify signs a message anonymously and verifies such an anonymous signature, ThresholdCheck checks whether a message has been signed by a certain number of independent users, Link shows whether two anonymous signatures on the same message are from the same signer, Trace finds the true signer if this signer has signed the same message twice, and the Disavow protocol is used to prove whether a signer and a signature are cryptographically bound if such a signer is required to do so (for instance, in the setting of law enforcement). These algorithms

and protocols will be described in the following subsections separately.

In the specification of these protocols and algorithms, we will use standard notation as follows. If S is any set then we denote the action of sampling an element from S uniformly at random and assigning the result to the variable x as $x \leftarrow S$. If A is any algorithm then we denote the action of obtaining x by running A on inputs $y_1, \ldots, y_n$ as $x \leftarrow A(y_1, \ldots, y_n)$. We denote the concatenation of two strings x and y as $x \| y$, and scalar multiplication of an integer a and a group element P as $a \cdot P$, while multiplication of two integers a and b is denoted ab. We write $\{0,1\}^t$ for the set of binary strings of length t and $\{0,1\}^*$ for the set of binary strings of arbitrary length.

The entities in the TAA scheme are called the *players*. There are three sets of players in our scheme: the set $\mathcal{I}$ of *issuers*, the set $\mathcal{S}$ of *signers* and the set $\mathcal{V}$ of *verifiers*. In our case the set $\mathcal{S}$ of signers coincides with the set $\mathcal{V}$ of verifiers and both are vehicles, while the set $\mathcal{I}$ of issuers is part of the trusted authorities (such as vehicle manufacturers and transport authorities).

A. The setup algorithm

To set the system up we need to select public parameters for all protocols and algorithms within the TAA scheme as well as the long term parameters for each issuer. As described in Section III-A, each signer/verifier is equipped with a tamper-resistant black box, which is a TPM-type device. (A TPM (Trusted Platform Module) is a tamper-resistant cryptographic chip as specified in [23].) We assume that prior to any system setup each black box has its private endorsement key SK embedded into it and that each issuer has access to the corresponding public endorsement key PK. We also assume a secure public key signature scheme has been selected for use with these keys, with signing algorithm $\mathrm{sig}_{\mathsf{SK}}$ and verification algorithm $\mathrm{ver}_{\mathsf{PK}}$. In addition, each black box would also generate an internal secret $\mathsf{TAAseed}$ specific to itself. This value would be stored securely within the black box and would never be disclosed or changed, as the black box is assumed to be tamper-resistant.

On input of an integer t (note that the size of this value is dependent on the security strength of the scheme), the setup algorithm executes the following:

1) At first, three cyclic groups, $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{G}_T$, of sufficiently large prime order q ($q \approx 2^t$) are selected. Two random generators are selected such that $\mathbb{G}_1 = \langle P_1 \rangle$ and $\mathbb{G}_2 = \langle P_2 \rangle$ along with a pairing $\hat{t} : \mathbb{G}_1 \times \mathbb{G}_2 \mapsto \mathbb{G}_T$. We write $\mathbb{G}_1$, $\mathbb{G}_2$ additively and $\mathbb{G}_T$ multiplicatively. The pairing $\hat{t}$ is a map from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$ such that
   (i) $\hat{t}$ is *bilinear*, that is, $\hat{t}(a \cdot P_1, b \cdot P_2) = \hat{t}(P_1, P_2)^{ab}$ for any two integers a and $b \in \mathbb{Z}_q$.
   (ii) $\hat{t}$ is *non-degenerate*, that is, $\hat{t}(P_1, P_2) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is the identity element of $\mathbb{G}_T$.
   (iii) $\hat{t}$ is *computable*, that is, there is a polynomial time algorithm for computing $\hat{t}(P, Q)$ for any $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.

   Note that the groups are selected so that solving the decisional Diffie–Hellman (DDH) problem in $\mathbb{G}_1$, the Gap Discrete Logarithm (Gap-DL) problem in $\mathbb{G}_1$ and $\mathbb{G}_2$ and the blind bilinear LRSW problem are hard. We shall give an informal description of these computational assumptions in Section VI.
2) Secondly, six hash functions are selected: $H_1 : \{0,1\}^* \mapsto \mathbb{Z}_q$ and $H_2 : \{0,1\}^* \mapsto \mathbb{Z}_q$, $H_3 : \{0,1\}^* \mapsto \mathbb{G}_1$, $H_4 : \{0,1\}^* \mapsto \mathbb{Z}_q$, $H_5 : \{0,1\}^* \mapsto \mathbb{G}_1$ and $H_6 : \{0,1\}^* \mapsto \mathbb{Z}_q$. Note that the hash-functions H_5 and H_6 are used to achieve the property of one-time anonymity. Note also that in implementation, the same hash function can be used for H_1, H_2, H_4 and H_6, and similarly for H_3 and H_5
3) Thirdly, for each issuer $\mathsf{i} \in \mathcal{I}$ the following is performed. Two integers are selected $x, y \leftarrow \mathbb{Z}_q$ and the issuer secret key isk is assigned to be (x, y). Then the values $X = x \cdot P_2 \in \mathbb{G}_2$ and $Y = y \cdot P_2 \in \mathbb{G}_2$ are computed. The issuer public key ipk is assigned to be (X, Y).
4) Finally, the system public parameters par are set to be $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{t}, P_1, P_2, q, H_1, H_2, H_3, H_4, H_5, H_6, \mathsf{ipk}_k)$ and are published.

B. The join protocol

This is a protocol between a given signer $\mathfrak{s} \in \mathcal{S}$ and an issuer $\mathsf{i} \in \mathcal{I}$. This protocol is identical to the Join protocol of the DAA scheme ([10]), except that we do not spilt the signer role between the TPM and computer platform, since in our application each vehicle has a simple computing device instead of a comprehensive computer. Intuitively, the Join protocol (shown in Figure 2) proceeds as follows. First a signer $\mathfrak{s}$ generates a secret value f using its internal seed $\mathsf{TAAseed}$, along with the value K_I provided by i and a count number cnt. We note that a signer could compute many values of f, one for each data string of "$\mathsf{cnt} \| \mathrm{K}_I$"[1]. The signer $\mathfrak{s}$ then computes a commitment comm on the value f and sends comm to the issuer i. The value comm essentially binds the signer's secret value f, the signer's long-term endorsement key SK and the issuer's nonce n_I all together. Upon receipt of comm, the issuer i performs some checks on it, including a comparison to the rogue list, which consists of the values of all exposed fs for those already compromised vehicles. If it verifies correctly, the issuer i is sure that the signer has prepared the commitment comm correctly and thus computes a credential $\mathsf{cre} = (A, B, C)$ and sends it back to the signer $\mathfrak{s}$, who then verifies the correctness of cre by performing a few pairing evaluations. Hereafter, the issuer can use its valid credential cre to sign any message anonymously.

