Vehicular Communications 25 (2020) 100279

Contents lists available at ScienceDirect

Vehicular Communications

www.elsevier.com/locate/vehcom

Accountable credential management system for vehicular communication

Salabat Khan [a,*], Liehuang Zhu [a], Xiaoyan Yu [b], Zijian Zhang [a,c,*], Mussadiq Abdul Rahim [a], Maqbool Khan [d], Xiaojiang Du [e], Mohsen Guizani [f]

[a] School of Computer Science and Technology, Beijing Institute of Technology, Beijing, PR China
[b] Department of Computer Science and Technology, Capital Normal University, Beijing, PR China
[c] Department of Computer Science, The University of Auckland, Auckland, New Zealand
[d] Department of Computer Science, Nanjing University, Nanjing, PR China
[e] Department of Computer and Information Sciences, Temple University, Philadelphia, PA, USA
[f] Department of Computer Science and Engineering, Qatar University, Qatar

ARTICLE INFO

Article history:
Received 8 December 2019
Received in revised form 3 May 2020
Accepted 24 June 2020
Available online 1 July 2020

Keywords:
Transparency log
Intelligent Transportation Systems (ITS)
Social Internet of Vehicles (SIoV)
Vehicular Public-key Infrastructure (V-PKI)

ABSTRACT

Social Internet of Vehicles (SIoV) is becoming a reality where private and secure communication is a prerequisite. Various standardization organizations and studies have reached a consensus to use Vehicular Public-key Infrastructure (V-PKI) in order to secure SIoV systems. However, significant security- and trust-related problems remain unsolved. This study presents an Accountable Credential Management System (ACMS) for vehicular communication to solve these problems. ACMS builds on transparency log (Distributed ledger technology) schemes for web PKI but addresses the challenges specific to vehicular communication. ACMS transparently handles certificate-related use cases, namely, certificate provision, registration, validation, and revocation. It also enhances the security of vehicular communication through constant monitoring; hence, assuring that no pseudonym certificate is accepted by vehicles before being logged and witnessed. With an efficient data structure known as the Accumulation Tree (AT), we extend the conventional transparency log to provide a cost-effective and trustworthy authentication process without relying on certificate revocation lists. Time cost and performance analyses show that the proposed approach is feasible and scalable than existing V-PKI schemes. Moreover, using Tamarin Prover, it is verified that our proposed technique provides defense against an active adversary.

1. Introduction

Smart vehicles can communicate with each other via the Internet of Things, giving rise to a new area named as Social Internet of Vehicles (SIoV) [1,2]. Intelligent Transportation System (ITS) is an attractive scenario of SIoV where inter-vehicles (V2V) communication has reduced vehicle crashes by 80% through the broadcast of Basic Safety Messages (BSMs) [1–4]. Unfortunately, these intelligent systems potentially expose vehicles to numerous cyber-attacks [1,5]. The security and protection of BSMs are of prime priority as they directly endanger public safety applications (e.g., Co-operative Collision Warning, Slow/Stopped Vehicle Alert (SVA), and Electronic Emergency Brake Light (EEBL)). Due to the importance of security of such applications, various standardization organizations and studies have recommended the application of Vehicular Public-key Infrastructure (V-PKI) in both U.S. and Europe [6–8]. V-PKI is a broad and complex framework in which Certification Authorities (CAs) (e.g., Enrollment Authority (EA) and Pseudonym Certification Authority (PCA)) are the primary trust servers, and the security of V-PKI relies on the trustworthiness of them.

SECMACE [9], SCMS [10], and VPKIaaS [11] are leading V-PKI proposals that extended their adversarial model from fully trustworthy to honest-but-curious V-PKI trust servers because of the recent high profile CAs failures [12] in web PKI. These CAs compromises in PKI [12] revealed that high integrity and security could not be achieved without monitoring CAs operations and reducing trust placed in them. In order to tackle such issues, transparency emerged as an elegant mechanism to achieve accountability in which information about authoritative decisions made by authorities (e.g., CAs) is made public, thus enabling any client to validate for themselves whether or not the authoritative decisions comply with what they observe to be the rules [13].

* Corresponding authors.
E-mail addresses: salabatwazir@gmail.com (S. Khan), zhangzijian@bit.edu.cn (Z. Zhang).

https://doi.org/10.1016/j.vehcom.2020.100279
2214-2096/

Google's Certificate Transparency (CT) [14,15] is one such proposal that has introduced transparency in web PKI (SSL/TLS) certificates. It made misbehavior detection easier via appending all issued certificates to a transparency log. Following CT, many proposals [16–19] have been presented that obtained stricter security goals such as resilience against compromised CAs. Because of the effectiveness and high security of these proposals, there has been a strong demand for transparency in various security-critical systems.

One specific area that has been relatively unexplored is V-PKI. Bringing transparency to V-PKI is the need of the day as there are a growing number of incidents in which users of various websites (e.g., Google) and social networking applications (e.g., Facebook) are successfully attacked by taking down CAs. A survey showed that attempts had been made to intercept about 0.2% of Facebook users' communications with malicious certificates [20], and 3 million forged certificates were recognized for the prime 10 thousand Alexa sites [21]. Introducing social interactions to vehicular communication increases the degree of security risks (e.g., attacks on vehicles through fake certificates), which is overlooked by existing research work [22,23]. Existing V-PKI schemes do not take into account such extreme cases (e.g., malicious certificates issuance by a dishonest CA), which can cause chaotic fatal traffic accidents [22–25].

In this work, we design a new key validation infrastructure for vehicular communication scenarios to address these challenges. Our proposal is called Accountable Credential Management System (ACMS) for vehicular communication, integrating an architecture for key (e.g., certificate) revocation of all parties (e.g., vehicles) with a transparent infrastructure for accountability of framework parties through constant monitoring as well as checks-and-balances.