C. The signing algorithm

This is an algorithm run by a signer $\mathfrak{s} \in \mathcal{S}$ to produce a signature on some message. Specifically, the signature should prove the knowledge of a discrete logarithm f and the knowledge of a valid credential cre corresponding to the value f without revealing either value.

Let msgt and msgb denote the title and main body, respectively, of the message to be signed. The title is specific to an

[1]This creates the possibility that a vehicle may use a different value of f for different types of announcements.

Signer (s)		*Issuer* (i)
		$n_I \leftarrow \{0,1\}^*$
$\mathsf{str} \leftarrow X \| Y \| n_I$	$\xleftarrow{\mathsf{comm}_{\mathrm{req}}}$	$\mathsf{comm}_{\mathrm{req}} \leftarrow n_I$
$f \leftarrow H_1(\mathsf{TAAseed} \| \mathsf{cnt} \| \mathrm{K}_I)$		
$u \leftarrow \mathbb{Z}_q$; $U \leftarrow u \cdot P_1$; $F \leftarrow f \cdot P_1$		
$v \leftarrow H_2(\mathsf{str} \| F \| U)$		
$w \leftarrow u + vf \pmod q$		
$\gamma \leftarrow \mathrm{sig}_{\mathsf{SK}}(F \| v \| w)$		
$\mathsf{comm} \leftarrow (F, v, w, \gamma)$	$\xrightarrow{\mathsf{comm}, n_I}$	If $n_I \notin \{\mathsf{comm}_{\mathrm{req}}\}$ then **abort**
		If $\mathrm{ver}_{\mathsf{PK}}(\gamma, (F \| v \| w)) = false$ then **abort**
		$U' \leftarrow w \cdot P_1 - v \cdot F$; $\mathsf{str} \leftarrow X \| Y \| n_I$
		If $F = f_i \cdot P_1$ for any f_i on the rogue list, or
		$v \neq H_2(\mathsf{str} \| F \| U')$ then **abort**
		$r \leftarrow \mathbb{Z}_q$; $A \leftarrow r \cdot P_1$; $B \leftarrow y \cdot A$
If $\hat{t}(A, Y) \neq \hat{t}(B, P_2)$	$\xleftarrow{\mathsf{cre}}$	$C \leftarrow (x \cdot A + rxy \cdot F)$; $\mathsf{cre} \leftarrow (A, B, C)$
or $\hat{t}(A + f \cdot B, X) \neq \hat{t}(C, P_2)$		
then **abort**		

Fig. 2. The Join Protocol

Signer (s)
Input: $f \in \mathbb{Z}_q$; $n_T \leftarrow \{0,1\}^*$; $\mathsf{msg} = (\mathsf{msgt}, \mathsf{msgb})$,
$\mathsf{msgt} \in \{0,1\}^*$, $\mathsf{msgb} \in \{0,1\}^*$
$a \leftarrow \mathbb{Z}_q$; $z \leftarrow \mathbb{Z}_q$
$J \leftarrow H_3(\mathsf{msgt})$; $K = f \cdot J$; $L \leftarrow z \cdot J$
$R \leftarrow a \cdot A$; $S \leftarrow a \cdot B$; $T \leftarrow a \cdot C$; $\tau \leftarrow \hat{t}(S, X)^z$
Version 2 only: $M \leftarrow H_5(\mathsf{msgt}) + H_6(n_T \| \mathsf{msgb} \| L \| R \| S \| T) \cdot P_1$;
$N = f \cdot M$; $O \leftarrow z \cdot M$
Version 1: $c \leftarrow H_4(R \| S \| T \| \tau \| J \| K \| L \| n_T \| \mathsf{msgb})$
Version 2: $c \leftarrow H_4(\tau \| J \| K \| M \| N \| O \| n_T \| \mathsf{msgb})$
$s \leftarrow z + c \cdot f \pmod q$
Version 1: $\sigma \leftarrow (R, S, T, J, K, c, s, n_T)$
Version 2: $\sigma \leftarrow (R, S, T, J, K, M, N, c, s, n_T)$
Output: σ

Fig. 3. The Sign Algorithm

event and the main body is an arbitrary data string. Let n_T denote a data string which contributes a verifiable randomness to the signature. It could be a time stamp, a series number or a random number.

The algorithm proceeds as in Figure 3. This algorithm is a modification of the Sign algorithm of [10]. Intuitively, the signer first randomizes his original credential $\mathsf{cre} = (A, B, C)$ to get a temporary but also valid credential $\mathsf{cre}' = (R, S, T)$. Then, by using the knowledge of f the signer can show that both $\mathsf{cre}' = (R, S, T)$ and the pair (J, K) correspond to the same secret f without revealing the value of f itself - a non-interactive zero knowledge proof ([21]). Given two (temporary) credentials, it is impossible to determine whether they are produced by the same signer, or to identify the signer. Consequently, anonymity is achieved (see more discussion in Section VI).

For the purposes of our TAA service, we specify two versions of the signing algorithm. In the first version the arbitrary basename of the DAA scheme is replaced by the meaningful event title msgt, and the algorithm follows the principle privacy property of DAA: unless the private key f is revealed or compromised, signer anonymity holds regardless of the signer's behaviour. If the signer signs a single event (that is, a single msgt value) more than once, these signatures can be linked to each other, since they will contain the same values of a base point J and its point multiplication K. However the signer's identity will still be hidden from the verifier or the issuer or both of them colluded. In the second version, we add a novel "*misbehaviour traceability*" property, which is not offered in [10], [22] or, to our best knowledge, any other existing work. That is, if a signer produces more than one distinct signatures on a single event, not only can these signatures be linked, but also the identity of the signer (*i.e.* the value $F = f \cdot P_1$) can be calculated from the signatures. The detailed procedure of tracing will be specified in the algorithm Trace.

D. The verification algorithm

This is an algorithm run by a verifier $\mathfrak{v} \in \mathcal{V}$ to check that a given signature σ does prove the knowledge of a discrete logarithm f as well as the knowledge of a valid credential issued on the same value of f, and that this value of f is not on the list of rogue values. The algorithm proceeds as in Figure 4. To follow the two versions of the Sign algorithm, we also specify the corresponding two versions in the Verify algorithm. This algorithm is modified from the Verify algorithm of [10] to reflect the modifications and extensions in the Sign algorithm that we have made.

E. The threshold checking algorithm

This algorithm, run by a given verifier $\mathfrak{v} \in \mathcal{V}$, is newly developed for our TAA solution and it brings us a unique property of light-weight and flexible threshold verification.

Each verifier maintains a list $\mathcal{B}$ of l events and associated information, $\mathcal{B} = \{B_i\}_{i=1}^{l}$. The information associated with an event B_i includes the event description event_i, an expiry time exp_i, and a collection of k_i signed announcements of the event $(\sigma_i^j, \mathsf{msg}_i^j)$, $1 \leq j \leq k_i$. There may also be a threshold t_i associated with B_i, or t_i may be a constant across all B_i's.

Verifier ($\mathfrak{v}$)

Input: $\mathsf{ipk}_k = (X, Y)$; $\mathsf{msg} = (\mathsf{msgt}, \mathsf{msgb})$

Version 1: $\sigma = (R, S, T, J, K, c, s, n_T)$

Version 2: $\sigma = (R, S, T, J, K, M, N, c, s, n_T)$

If $K = f_i \cdot J$, for any f_i in the set of rogue secret keys, or

$\hat{t}(R, Y) \neq \hat{t}(S, P_2)$, or

$J \neq H_3(\mathsf{msgt})$ return **reject**

$\rho_a^\dagger \leftarrow \hat{t}(R, X)$; $\rho_b^\dagger \leftarrow \hat{t}(S, X)$; $\rho_c^\dagger \leftarrow \hat{t}(T, P_2)$

$\tau^\dagger \leftarrow (\rho_b^\dagger)^s \cdot (\rho_c^\dagger / \rho_a^\dagger)^{-c}$

$L^\dagger \leftarrow s \cdot J - c \cdot K$

Version 2 only: If $M \neq H_5(\mathsf{msgt}) + H_6(n_T \| \mathsf{msgb} \| L^\dagger \| R \| S \| T) \cdot P_1$

return **reject**

Version 2 only: $O^\dagger \leftarrow s \cdot M - c \cdot N$

Version 1: If $c \neq H_4(R \| S \| T \| \tau^\dagger \| J \| K \| L^\dagger \| n_T \| \mathsf{msgb})$ return **reject**

Version 2: If $c \neq H_4(\tau^\dagger \| J \| K \| M \| N \| O^\dagger \| n_T \| \mathsf{msgb})$ return **reject**

Otherwise return **accept**

Fig. 4. The Verify Algorithm

Verifier ($\mathfrak{v}$)

Input: (σ, msg), $\mathcal{B} = \{B_i\}_{i=1}^{l}$

$\sigma = (R, S, T, J, K, (M, N), c, s, n_T)$, $\mathsf{msg} = (\mathsf{msgt}, \mathsf{msgb})$

$B_i = (\mathsf{event}_i, \mathsf{exp}_i, t_i, \{\sigma_i^j = (\ldots K_i^j \ldots), \mathsf{msg}_i^j\}_{j=1}^{k_i})$

If **reject** $\leftarrow \mathsf{Verify}(\sigma, \mathsf{msg})$ return **reject**

For $i = 1$ to l

{If $\mathsf{event}_i = \mathsf{msgt}$ then

For $j = 1$ to k_i

{ If $\sigma = \sigma_i^j$ return **reject**

If $K = K_i^j$ then $\mathsf{action}(\mathsf{log}, \mathsf{trace})$; return **reject**

}

If $k_i = t_i - 1$ then

$\mathsf{action}(\mathsf{respond}, \mathsf{log}, \mathsf{endorse}, \mathsf{remove})$; return **accept**

$k_i \leftarrow k_i + 1$; $(\sigma_i^{k_i}, \mathsf{msg}_i^{k_i}) \leftarrow (\sigma, \mathsf{msg})$;

$\mathsf{action}(\mathsf{endorse})$; return **accept**

}

If $l < \mathsf{cap}$

$B_{l+1} \leftarrow (\mathsf{msgt}, \mathsf{exp}_{l+1}, t_{l+1}, \{\sigma, \mathsf{msg}\})$; $\mathcal{B} \leftarrow \mathcal{B} \cup \{B_{l+1}\}$; $l \leftarrow l + 1$

$\mathsf{action}(\mathsf{endorse})$; return **accept**

return **reject**

Fig. 5. The ThresholdCheck Algorithm

Verifier ($\mathfrak{v}$)

Input: $(\sigma_b, \mathsf{msg}_b)$, $\mathsf{msg}_b = (\mathsf{msgt}_b, \mathsf{msgb}_b)$, $b = 0, 1$

If $\mathsf{msgt}_0 \neq \mathsf{msgt}_1$ return $\perp$

If **reject** $\leftarrow \mathsf{Verify}(\sigma_b, \mathsf{msg}_b)$ for any $b \in \{0, 1\}$ return $\perp$

If $J_0 = J_1$ and $K_0 = K_1$ return **linked**

Otherwise return **unlinked**

Fig. 6. The Link Algorithm

We assume that an external program periodically examines the list and deletes B_i when it expires.

When a signed announcement (σ, msg) is received, the verifier checks that the signature is valid, and then checks the signed event against the list $\mathcal{B}$. If the event announced has not been reported before, then a new event B_{l+1} is created and appended to $\mathcal{B}$, as long as memory capacity cap has not been exceeded. Otherwise the message is simply dropped. If there is already an entry B_i for the announced event, then the signature is checked against the list of signatures already received to ensure it is not a duplicate message signed by some signer again. If it is, it can be logged and rejected. If it is not, it is added to the list.

When the number of signed announcements reporting this event reaches the threshold t_i, action is taken. The action procedure is an external procedure which may be log, endorse, respond, trace or remove: log denotes the recording of a signed announcement, endorse denotes the generating of a signed announcement on the relevant event, respond denotes the taking of appropriate actions in response to the event, trace denotes invoking the Trace algorithm, and remove denotes the removal of a B_i either because it has expired, or because action has been taken. We will use $\mathsf{action}(\mathsf{log}, \mathsf{endorse}, \ldots)$ to denote the actions that the program might take at various points of the threshold checking process. More details of this procedure can be considered in implementations. The algorithm proceeds as in Figure 5.