ACMS addresses common key management operations and gracefully handles adversarial incidents such as vehicle key loss or compromise. We design ACMS to make progress towards a transparent vehicular public-key management architecture with certificate revocation that reduces the trust placed in any single party. To achieve these goals, we leverage globally visible transparency log while keeping in mind the privacy of vehicles that enables public auditing of pseudonym certificate information and makes authorities accountable for their conduct. In summary, this study contributions are below:

- We present a secure, transparent, and accountable V-PKI scheme based on a transparency log. ACMS makes it possible to identify compromised authorities (Enrollment Authority and PCA) as well as to detect maliciously acquired or mistakenly issued fake certificates.
- ACMS guarantees strong security while still maintaining the level of privacy offered by leading V-PKI schemes.
- A formal machine-checked security analysis of ACMS is conducted, justifying its resilience against PCA compromise using Tamarin Prover [26]. Lastly, a performance analysis is performed, which shows that the proposed scheme is more efficient and scalable than exiting state-of-the-art V-PKI proposals.

1.1. General notations

For the comfort of the readers, Table 1 displays the key notations and symbols that are used in this article.

Table 1
Key notations.

Symbol	Description
g	Generator of an elliptic curve group
$Cert_P^A$	A certificate signed by authority A for party P
$Cert_i^{PCA}$	Pseudonym certificate i issued by PCA
U_i, u_i	Public and secret keys relating to $Cert_i^{PCA}$
A, a	Public and secret caterpillar keys
$\hat{A}, \hat{a}$	Unified public and secret cocoon keys
f	Pseudorandom function
π	Proof generated by transparency log maintainer
pk_P, sk_P	Public and secret keys of party p
$sign(s, sk_P)$	Signing statement s with secret-key sk of party P
$verify(s, pk_P)$	Verification of signature on statement s, with public-key of party P
$Enc(s, k)$	Encryption of a statement s with key k
$Dec(s, k)$	Decryption of a statement s with key k

2. Related work

In this section, we first review leading V-PKI proposals and then overview SCMS, which is closest to our proposal. Schemes such as SeVeCom [27,28], CAMP VSC3 [29], PRESERVE [30], C2C-CC [31], the IEEE 1609.2V2 standard [32], ETSI [33], and IEEE standard V-PKI [34] relied on the pseudonym certificate for privacy-preserving authentication in the V-PKI. While some other registry-based schemes [35,36] were proposed which tried to eliminate the performance issue of V-PKI. In these proposals, each vehicle and Roadside Units (RSUs) store the public-keys of all vehicles as well as RSUs. Nevertheless, registry-based schemes did not take into account the privacy of vehicles, and they assumed that V-PKI servers are fully trusted. Several proposals such as PRESERVE [30] and C2C-CC [31] allowed direct communication between the CA and PCA for pseudonym certificate issuance. This raised concerns that CA could link the real identity of the vehicle with her corresponding pseudonym certificate [9]. In a solution, ticket-based pseudonym certificate issuance proposals [37,38] were put forward, in which CA issues authenticated, yet anonymous, tickets for vehicles to acquire a pseudonym certificate from PCA. However, a common problem with ticket-based schemes is that PCA can still associate the batch of pseudonym certificates issued for a vehicle acquired as a result of a single pseudonym[1] acquisition request.

To overcome this issue, [29,39] proposed the idea of a proxy-based solution in which the registration authority shuffles and aggregates pseudonym acquisition requests before sending them to the PCA for pseudonym certificate issuance such that PCA could not link the set of pseudonyms to a vehicle. [9,40] used the idea of using pseudonyms with a non-overlapping lifetime to prevent a compromised vehicle from obtaining multiple valid sets of pseudonym certificates. [9,40,41] used the idea that CA keeps records of tokens generated for each vehicle and does not issue a token for the subject during a specified interval of time. The recent works in [10,11,29] proffered stout privacy against compromised insider, but they have not addressed the challenge of fake certificate issuance by CAs (e.g., EA and PCA). A compromised signing authority such as PCA may arbitrarily authorize any entity to join the vehicular communication by endorsing fake but valid pseudonym certificates.

Aside from the above mentioned problem, transparency, that has become a demanding requirement for security-critical systems, is still missing in V-PKIs; and CT [14,15] introduced it in web PKI via leveraging transparency logs. Subsequent proposals such as AKI [16] and ARPKI [17] tendered stout defense against compromised insiders (e.g., CAs) as well as solved the revocation problem left open by CT. These log-based proposals operate with two new entities: 1) transparency log server: maintaining a publicly visible history of CA-issued certificates, and 2) monitor: a third party watching transparency log for malicious entries. However, these schemes are incompatible with enormously grow-

[1] In this article, pseudonym and pseudonym certificate are used interchangeably.

Table 2
Strengths and weaknesses of the state-of-the-art V-PKI. Underlined entries show major limitation of the corresponding system.

Metrics	SeveCom [27,28]	CAMP VSC3 [29]	PRESERVE [30]	C2C-CC [31]	IEEE V-VPK [34]	VeSPA [37]	CoPRA [38]	SECMACE [9]	SCMS [10]	Reg.schemes [35,36]	Proposed Scheme
Privacy preservation	✓	✓	✓	✓	✓	✓	✓	✓	✓	×	✓
Sybil-attack prevention	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Transparent credential management	×	×	×	×	×	×	×	×	×	×	✓
Built-in misbehaviour detection support	×	×	×	×	×	×	×	×	×	×	✓
Resilience against compromised CA (EA and PCA)	×	×	×	×	×	×	×	×	×	×	✓
Reduced trust in a single party (e.g., EA and PCA)	×	×	×	×	×	×	×	×	×	×	✓
Machine-checked formal security proof	×	×	×	×	×	×	×	×	×	✓	✓

ing numbers of pseudonym certificates, which would overwhelm their transparency log. Similarly, AKI and ARPKI restricted domain servers to have a single certificate, and CT could not shield clients against compromised CAs. However, in V-PKI, every vehicle has multiple pseudonym certificates to hide their real identity, and security against compromised insiders is also the top priority. Similarly, these schemes relied on a Merkle hash tree or binary hash tree, which has $log_2(m)$ proof size and computation cost where m is the number of entries in the tree. Table 2 shows the research gap addressed by the proposed scheme, which is left despite the previous works.