F. The linking algorithm

This algorithm, run by a given verifier $\mathfrak{v} \in \mathcal{V}$, is identical to the Link algorithm of [10]. Given a pair of signatures for the same event (*i.e.* with the same value of msgt), the verifier executes the Link algorithm (illustrated in Figure 6) to check if the pair of signatures were produced by the same signer.

G. The tracing algorithm

This algorithm, run by a given verifier $\mathfrak{v} \in \mathcal{V}$, is newly developed for our TAA solution. It is only applicable if Version 2 of the Sign algorithm was used, as Version 1 is not designed to support tracing. Intuitively, given a pair of signatures with the same value of msgt, the verifier checks if the pair of signatures were produced by the same signer. If yes, the verifier computes the identity of the signer, written as (F, f), from the two signatures. The algorithm proceeds as in Figure 7. Note that the value F is sufficient to identify the signer and that if $f \neq \perp$, this value will be put in the rogue list.

H. The disavowing protocol

This protocol, run between a given signer $\mathfrak{s} \in \mathcal{S}$ and a verifier $\mathfrak{v} \in \mathcal{V}$, is also newly developed for our TAA solution and it brings us the property of non-repudiation. Intuitively, given a signature σ signed on $\mathsf{msg} = (\mathsf{msgt}, \mathsf{msgb})$, where **accept**$\leftarrow\mathsf{Verify}(\sigma, \mathsf{msg})$ holds, this protocol determines whether the signer $\mathfrak{s}$ is the signer of σ or not. The verifier $\mathfrak{v}$ creates a nonce n'_T and sends it to the signer $\mathfrak{s}$ as a challenge, and $\mathfrak{s}$ responds with a signature σ' signed on (msg, n'_T). If **reject**$\leftarrow\mathsf{Verify}(\sigma', \mathsf{msg})$, $\mathfrak{v}$ concludes that $\mathfrak{s}$ is attempting to disavow. If **accept**$\leftarrow\mathsf{Verify}(\sigma', \mathsf{msg})$ and **unlinked**$\leftarrow\mathsf{Link}(\sigma, \sigma')$, $\mathfrak{v}$ concludes that $\mathfrak{s}$ is not the signer of σ. If **accept**$\leftarrow\mathsf{Verify}(\sigma, \mathsf{msg})$ and **linked**$\leftarrow\mathsf{Link}(\sigma, \sigma')$, $\mathfrak{v}$ concludes that $\mathfrak{s}$ is the signer of σ. If the second version of the Sign algorithm is used, $\mathfrak{v}$ can further retrieve the signer's identity by running the Trace algorithm instead of the Link algorithm, to obtain $F \leftarrow \mathsf{Trace}(\sigma, \sigma')$. Our protocol proceeds as shown in Figure 8.

Verifier ($\mathfrak{v}$)

Input: $(\sigma_b, \mathsf{msg}_b)$, $\mathsf{msg}_b = (\mathsf{msgt}_b, \mathsf{msgb}_b)$, $b = 0, 1$
$\sigma_b = (R_b, S_b, T_b, J_b, K_b, M_b, N_b, c_b, s_b, (n_T)_b)$,
If $\mathsf{msgt}_0 \neq \mathsf{msgt}_1$ return $\perp$
If **reject**$\leftarrow \mathsf{Verify}(\sigma_b, \mathsf{msg}_b)$ for any $b \in \{0, 1\}$ return $\perp$
If $K_0 \neq K_1$ return **unlinked**
$L_b^\dagger \leftarrow s_b \cdot J_b - c_b \cdot K_b$
$h_0 \leftarrow H_6((n_T)_0 \| \mathsf{msgb}_0 \| L_0^\dagger \| R_0 \| S_0 \| T_0)$
$h_1 \leftarrow H_6((n_T)_1 \| \mathsf{msgb}_1 \| L_1^\dagger \| R_1 \| S_1 \| T_1)$
If $h_0 = h_1$ returns $\perp$ (i.e., $(\sigma_0, \mathsf{msg}_0)$ and $(\sigma_1, \mathsf{msg}_1)$ are the same)
If $L_0^\dagger = L_1^\dagger$ and $s_0 \neq s_1$, $f \leftarrow (s_0 - s_1)/(c_0 - c_1)$; otherwise $f \leftarrow \perp$
$F \leftarrow (N_0 - N_1)/(h_0 - h_1)$
Output (F, f)

Fig. 7. The Trace Algorithm

I. A note on revocation

In our TAA scheme, as in any system supporting revocation lists, the longer the rogue list is, the worse the practical efficiency of the Verify algorithm will be. To avoid the situation where the revocation list becomes too large, we should consider the permanent revocation of the revealed or compromised credentials. A simple scheme for revocation by updating issuers' keys and signers' credentials is described in [10]. We give a brief description here. The issuer $\mathfrak{i}$ has private key $\mathsf{isk} = (x, y)$ and public key $\mathsf{ipk} = (X = x \cdot P_2, Y = y \cdot P_2)$. The signer $\mathfrak{s}$ has secret f with commitment value $F = f \cdot P_1$. Its corresponding credential issued by $\mathfrak{i}$ is $\mathsf{cre} = (A = r \cdot P_1, B = y \cdot A, C = x \cdot A + rxy \cdot F)$. During the Join Protocol, the issuer would typically store F together with the credential cre, or at least C, the last part of cre. To update its keys, the issuer $\mathfrak{i}$ first updates its secret key x to a new value $\dot{x}$. Let $\beta = \dot{x}/x$. The issuer then updates its public key by replacing X by $\dot{X} = \beta \cdot X$. Now, for each currently legitimate signer $\mathfrak{s}$ with credential $\mathsf{cre} = (A, B, C)$, $\mathfrak{i}$ replaces C with $\dot{C} = \beta \cdot C$. The issuer then publishes its new public key $(\dot{X}, Y)$ and makes $\dot{C}$ available to $\mathfrak{s}$. The signers to be revoked will not have their credentials updated and hence their signatures will not verify correctly under the new public key $(\dot{X}, Y)$.

V. Example

We give an example of how the various components of our TAA scheme can be used to resolve Bob's problem in Scenario 1.1. For simplicity, here we only use Version 1 of our TAA scheme. Suppose Bob, driving along in his car, receives a message $(\mathsf{msg}_0, \sigma_0)$, where

$$\begin{aligned} \mathsf{msg}_0 &= (\mathsf{msgt}_0 = \mathrm{traffic\_jam}\mathit{LocTimeDate}, \mathsf{msgb}_0 = \perp), \\ \sigma_0 &= (R_0, S_0, T_0, J_0, K_0, c_0, s_0, (n_T)_0). \end{aligned}$$

Bob realises that a vehicle is reporting a traffic jam at location *Loc*, time *Time* and date *Date*. He (or rather, the black box BB_B associated with his vehicle) uses the ThresholdCheck algorithm to process the message. Firstly Bob verifies that the signature is valid using the Verify algorithm. If it is not valid Bob rejects the announcement. If it is valid Bob checks whether there have already been signed announcements reporting the traffic jam. If it has not been reported, then a new list can be created for it. If there have already been announcements, Bob checks for duplicates by running the Link algorithm. If it transpires that the vehicle that made this announcement has already reported the event msgt_0 before, then there would be another signature σ_0' recorded that would contain $K_0' = f_0 \cdot J_0'$ where $J_0' = H_3(\mathsf{msgt}_0) = J_0$. Hence $K_0' = K_0$, and the ThresholdCheck algorithm would log and reject this announcement.

If this announcement is legitimate, Bob waits to see if there are more reports on this event. Subsequently he receives n more messages $(\mathsf{msg}_i, \sigma_i)$, $1 \leq i \leq n$, where

$$\mathsf{msg}_i = (\mathsf{msgt}_0, \mathsf{msgb}_i), \sigma_i = (R_i, S_i, T_i, J_i, K_i, c_i, s_i, (n_T)_i).$$

As Bob receives each message, he uses the ThresholdCheck algorithm to process the message as described above. If the number of announcements of the event reaches the predetermined threshold, Bob believes that the reported event is true and takes appropriate action, such as avoiding the route with the traffic jam. If the threshold is not reached, the announcements are added to the list. In either case Bob may wish to add his own endorsement of the event msgt_0. To do this, his black box BB_B would form $\mathsf{msg}_B = (\mathsf{msgt}_0, \mathsf{msgb}_B)$, where msgb_B may consist of all the previous announcements he had received, or it may be left blank. A signature σ_B will then be generated using the Sign algorithm and $(\mathsf{msg}_B, \sigma_B)$ will be broadcast.

If the reported traffic jam is not true then it is unlikely that there will be sufficient number of vehicles endorsing the event to reach the threshold. The list containing the false announcement would very likely remain below threshold until it is removed at expiry time. Bob continues with his journey without taking any action.

VI. Security analysis

In this section, we informally show that our TAA scheme satisfies the three security requirements of reliability, privacy, and auditability as described in Section I. Before starting to present the security analysis, we first give an informal description of the computational assumptions on which the security of the TAA scheme is based. For formal definitions and further details we refer the readers to [10].

The **blind bilinear LRSW assumption** is described as follows. Let $\mathbb{G}_1 = \langle P_1 \rangle$, $\mathbb{G}_2 = \langle P_2 \rangle$ and $\mathbb{G}_T$ be cyclic groups of a prime order q, and let $\hat{t} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a pairing. Let $X = x \cdot P_2$, $Y = y \cdot P_2$ for some random $x, y \in \mathbb{Z}_q$. Suppose there is an oracle (*i.e.* a helping algorithm) that, on input $F \in \mathbb{G}_1$, outputs a triple $(A, y \cdot A, x \cdot A + rxy \cdot F)$ where $A = r \cdot P_1$ for some random $r \in \mathbb{Z}_q$. Then there is no efficient algorithm that queries the oracle a polynomial number of times and outputs (f, A, B, C) such that $F = f \cdot P_1$, $A \in \mathbb{G}_1$, $B = y \cdot A$, $C = x \cdot A + fxy \cdot A$, where $F = f \cdot P_1$ has not been queried before.

The **decisional Diffie–Hellman (DDH) assumption** in the group $\mathbb{G}_1 = \langle P_1 \rangle$ is as follows. For any polynomial algorithm, it is hard to tell a triple $(X = x \cdot P_1, Y = y \cdot P_1, Z = z \cdot P_1) \in \mathbb{G}_1^3$ from a triple $(X = x \cdot P_1, Y = y \cdot P_1, Z = xy \cdot P_1) \in \mathbb{G}_1^3$, where $x, y, z \in \mathbb{Z}_q$ are all unknown random numbers. Namely, $xy \cdot P_1$ looks like a random element in $\mathbb{G}_1$, given $x \cdot P_1$, $y \cdot P_1$ and the related pairing $\hat{t}$ from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$. More accurately,

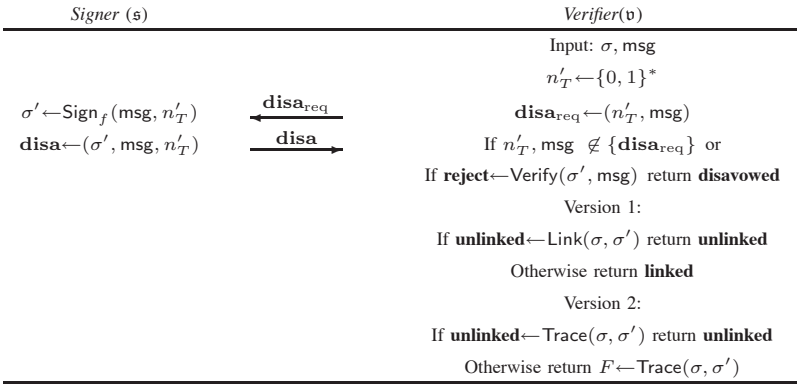

Fig. 8. The Disavow Protocol

this is called the *external Diffie-Hellman* assumption ([10]). Note that we are working in asymmetric pairing setting (i.e., $\mathbb{G}_1 \neq \mathbb{G}_2$), while in a symmetric pairing the DDH assumption does not hold any more.

The **Gap-DL assumption** in a group $\mathbb{G}_1 = \langle P_1 \rangle$ is as follows. Let $X = x \cdot P_1$ for some unknown $x \in \mathbb{Z}_q$ and $\hat{t}$ the related pairing from $\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$. Suppose there is an oracle (a helping algorithm) that, on input $Y \in \mathbb{G}$, will return $x \cdot Y$. Given the value X, there is no polynomial algorithm that computes x after querying the oracle a polynomial number of times.