2.1. Secure credential management system for V2X communication

Here we review Secure Credential Management System (SCMS) [10,29] in detail for two reasons: (1) SCMS's design inspires ACMS and applies some of its privacy design blocks; (2) ACMS addresses several security loopholes and shortcomings that we have identified in the SCMS. SCMS is a leading V-PKI proposal whose early version has already been deployed, implemented, and tested by Safety Pilot Model Deployment [42] and is going to be applied for USDOT's connected vehicle project [43]. It supports the secure management of certificates and preserves the privacy of vehicles. It operates with the following main agents:

1. SCMS Manager: It defines rules and policies for reviewing misconduct and revocation requests.
2. Device: Any device (e.g., a vehicle, an after-market safety device (ASD)) that can send and receive messages.
3. Device Configuration Manager (DCM): Approves that a device is eligible to obtain an enrolment certificate, and helps in bootstrapping and configuration settings.
4. Electors: Entities responsible for endorsing or revoking a root CA.
5. Root Certification Authority (RCA): It issues certificates for intermediate CA.
6. Intermediate CA (ICA): It defends and shields RCA against cyber-attacks.
7. Enrollment CA (ECA): Signs enrollment credentials for devices.
8. Linkage Authority (LA): Generates hash values that act as pre-linkage values, and it is embedded in certificates.
9. Misbehaving Authority (MA): Processes any misconduct, reports it to identify a misbehaving device besides revoking them by adding to CRL.
10. Pseudonym CA (PCA): Issues pseudonym certificates to vehicles.
11. Registration Authority (RA): Combines certificate acquisition requests from devices, shuffles the requests before forwarding them to PCA.

BCAM [44] introduced activation codes mechanism to mitigate bidirectional communication for pseudonym provision and distribute Soft Revocation List (SRL), CRL, and device specific values to revoke a vehicle in case of software and hardware corruption of vehicles. [45,46] identified privacy attacks on [44] and SCMS linkage values and proposed improvements to avoid these attacks.

Weaknesses of SCMS. First, SCMS relies on the honest-but-curious adversarial model in which trusted authorities (e.g., ECA and PCA) might authorize an illegitimate entity to join a vehicular communication network [11]. In other words, if an adversary compromises a trusted authority in SCMS, then the adversary can successfully join the network and can communicate with other participants as an authorized party in the network. For example, in the current design of SCMS, compromising only PCA is sufficient to generate fake pseudonym certificates and use them in conducting attacks against vehicles.

Second, SCMS lacks transparency and misbehavior detection mechanism that is the operation of signatory authorities (e.g., ECA and PCA) are not open to public scrutiny and monitoring to detect their compromises [11]. This enables signatory authorities to misbehave without leaving any proof of misconduct [17]. These advanced security goals are already achieved by some proposals [16–18] outside vehicular communication.

Besides security and trust-related problems of SCMS, it still relies on CRL for certificate revocation, which raises performance concerns. For example, CRL size increases with the increase in the number of revoked vehicle and searching for revoked certificates requires O (n) time where n represents the number of entries in a CRL. This induces a considerable delay to the handshake process in vehicular communication, which is unacceptable in safety-related applications.

The first loophole is addressed by enhancing the authentication mechanism in which clients accept only those certificates, which have proof from the Transparency Log Maintainer (TLM) (see Section 4.2), which prove that they are cross-checked and not yet revoked. The second issue is solved by exposing CAs (e.g., EA and PCA) operations to public scrutiny through logging and monitoring. The performance challenge is handled by introducing Accumulation Tree (AT), which has constant O (1) verification time and proof size independent of the number of revoked vehicles.

3. Preliminaries and adversary model

3.1. Merkle hash tree (MHT)

MHT[2] [47] is maintained in the form of binary tree composed of hashed nodes and leaves. The leaves are typically the hashes of

[2] In this article, Merkle tree and Merkle hash tree are used interchangeably.

a statement signed by authorities. It is typically maintained in either chronological or lexicographical sorted form depending on the types of proof, users are interested in. For example, chronological MHT can deliver efficient membership and consistency (showing that any two versions of MHT are consistent) proof while lexical MHT can generate membership as well as non-membership evidence.

Lemma 1. *(Collision-resistance of MHT). If H is collision-resistant hash function, then the MHT is collision-resistant.*

Proof. This follows from works by [47,48]. □

3.2. Bilinear pairing

Let G, and G_T be a multiplicatively-written cyclic group of order n, and g be their generator. A multiplicative bilinear function on (G, G_T) is a map $e : G \times G \rightarrow G_T$ that if it satisfies the following features

1. **Bilinearity:** ∀ P and Q ∈ G, ∀ l ∈ $\mathbb{Z}_p$ and m ∈ $\mathbb{Z}_p$ $e(P^l, Q^m) = e(P, Q)^{lm}$
2. **Non-generacy:** ∀ g ∈ G, $g \neq 0 \Rightarrow e(g, g) \neq 1$
3. **Computability:** The map (e) must be efficiently computable.

3.3. Cryptographic accumulators

Cryptographic accumulators were introduced by Benaloh and de Mare [31] as a decentralized alternative to digital signature and defined as a one-way function. Cryptographic accumulators were further made dynamic by permitting insertion and deletion of elements into the set of an element dynamically. Our scheme applies bilinear-pairing oriented cryptographic accumulators having constant membership communication and computation cost. Let BMSetup(1^λ) be a randomized algorithm that returns bilinear public parameters. We suppose that we have an object of bilinear-map and a set L = {$c_1, c_2, ..., c_n$}, such that $\forall c_i \in \mathbb{Z}_p^*$. Let s be the trapdoor information from $\mathbb{Z}_p^*$. Then the accumulation value of L is

$$Acc_s(L) = g^{(c_1+s)(c_2+s)...(c_n+s)} = g^{\prod_{c\in L}(c+s)} \quad (1)$$

Where s and the set $\{g^{s^i} \| 0 \leq i \leq q\}$ are trapdoor and public-key respectively and q is an upper-bound on $|L| = m$. As in [49], for any $c \in L$, then membership witness $W_{c,L} \in G$ of c is defined to be the value $W_{c,L} = g^{\prod_{c_j \in L: c_j \neq c}(c_j+s)}$ that can satisfy the membership validation check, which, applying e(.,.) and the well-known group element g^s can be generalized in practice [50] as

$$e(W_{c,L}, g^{W_{c,L}}.g^s) = e(Acc_s(L), g) \quad (2)$$

That is, any $c \in L$ has a unique membership evidence and let $f_X(s) \stackrel{\triangle}{=} \prod_{c\in L}(c+s)$, then, witness is $W_{c,L} \stackrel{\triangle}{=} g^{\frac{f_X(s)}{(c+s)}} = g^{qL,c(s)}$ (since (c+s) divides $f_L(s)$), for some polynomial $qL, c(s)$ of degree m-1, that is defined by set $L - c$. This kind of accumulator is secure under q-strong DH assumption [50].

Definition 1. (q-strong DH (Diffie Hellman) Assumption [50]) Let g be the generator of G (multiplicative cyclic of prime order q) and $s \in \mathbb{Z}_p^*$. Any probabilistic polynomial time (PPT) adversary $\mathcal{A}$ that is given set $\{g^{s^i} : 0 \leq i \leq q\}$ can find a pair $(c, g^{\frac{1}{c+s}}) \in \mathbb{Z}_p^* \times G$ with probability at most $O(\frac{1}{p})$.

3.4. Accumulation tree (AT)

Accumulation Tree (AT) is an authenticated data structure with $[1/\epsilon]$ levels, where ϵ is chosen upon during setup such that $0 < \epsilon < 1$ having m leaves [51,52]. Each internal node of the AT has a degree (m^ϵ) and the tree has a constant depth for a fixed value of ϵ. The most significant property of the AT is that, for a particular element, it holds a constant membership proof size and computational cost that is O(1).

3.5. Transparency log

Bitcoin [53] has achieved tremendous success, but eliminating all forms of central authority highly limits its ability to achieve widespread adoption. Hence, technological solutions such as Google Certificate Transparency (CT) [14,15] and following proposals [16–19] have emerged that instead seek to offer more openness, visibility and transparency into currently centralized authorities. Currently, many CAs support CT, which is currently deployed in web servers and major browsers. CT is an elegant mechanism to detect compromised CA and malicious certificates via leveraging a transparency log.

Transparency log keeps the history of all issued certificates by maintaining them in an append-only form. It is maintained in the form of an authenticated data structure (e.g., Merkle tree) that ensures transparency and accountability of authorities via providing a globally visible view of authority signed statements. Hence, it is a simple network service that keeps a record of authority signed statement. It is publicly auditable and cryptographically assured because of authenticated data structure (e.g., Merkle tree hashes) that prevent tampering and misconduct. Transparency log can present efficient proof of membership of a statement as well as consistency. For more details on CT and transparency log, we refer readers to [14,15].

3.6. Unified butterfly key expansion (UBK)

Butterfly Key expansion method was first proposed in [10] to efficiently acquire arbitrarily large batches of short-lived pseudonym certificates with a single request. It was further optimized in [54] by presenting UBK that eliminates the double execution of the procedure by the Regional Authority (RA). In UBK, a vehicle generates a caterpillar private/public-key pair ($a, A = a.g$), where g is the generator of the elliptic curve group. The public caterpillar key A is sent to RA together with a suitable pseudorandom function f. The RA employs A in the generation of β cocoon keys $\hat{A}_i = A + f(i).g$ for a handful amount of vehicles, shuffles them, and forwards the resulting bundle to the PCA.

3.7. Adversary model

Our proposed scheme can defend vehicles against the following kinds of attacks.

1. Attacks on clients' privacy from outside the ACMS and ACMS insiders.
2. Attacks on vehicles from EA and PCA by issuing fake enrollment certificate and pseudonym certificates.

We thwart the first item listed above by the pseudonym certificate and by the design of ACMS. The second item is addressed via logging and validating each enrollment and pseudonym certificate before being used in the authentication. We consider the standard Dolev-Yao adversary model [55] that is a more realistic and stronger adversary, where the adversary can eavesdrop, delete,

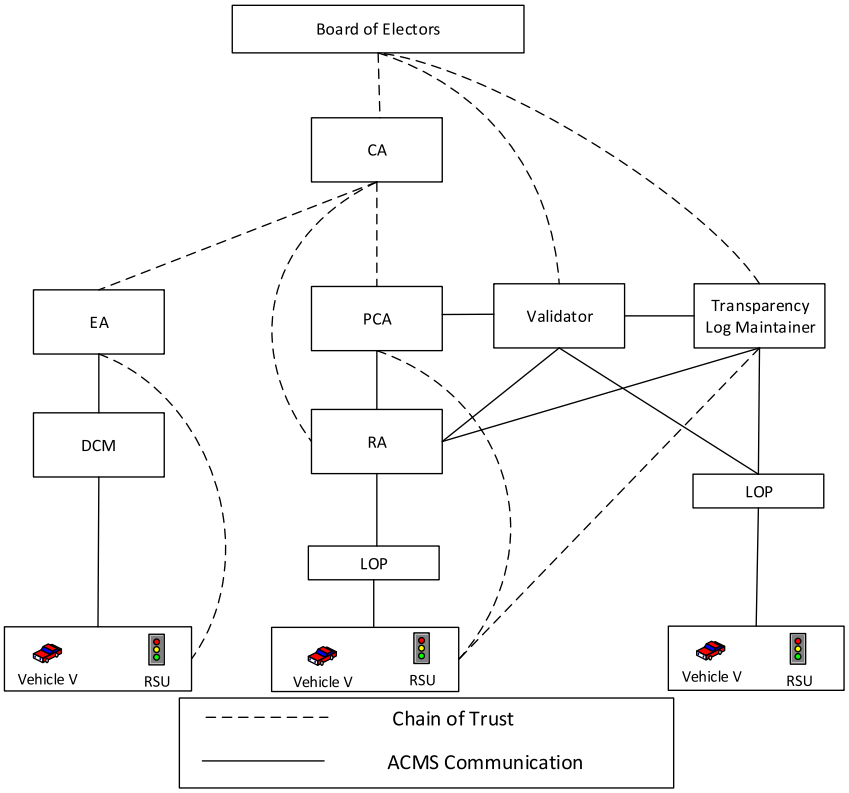

Fig. 1. A high-level overview of ACMS.

intercept, and inject false messages into the communication channel. We assume that all components are securely initialized, know the initial set of electors, CAs, PCA, TLM, and validator and their certificates.