Intuitively, the blind bilinear LRSW assumption implies that any polynomial attacker, who can corrupt and control a polynomial number of credentials held by the corresponding vehicles, cannot forge a new credential (f, A, B, C) by itself without the issuer's help. This essentially means that the Join protocol is secure, *i.e.*, only a registered vehicle is able to get its valid credential cre with respect to one secret signing key f from the issuer. The Sign algorithm gives a non-interactive zero knowledge proof ([21]) showing that the signer knows the secret f corresponding to a valid (and unforgeable) but shuffled credential (R, S, T). In other words, this implies that the Sign algorithm is secure, that is, without holding a valid credential nobody can produce a valid anonymous signature for any message. Together they provide both entity authentication and data integrity, as only legitimate vehicle can generate a valid (anonymous) signature and such a signed message cannot be forged or modified without being detected.

Moreover, if a vehicle signs a message with the same content (msgt) multiple times, all those signatures can be linked since they include the same pair (J, K), (as $J = H_3(\mathsf{msgt})$, $K = f \cdot J$). (This is also the rationale behind the Link algorithm.) Hence our TAA scheme provides distinguishability of origin, which enables the ThresholdCheck algorithm to correctly determine the number of vehicles announcing a particular event. If an event is not true, these announcements will not result in any action, since we assume that the number of malicious vehicles is relatively few. Consequently threshold verification is supported in our TAA scheme. This also means that our scheme can provide a high level of message truthfulness and a certain level of system robustness against active adversaries. Hence our TAA scheme satisfies the requirement of reliability.

The essence of the Sign algorithm is to shuffle the original credential (A, B, C) to a temporary one (R, S, T) in such a way that it is infeasible to link a shuffled credential to the original one, or to link two shuffled credentials. This is due to the following unique feature in the asymmetric pairing setting: there is no isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$ (more details in [10]). The temporary credentials do not contain any useful information that can be used to link two signatures or indicate their owners' identities, so this justifies why anonymity is provided in the TAA scheme. As long as a vehicle behaves correctly and does not sign a message with the same content more than once, its signatures will be unlinkable by all parties, since under the DDH assumption, it is infeasible to decide whether two pairs (J, K) and (J', K') are generated using the same secret f or not. Therefore, unlike the existing schemes based on group signatures [8], [15], our scheme protects the privacy of a well-behaved vehicle against active adversaries as well as "honest-but-curious" authorities. Hence our TAA scheme satisfies the privacy requirement.

Finally, we consider the requirement of auditability. Revocation is supported in our TAA scheme by publishing the rogue list, which contains the secret f of all compromised vehicles. To prevent previously legal but currently illegal users from generating valid message announcements, revocation can also be performed by updating issuers' keys and legitimate signers' credentials, as described in Section IV-I. Non-repudiation is achieved by virtue of the Disavow protocol since a vehicle's secret signing key is known only to itself. Traceability is provided by version 2 of the Sign algorithm. Using the Trace algorithm (for version 2), a misbehaving vehicle signing the same message multiple times can be traced and its identity (the value F) revealed. Hence our TAA scheme allows a reasonable amount of auditability while protecting honest users' privacy.

VII. Performance Analysis

Table I summarizes the performance of our TAA scheme. We consider only the three most expensive operations: scalar multiplications in $\mathbb{G}_1$, exponentiation in $\mathbb{G}_T$, and pairing evaluations, since the overheads of hash evaluations and arithmetic operations in $\mathbb{Z}_q$ are very small compared to these expensive operations. As observed in [10], in usual implementations one exponentiation in $\mathbb{G}_T$ costs about 4 scalar multiplications in $\mathbb{G}_1$. This means that we can transform all exponentiations in

TABLE I
PERFORMANCE OF THE TAA SCHEME

	Party	Computational cost†
Join	Issuer	$6 \cdot \mathbb{G}_1 + \mathrm{ver}_{\mathsf{PK}}$
	Signer	$3 \cdot \mathbb{G}_1 + 4 \cdot P + \mathrm{sig}_{\mathsf{SK}}$
Sign	Signer	v1: $6 \cdot \mathbb{G}_1 + 1 \cdot P$ v2: $9 \cdot \mathbb{G}_1 + 1 \cdot P$
Verify	Verifier	v1: $5 \cdot \mathbb{G}_1 + 5 \cdot P$ v2: $8 \cdot \mathbb{G}_1 + 5 \cdot P$
Link	Verifier	verify two signatures
Trace	Verifier	v2: verify two signatures $+1 \cdot \mathbb{G}_1$
Disavow	Signer	sign a message
	Verifier	link or trace two signatures

† Here, $n \cdot \mathbb{G}_1$ denotes n scalar multiplications, $m \cdot P$ denotes m pairing operations, and all required exponentiations in $\mathbb{G}_T$ have been transformed into scalar multiplications in $\mathbb{G}_1$, while v1 and v2 refer to the two versions of the proposed TAA scheme.

$\mathbb{G}_T$ required in our TAA scheme into scalar multiplications in $\mathbb{G}_1$ to get a faster implementation. For example, to calculate $\hat{t}(S,X)^x$ in the Sign algorithm we can first compute $x \cdot S$ and then get the value of $\hat{t}(S,X)^x$ by computing $\hat{t}(x \cdot S, X)$. This trick also applies to the Verify algorithm. The numbers given in Table I reflect the results after this transformation. In addition, we will discuss the cost of checking the rogue list in Table III, rather than list it in this table. We also remark that the ThresholdCheck algorithm is not considered here as its main job is to validate and link a number of signatures and it does not require any expensive operation.

In Table I, $n \cdot \mathbb{G}_1$ denotes n scalar multiplications in $\mathbb{G}_1$, and $m \cdot P$ means m pairing operations. Note that in the Join protocol, the signer also needs to sign her commitment F using a standard signature, which is denoted by $\mathrm{sig}_{\mathsf{SK}}$, while the issuer will verify this signature, denoted by ver_{PK}. Here, v1 and v2 refer to the two versions of the proposed TAA scheme. In the Trace algorithm, as both $L_0^\dagger$ and $L_1^\dagger$ have been calculated during the process of signature verification, we assume they are available for the Trace algorithm. So, for the Link and Trace algorithms there are almost no extra costs except the necessary overheads to verify the two signatures. The Disavow protocol is also very simple.