4. Accountable credentials management system (ACMS) for vehicular communication

In this section, first, we describe the design and architecture of ACMS by briefly discussing its main components. Later on, we explain in detail the structure of our proposed transparency log and the working of the protocol.

4.1. Main components of ACMS

The following are the main agents involved in the secure operation of ACMS.

- Vehicle: A set of vehicles and each vehicle is fitted with an On-Board Unit (OBU). A vehicle wants to communicate with other entities securely, and any vehicle with a valid pseudonym certificate can communicate with other vehicles and with the rest of the infrastructure (e.g., RSUs) securely.
- Roadside Unit (RSU): A set of RSUs that deliver essential information to vehicles. Notably, they are deployed to improve network performance and to extend vehicle coverage in vehicular communications.
- Board of Electors: A group of entities responsible for endorsing or revoking trust anchors (e.g., a CA, TLM, and validator). In order to ensure transparency, international standardization bodies such as IETF[3] and ISO[4] can define requirements for an elector. Any organization that meets the requirement can act as an elector. Similarly, rules for endorsing or revoking should also be specified according to international standards. To distribute the trust, the board of electors can use multi-signature algorithms such as CoSi [56] to endorse or revoke trust anchor.
- Certification Authority (CA): It acts as a root of trust and the main trust anchor in the ACMS.
- Enrollment Authority (EA): Endorses long-term enrollment certificates for vehicles and RSUs and publishes them to the Transparency Log Maintainer.
- Pseudonym CA (PCA): Signs short-lived pseudonym certificates and registers them on Transparency Log Maintainer.
- Registration Authority (RA): Combines certificate acquisition requests from requester, shuffles the requests before forwarding the requests to PCA.
- Transparency Log Maintainer (TLM): It is maintained in the form of a special cryptographic secure and authenticated data structure (MHT and AT). Hence, TLM is a simple network service that keeps a record of all certificates. It is publicly verifiable and cryptographically assured because of the secure and authenticated data structure that prevents tampering and misconduct.
- Validator (VL): Validator watches for suspicious credentials in the TLM, such as illegitimate pseudonym certificates. A validator also verifies that all logged credentials are visible in the TLM. It does this by periodically fetching all fresh entries that have been inserted into a TLM. Validators can be operated by independent parties such as the traffic police department or law enforcement department.

ACMS also has additional components (e.g., device configuration module (DCM) and Location Obscurer Proxy (LOP)), but the functions of these components are similar to those of their counterparts as in [10,29] respectively, and hence we do not review them here. Fig. 1 shows the main components of ACMS and their logical

[3] Internet Engineering Task Force: https://www.ietf.org.
[4] https://www.iso.org/home.html.

roles within the system. Three pairs of RSU and vehicle V are used in Fig. 1 to demonstrate three different use cases of ACMS. The left-most use case is used to illustrate the connection required for the enrollment certificate provision, the middle one shows the connections needed for pseudonym certificate acquisition, and the last one shows secure communication with the rest of the parties for sharing valuable information with each other. There are two types of connections in the ACMS. The dashed lines show the credential chain of trust for certificate and signature verification against the applicable certificate while solid lines denote secure regular communications, including monitoring and reporting of misconduct.

4.2. Transparency log maintainer (TLM)

We first describe the structure and then briefly discuss the algorithms and construction of TLM. It keeps an append-only ledger of EA and PCA issued certificates in the form of authenticated data structures. Since vehicular communications need a data structure that is efficient both in terms of communication and computation cost, and in literature, verifiable authenticated data structures are extensively studied [14,15,47,57]. To our knowledge of the literature survey, no single data structure is sufficient to provide efficient proof in terms of both communication and computation costs. Hence, the database of our proposed transparency log is maintained in the form of a pair of Merkle tree and Accumulation tree.

Accumulation Tree (AT): The accumulation tree (AT) keeps PCA signed pseudonym certificates of all participants. Let L be the active set of pseudonym certificates (e.g., pseudonym certificates valid for the week); the L is implemented as an AT. Each leaf node in AT contains a certificate $Cert_i^{PCA}$ and the membership evidence $W_{Cert_i^{PCA}}$ of the pseudonym. The value of accumulation $Acc_s(L)$ is inserted in the last leaf-node of the Merkle hash tree.

Merkle Hash Tree (MHT): The MHT is maintained in chronological order [14,15], which is tamper-resistant, where the tree stores all enrollment certificates of all devices and the accumulation value of AT as the last node.

The MHT makes the TLM immutable and tamper-resistant while AT ascertains cost-effective (non)membership proof of pseudonym certificates.

Algorithms and Construction:

The ACMS clients and server implements the following algorithms and Fig. 2 gives the detailed construction. For clarity, calls made by TLM are prefixed with TLM and those made by devices (e.g., vehicles, RSUs and validators) are prefixed with D.

$TLM.Setup(1^\lambda) \rightarrow (PP)$: A randomized algorithm that runs BLSetup(1^λ) to return Public Parameters (PP) where λ is a security parameter.

$TLM.AddElem((Cert_i^{PCA}, Cert_j^{PCA}, ..., Cert_n^{PCA}), L_i, d_i, PP) \rightarrow (L_{i+1}, d_{i+1})$: Deterministic algorithm that inserts a new set of certificates $(Cert_i^{PCA}, Cert_j^{PCA}, ..., Cert_n^{PCA})$ to the version i list of TLM and returns a new version i+1, a new list L_{i+1} and signed digest d_{i+1} where L represents list of credentials.

$D.VerDigest(Li, d_i) \rightarrow (0/1)$: A deterministic algorithm that takes as an input a list L_i of pseudonyms and digest d_i and returns either true (1) or false(0).

$TLM.ProveElemMemb(Cert_k^{PCA}, L_i, PP) \rightarrow (Cert_k^{PCA}, \pi_i)$: Deterministic algorithm that proves membership for pseudonym certificate $Cert_k^{PCA}$. It searches for the pseudonym, if search is successful, it generates proof π_i and sends it back to the requester.

$D.VerElemMemb(Cert_k^{PCA}, \pi_i, d_i) \rightarrow (0/1)$: Verifies that the pseudonym $Cert_k^{PCA}$ is present in the TLM by verifying the proof generated by ProveCertMem(.) and returns either true or false.

$TLM.ProveAppendOnly(L_i, L_j, PP) \rightarrow (\pi_{i,j})$: Deterministic algorithm that proves that TLM version j extends version i. In other words, it proves that $L_i \subseteq L_j$.

$D.VerAppendOnly(pk_{TLM}, d_i, i, d_j, j, \pi_{i,j}) \rightarrow (0/1)$: Deterministic algorithm that assures that log maintainer remains append-only. Verifies that $\pi_{i,j}$ correctly validates that the MHT relating to digest d_j is an extension of the tree with digest d_i.

4.3. ACMS protocol description

In the remaining subsection, we describe the ACMS protocol in detail and a vehicle V is used to demonstrate the use cases.

4.3.1. Enrollment certificate provision

To start using ACMS, a vehicle say V must first acquire a valid enrollment certificate signed by a threshold number of EAs, which we call the multi-signature enrollment certificate (MEC). The MEC is encoded as a series of standard V-PKI enrollment certificates to ensure backward compatibility with current standards, where multiple certificates[5] endorse a common enrollment public-key. Hence, with $n \geq 1$ EA signatures, MEC is defined as below:

$$MEC_V = \{Cert_V^{EA_1}, Cert_V^{EA_2}, ..., Cert_V^{EA_n}\} \quad (3)$$

Where $Cert_V^{EA_i}$ is a certificate authenticating V's public-key and endorsed by EA_i. The board of electors can define the threshold number of EAs, and every enrollment certificate in the MEC_V can be obtained in the same way as in [10]. Moreover, each individual certificate in the MEC is authenticated by a distinct EA (i.e. a single EA cannot certify over one certificate within MEC). Fig. 3 depicts the flow of messages for MEC acquisition and registration, which are briefly described relative to the numbering in Fig. 3.

1. A vehicle V sends a request to the designated number of EAs to get an individual enrollment certificate (1a, 1b).
2. Each EA endorses a certificate after the successful validation of V (2a, 2b).
3. V forwards the MEC_V to validator for registration in a TLM.
4. The validator sends the MEC_V to a TLM for registration if a threshold number of EAs signs it, and each individual certificate in MEC_V is valid.
5. The TLM sends back a timestamp for it to the validator, promising that it would add the MEC_V to its database in the next update.
6. The validator countersigns the timestamp before forwarding the timestamp and MEC_V to the vehicle.

Multi-signature enrollment certificate and its registration ensure transparency and prevent the illegal enrollment of entity by less than the threshold number of EAs.

4.3.2. Pseudonym certificates acquisition and registration

The pseudonym provision method is the most tricky in the ACMS because it has to preserve vehicle privacy as well as

[5] An alternative method for MEC implementation can be designing a new certificate that can be signed by multiple parties or having a dedicated extension in the existing standard certificate (e.g., X.509 certificate extension) that allow authenticate a single public-key by multiple parties (e.g., CAs).

- Setup: It calls BMSetup that returns (q,g,G,G_T,e) as the bilinear-pairing parameters. For the set L of pseudonyms, the accumulation value is computed as $Acc_s(L) = g^{\prod_{Cert_i^{PCA} \in L}(Cert_i^{PCA}+s)}$. Setup constructs AT by inserting the active pseudonym set represented as L along with their membership proof, whereas the membership witness is calculated as $W_{Cert_i^{PCA},L} = g^{\prod_{Cert_i^{PCA} \in L: Cert_i^{PCA} \neq Cert_i^{PCA}}(Cert_i^{PCA}+s)}$. Finally, pseudonyms are also appended to MHT and root digest d_i is returned. The d_i, accumulation value, PP are published.
- AddElem: 1) Adds $Cert_k^{PCA}$ to AT. Computes the fresh accumulation value as $Acc_s(L') = Acc_s^{(Cert_k^{PCA}+s)}$. The membership witness and AT is also updated. Append $Cert_k^{PCA}$ to the right of MHT and the corresponding digest d_j is refreshed.
- VerDigest: Let the list L_i consists of $(Cert_1^{PCA}, Cert_2^{PCA}, Cert_3^{PCA}, ...Cert_n^{PCA})$ certificates, then the algorithm checks $h(h(h(Cert_1^{PCA}, Cert_2^{PCA}), h(Cert_3^{PCA}, Cert_4^{PCA}))....h(Cert_{n-1}^{PCA}, Cert_n^{PCA})) \stackrel{?}{=} d_i$.
- ProveElemMemb: On input PP, L, and $Cert_k^{PCA}$, it generates $\pi = W_{Cert_i^{PCA},L} = g^{\prod_{Cert_i^{PCA} \in L: Cert_i^{PCA} \neq Cert_k^{PCA}}(Cert_i^{PCA}+s)}$.
- VerElemMemb: Given the witness $\pi = g^{\prod_{Cert_k^{PCA} \in L: Cert_k^{PCA} \neq c}(Cert_k^{PCA}+s)}$, this algorithm computes $e(W_{Cert_k^{PCA},L}, g^{Cert_k^{PCA}}.g^s) \stackrel{?}{=} e(Acc_s(L), g)$, and accepts if it holds, otherwise rejects.
- ProveAppendOnly: It generates proof $\pi_{k,l}$ that consists of single hash value per level of the current MHT version. Proof $\pi_{k,l} = (h_1, h_2, ...)$ is the series of hash values needed to verify that the fresh tree having digest d_l is an extended version of a previous one having digest d_k.
- VerAppendOnly: Given the proof returned by ProveAppendOnly, the validator validates $h(...(h(d_k, h_1), h_2)...) \stackrel{?}{=} d_l$ by executing VerifyAppendOnly algorithm.