A. Performance comparison with other schemes

In this section, we compare our TAA scheme with the two most relevant schemes, Hybrid [8] and GSIS [15]. Specifically, we conduct our comparison in three categories: functionalities, message and signature sizes, and computation time.

Functionalities. In Table II, we compare the schemes by considering the three main functionalities of reliability, privacy, and auditability. These are further divided into eight security requirements as discussed in Section I. Entity authentication and data integrity are satisfied in all proposals, as secure digital signatures are used to sign messages. However, only our TAA scheme provides flexible threshold verification, as neither GSIS nor Hybrid provides distinguishability of origin. Anonymity and revocation are also achieved by all schemes, as some kind of group signature techniques are used. However, Hybrid can only partially provide unlinkability (marked by a 'half tick' ✓ in Table II), as discussed in Section II. Apart from our TAA scheme version 1, all solutions satisfy the traceability property. Finally, neither Hybrid nor GSIS can achieve non-repudiation, as the issuer also holds the group signing key of a vehicle. In our TAA scheme, as in the DAA scheme proposed in [10], the issuer does not know the most important piece of the signing key: the secret f. Consequently, our scheme (both versions) satisfies the non-repudiation requirement.

In the following, we shall mainly compare our TAA scheme to GSIS as their structures are much closer than that of Hybrid.

Message and signature sizes. In GSIS [15], the size of a group signature σ is $|\sigma| = 3|\mathbb{G}_1| + 6|q|$. In our scheme TAA v1, $|\sigma| = 5|\mathbb{G}_1| + 3|q|$, and in v2, $|\sigma| = 7|\mathbb{G}_1| + 3|q|$. To achieve 80-bit security level, approximately the same level as a standard 1024-bit RSA signature, we can use the elliptic curves suggested in [20] by picking a 160-bit prime q and a group $\mathbb{G}_1$ where each element is 161 bits. So, the signatures sizes in GSIS and two versions of our TAA scheme are 1443 bits (or 181 bytes), 1285 bits (or 160 bytes) and 1607 bits (or 201 bytes), respectively. Hence we conclude that our TAA scheme has similar signature size as GSIS. The signature size in Hybrid is also about 200 bytes. According to the simulation results given in [8], [15], the signature sizes of both versions of our TAA scheme are suitable for VANETs.

Computation time. According to the implementation results given in [20], to achieve 80-bit security level we can set $|q| = 160$ and $|\mathbb{G}_1| = 161$, and select appropriate curves, so that one multiplication in $\mathbb{G}_1$ and one pairing evaluation can be done within 0.6 ms and 4.5 ms, respectively. Those results were obtained from running an Intel Pentinum IV 3.0 GHZ machine, which has similar performance as the CVIS vehicle machine developed for the future communications in VANETs ([1]). Then, according to the performance analysis given in Table I and similar analysis on GSIS ([15]), we get Table III showing the comparison of computation times (n denotes the size of the revocation list).

From Table III, we can see that signing a message in both versions of our TAA scheme takes almost the same time as in GSIS, while the time for verifying a signature in our scheme is approximately doubled. The main reason for this is that a few pairing evaluations in GSIS can be pre-computed, but this is not possible in our TAA scheme. On the other hand, revocation check in our scheme is significantly faster than in GSIS, since for each entry in the rogue list (or revocation list), our scheme needs to perform one only scalar multiplication in $\mathbb{G}_1$ but GSIS has to evaluate three pairings. In practice, signature verification always involves revocation check, so the overhead for validating a signature in our TAA scheme is always less than in GSIS if the revocation list is not empty, *i.e.*, $n \geq 1$. By assuming two signatures have been verified (the usual case in practice), our Link algorithm has no other overhead except checking whether two strings are the same. To run the Disavow protocol, one new signature on the same message should be generated and verified.

From the above comparison results, we can conclude that our TAA scheme maintains similar performance to the existing

TABLE II
COMPARISON OF FUNCTIONALITIES

	Reliability:			Privacy:		Audiability:		
	Auth.	Integ.	Threshold.	Anonym.	Unlink.	Trace.	Revoc.	Non-repud.
Hybrid [8]	✓	✓	x	✓	✓	✓	✓	x
GSIS [15]	✓	✓	x	✓	✓	✓	✓	x
Our TAA v1	✓	✓	✓	✓	✓	x	✓	✓
Our TAA v2	✓	✓	✓	✓	✓	✓	✓	✓

Auth.: entity authentication; Integ.: data integrity; Threshold.: threshold verification; Anonym.: anonymity; Unlink.: unlinkability; Trace.: traceability; Revoc.: revocation; Non-repud.: non-repudiation.

TABLE III
COMPARISON OF COMPUTATION TIME (IN MS)

	Sign	Verify	Revocation check‡	Link	Trace	Disavowal
GSIS [15]	7.5	13.8	$13.5\times n$	–	1.2	–
Our TAA v1	8.1	25.5	$0.6\times n$	0	–	33.1
Our TAA v2	9.9	27.3	$0.6\times n$	0	0.6	37.8

‡ n used in the 4th column denotes the length of the revocation list.

solutions of [8], [15], while providing the additional features of flexible threshold verification, a higher level of user privacy, and non-repudiation.

VIII. CONCLUSION

We have presented a novel Threshold Anonymous Announcement scheme for VANET communication based on direct anonymous attestation and one-time anonymous authentication. Our scheme resolves the issues of non-repudiation and distinguishability of origin which were previously unresolved in the schemes of [15] and [8], thereby enabling a reliable and auditable TAA service while preserving user privacy against both authorised parties and adversaries.

One important criteria in assessing TAA schemes is performance. A good balance between hardware and software should be considered. In trusted computing technology, we consider a cheap TPM and a powerful platform. In VANETs, where computation and storage may be substantial, and where communication characteristics may differ according to applications, the type of platform used would have some influence on the type of schemes that can be implemented. Further work on evaluation criteria to better assess present schemes would also be useful. Other aspects of TAA schemes that has yet to be fully investigated include the possiblity of incorporating roadside infrastructure to improve the efficiency of tracing and revocation. This will also require a careful study of security assumptions made on this infrastructure. Another challenge for future work is to explore the formal definitions of the rich and diverse security properties desired in VANETs and give rigorous proofs to the security of the proposed TAA scheme.

REFERENCES

[1] The CVIS Project. http://www.cvisproject.org/ Last accessed December 10, 2009.

[2] National Highway Traffic Safety Administration, United States Department of Transportation. *Vehicle Safety Communications Project - Final Report*, April 2006. Available at http://www.nhtsa.dot.gov/portal/nhtsa_static_file_downloader.jsp?file=/staticfiles/DOT/NHTSA/NRD/Multimedia/PDFs/Crash%20Avoidance/2006/Vehicle%20Safety%20Communications%20Project%20-%20Final%20Report.pdf Last accessed December 10, 2009.

[3] D. Boneh, X. Boyen and H. Shacham. *Short group signatures*. In Proc. Crypto 2004, LNCS 3152, pp. 41–55, Springer-Verlag, 2004.

[4] D. Boneh and H. Shacham. *Group signatures with verifier-local revocation*. Proc. 11th ACM conference on computer and communications security 2004, pp. 168–177, ACM Press, 2004.

[5] E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In *Proc. 11th ACM Conference on Computer and Communications Security*, pages 132–145. ACM Press, 2004.

[6] E. Brickell, L. Chen and J. Li. *Simplified security notions of direct anonymous attestation and a concrete scheme from pairings*. International journal of information security, volume 8(5), pp. 315-330, October 2009.

[7] E. Brickell and J. Li. *Enhanced Privacy ID: a direct anonymous attestation scheme with enhanced revocation capabilities*. In Proc. 2007 ACM workshop on privacy in electronic society, pp. 21–30, 2007.

[8] G. Calandriello, P. Papadimitratos, J.-P. Hubaux and A. Lioy. *Efficient and robust pseudonymous authentication in VANET*. In VANET'07: Proc. fourth ACM international workshop on vehicular ad hoc networks, pp. 19–28, 2007.

[9] L. Chen and S. L. Ng. *Comments on "Proving reliability of anonymous information in VANETs" by Kounga et al*. IEEE Trans. Veh. Technol., volume 59(3), pp. 1503–1505, March 2010.

[10] L. Chen and P. Morrissey and N. P. Smart. *DAA: Fixing the pairing based protocols*. Cryptology ePrint Archive: Report 2009/198, available at http://eprint.iacr.org/2009/198 Last accessed December 10, 2009.

[11] 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, Issue. 4, pp 1876-1886.

[12] J. R. Douceur. *The Sybil attack*. In Proc. 1st international peer to peer systems workshop (IPTPS 2002), March 2002.

[13] P. Golle, D. Greene and J. Staddon. *Detecting and correcting malicious data in VANETs*. In Proc. 1st ACM international workshop on vehicular ad hoc networks, pp 29–37, 2004.

[14] G. Kounga, T. Walter and S. Lachmund. *Proving reliability of anonymous information in VANETs*. IEEE Trans. Veh. Technol., Vol. 58, Issue 6, July 2009, pp. 2977-2989.

[15] X. Lin, X. Sun, P.-H. Ho and X. Shen. *GSIS: Secure vehicular communications with privacy preserving*. IEEE Trans. Veh. Technol., vol. 56, no. 6, pp. 3442–3456, 2007.

[16] B. Ostermaier, F. Dötzer and M. Strassberger. *Enhancing the security of local danger warnings in VANETs - a simulative analysis of voting*

schemes. In Proc. 2nd international conference on availability, reliability and security, pp. 422–431, 2007.
[17] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung and J.-P. Hubaux. *Secure vehicular communication systems: design and architecture*. IEEE Commun. Mag., November 2008, pp. 100–109.
[18] P. Papadimitratos and J. -P. Hubaux. *Report on the "Secure Vehicular Communications: Results and Challenges Ahead" Workshop*. ACM Mobile Computing and Communications Review (MC2R), April 2008.
[19] M. Raya, A. Aziz and J.-P. Hubaux. *Efficient secure aggregation in VANETs*. In Proc. 3rd international workshop on vehicular ad hoc networks - VANET'06, pp. 67–75, 2006.
[20] M. Scott. *Efficient implementation of cryptographic pairings*, 2007. http://ecrypt-ss07.rhul.ac.uk/Slides/Thursday/mscott-samos07.pdf Last accessed December 10, 2009.
[21] D.R. Stinson. *Cryptography - Theory and practice (3rd ed)*. Chapman & Hall/CRC Press, 2006.
[22] I. Teranishi, J. Furukawa, and K. Sako. *k*-times anonymous authentication (extended abstract). In Proceedings of ASIACRYPT 2004, LNCS 3329, pp. 308–322. Springer, 2004.
[23] Trusted Computing Group. http://www.trustedcomputinggroup.org Last accessed December 10, 2009.

Liqun Chen received her BSc (1982), MSc (1985) and PhD (1988) in Information Science and Engineering from Southeast University, P.R.China. Prior to joining HP Labs Bristol UK as a senior researcher, she worked at Royal Holloway, University of London, Oxford University and Southeast University. She is an honorary senior lecturer in the School of Computer Science, University of Birmingham, a visiting professor in the College of Computer Science, Beijing University of Technology, and an associate editor of IEEE Transactions on Vehicular Technology. Her research interests include cryptographic protocols, trusted computing, computer and network security, mobile telecommunication system security and formal security proof. She has published over 60 research papers, held over 30 granted patents, and edited four international security standards. Her home page is http://www.hpl.hp.com/personal/Liqun_Chen/.

Siaw-Lynn Ng was awarded a B.Sc. (Hons.) degree in Mathematics from the University of Adelaide in 1995, and a Ph.D. in Mathematics from Royal Holloway, University of London in 1999. She joined the Information Security Group at Royal Holloway as a post-doctoral research assistant and is currently a lecturer in the Group. Her main research interests are in combinatorics and information security. Her home page is http://www.isg.rhul.ac.uk/~slng/.

Guilin Wang is a senior lecturer in the School of Computer Science and Software Engineering, University of Wollongong, Australia. Before this, he was a lecturer in University of Birmingham, UK, a research scientist in the Institute for Infocomm Research (I^2R), Singapore, and an assistant professor in the Institute of Software, Chinese Academy of Sciences, where he received his PhD degree in computer science in March 2001. Dr. Wang has served as a program co-chair for two international security conferences (ICICS'08 and ISA'09), a program committee member for more than 30 international information security related conferences or workshops, and a reviewer for more than 20 international journals. Up to now, he has about 60 technical publications in the areas of applied cryptography, information security, and electronic commerce. In particular, Dr. Wang is interested in the analysis and design of digital signatures and security protocols.