Fig. 2. Construction of TLM.

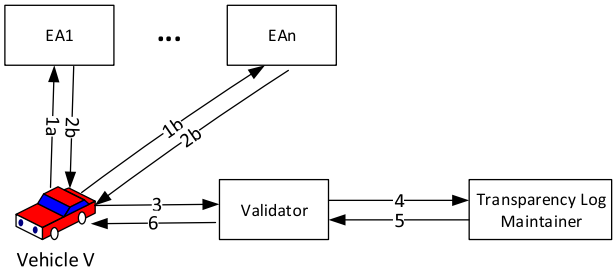

Fig. 3. Multi-signature Enrollment Certificate Acquisition and Registration.

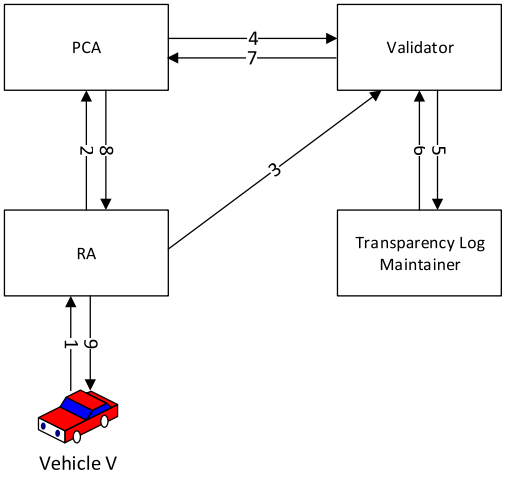

Fig. 4. Pseudonym Certificate Acquisition and Registration.

transparency. Fig. 4 depicts the messages flow for vehicle V's pseudonym certificate acquisition and registration, which are briefly described relative to the numbering in Fig. 4.

1. A vehicle V generates a caterpilar key-pair (a, A=a.g) and V then forwards pseudonym certificate signing request CSR = Enc({sign((A,f()),sk_{MEC}),MEC}, pk_{RA}) to RA, where f() is pseudo random function and sk_{MEC} is the vehicle private-key of multi-signature enrollment certificate MEC.
2. The RA generates α (20 for a week [10,31]) cocoon public-keys by expanding the seed value $\hat{A}_i = A + f(i).g$ for the vehicle if the MEC is not revoked as well as this is the only request by the vehicle. The RA receives several CSR from different vehicles. The RA performs shuffling over the individual expanded pseudonym certificate requests. The RA then forwards pseudonym certificate signing requests CSR_{RA2PCA}=sign({$\hat{A}_i$, $h(CSR_{RA2PCA})$},sk_{RA}) for individual pseudonyms to the PCA where $h(CSR_{RA2PCA})$ is hash value of RA pseudonym certificate signing request to the PCA.
3. The RA forwards a list of MEC of different vehicles requesting pseudonym certificates and the number of requests it has generated. The validator performs validation as in step 2 on the each MEC; the list enables validators to know the number of vehicles and number of pseudonym certificates requested by them without knowing which pseudonym certificate belongs to which MEC.
4. The PCA generates the public-key $U_i = \hat{A}_i + r_i.g$ for a random value r_i and creates a signed certificate $Cert_i^{PCA} = \{U_i, meta, \sigma\}$, where meta data include certificate related information (e.g., validity) and $\sigma = sign((U_i, meta), pk_{PCA})$. The PCA sends pseudonym certificate registration request RegReq = sign(Enc({$Cert_i^{PCA}$,pk_{TLM}),pk_{PCA}) to the validator for registration in the TLM.
5. The Validator verifies the request and forwards RegReq = sign(Enc({$Cert_i^{PCA}$,pk_{TLM}),pk_{PCA}) to the TLM if not exceeding the number of requests generated by the RA.
6. The TLM generates signed-pseudonym-timestamp SPT_i = sign(($Cert_i^{PCA}$, t), sk_{TLM}) for each request and delivers RegConf =sign(Enc({$Cert_i^{PCA}$, SPT_i},pk_{PCA}), sk_{TLM}) back to the validator.
7. The PCA collects registration confirmation acknowledgment RegConf =sign(Enc({$Cert_i^{PCA}$, SPT_i},pk_{PCA}), sk_{VL}) from validator for each certificate.
8. The PCA generates package $Pkg_i = Enc((\{Cert_i^{PCA}, r_i\}, \hat{A}_i))$ and sends it back to the RA.
9. The RA assembles the encrypted packages for predefined time (one week). The RA bundles them for a given vehicle and provides encrypted packages to the given vehicle for download. The vehicle downloads the packages and calculates the corresponding decryption key $\hat{a}_i = a + f(i)$ say for package i and retrieves $Cert_i^{PCA}$ by decrypting the package $Dec(Pkg_i, \hat{a}_i)$. The vehicle verifies the certificate $verify(Cert_i^{PCA}, pk_{PCA})$, com-

putes the private-key $u_i = \hat{a}_i + r_i$ relating to the public-key U_i, and further checks $u_i.g \stackrel{?}{=} U_i$.

It is obvious from the above provision process, that even though RA knows the MEC of the vehicle that requests pseudonym as well as the RA hands over pseudonyms to the vehicle, still it is not able to extract information from the pseudonym packages as the PCA encrypt the packages to the vehicle as in [10].

4.3.3. Inserting MEC and pseudonym certificate into TLM database

Before each update, each TLM has two lists of credentials for adding into its database: 1) a list of new MECs, and 2) a list of new pseudonym certificates. During update, all fresh MECs and pseudonym certificates are added to the TLM's trees; the PP are updated, and the following operations are carried out on log trees.

1. Adding the pseudonym certificates to AT: Let L be the existing set of pseudonym certificates then the new accumulation value is calculated relative to the fresh set $L' = L \cup \{Cert_i^{PCA}, Cert_{i+1}^{PCA}, ..., Cert_{i+n}^{PCA}\}$, $Acc'_s = Acc_s^{(X+s)}$, where $X = \{Cert_i^{PCA}, Cert_{i+1}^{PCA}, ..., Cert_{i+n}^{PCA}\}$ is the new set and s is private key. For each pseudonym certificate, the membership evidence is refreshed as $W' = W_i^{(Cert_i^{PCA}+s)}$ for each pseudonym and the tree parameters are updated.
2. Adding the MEC to MHT: Create a new node for each MEC, insert the node in MHT in chronological form, also add the root of AT as the last node, and update root of the MHT.

The validator also countersigns the digests and PPs of TLM after each update and publishes it.

4.3.4. Secure communication in ACMS

After accomplishing the initial pseudonym say $Cert_i^{PCA}$ registration process, vehicle V retrieves the pseudonym non-revocation proof from TLM. When V broadcasts signed safety messages or starts communication with another vehicle, V transfers a $Cert_i^{PCA}$, along with TLM proof π_i. $Cert_i^{PCA}$ is validated using the PCA's public-key, and the proof is validated against the TLM's PPs. Algorithm 1 shows the communication and validation where the Pre-Validate function encompasses verification whether a certificate is expired or not, and d_i represents the digest (root hash of ith version) of the TLM.

Algorithm 1: Anonymous Communication.

```
Input: Cert_i^PCA, π_i, d_i, pk_PCA
Output: success or failure
begin
    if ((Pre-Validate(Cert_i^PCA) = 0)) then
        return failure
    end
    if ( VerElemMemb(Cert_i^PCA, π_i, d_i) = 0) then
        return failure
    else if (verify(Cert_i^PCA, pk_PCA) = 0) then
        return failure
    else
        success
    end
```

4.3.5. Pseudonym certificates resolution and revocation

When a vehicle starts misbehaving or gets compromised, the vehicle must be revoked and blacklisted. The validator sends pseudonym mapping request $MapReq = sign(Cert_j^{PCA}, sk_{VL})$ to PCA to map the pseudonym certificate of misconducting vehicle having pseudonym $Cert_j^{PCA}$ to the corresponding hash value $h(CSR_{RA2PCA})$ of the certificate signing request. The PCA sends back a reply response $MapReply = sign(h(CSR_{RA2PCA}), pk_{PCA})$ that consists of the corresponding hash value and the RA name to the validator. The RA maps the hash value of the pseudonym to the corresponding vehicle's enrollment certificate and puts it in a blacklist. The RA sends the $h(CSR_{RA2PCA})$ values of the corresponding vehicle to the validator along with the enrollment certificate. The validator forwards the set of hash values to the PCA for identifying the corresponding pseudonym certificates. It transfers them to TLM for revocation. The TLM generates signed-revocation-timestamp $SRT = sign(h(LinkageValueList), sk_{TLM})$ and sends back to it. The TLM revokes them by removing them from the accumulation tree in the next update, and the following operation is carried out on its trees.

1. Removing the pseudonym certificates from AT: Computes the fresh accumulation value for new list $L' = L \backslash X$, where $X = \{Cert_i^{PCA}, Cert_{i+1}^{PCA}, ..., Cert_{i+n}^{PCA}\}$ as $Acc'_s = Acc_s^{\frac{1}{X+s}}$.
2. Adding the AT root to MHT: The new updated root of AT is inserted as the last node to the MHT.

In order to track misbehaving vehicles, each RA keeps MEC, and the hash value of RA to PCA pseudonym certificate acquisition request $h(CSR_{SA2PCA})$, and each PCA keeps pseudonym certificates and $h(CSR_{SA2PCA})$ values.

4.3.6. Checks-and-balances among parties

All parties of ACMS require to verify that the authorities (e.g., PCA and EA) did not issue fake certificates and TLM did update her database correctly according to the MECs and pseudonym certificate registration requests it has received. These checks are done by a trusted validator, trust anchors (CA, and Electors), and clients (vehicles and RSUs) to watch malicious certificates and correctness of the TLM. Vehicles randomly send a set of pseudonym certificates to the validator to validate the authenticity of the certificates and correctness of the record in TLM. Similarly, RSUs can perform random checking on the set of certificates and records that they received during communications from the vehicles. If all checks are verified, the entire log is verified. Vehicles and RSUs need to perform the random checking periodically (e.g., once an hour or day). In brief, every party can contribute to detecting malicious actions and misconducting parties (e.g., PCA, EA, TLM, and vehicle compromises).

5. Security analysis

In this section, we conduct both formal and informal security analysis of ACMS. We presume that the cryptographic building blocks that we use, such as UBK, signature algorithms, hash function and encryption, and decryption functions are secure. We further presume that a vehicle V has correctly logged its pseudonyms at the TLM. More concretely, given any vehicle say V with a pseudonym certificate $Cert_i^{PCA}$, an adversary $\mathcal{A}$ attempts to act as an authorized participant say V to another vehicle say $V1$ during communication. The attack succeeds if $V1$ accepts $\mathcal{A}$ messages while believing that the messages are from a legitimate entity V.

$\mathcal{A}$ **compromises EA's private-key.** As long as less than the threshold number of EAs are corrupted, there is no chance that an illegal entity would join the systems. Even in the case, more than the threshold numbers of EAs are corrupted, it is highly likely that the validator would detect it, and it is highly unlikely that more than a threshold number of entities are corrupted. Hence, it is difficult for an adversary to compromise a threshold number of EAs.

$\mathcal{A}$ **compromises PCA.** Let suppose that PCA has issued a set of pseudonym certificates for $\mathcal{A}$. Let $\mathcal{A}$ send a maliciously acquired

$Game^{Proof}_{Forge}(\mathcal{A})$

1: $(Cert_i'^{PCA}, L_i, d_i, \pi_i') \leftarrow \mathcal{A}()$
2: **return** 1
3: $if((VerDigest(L_i, d_i) = 1) \wedge (VerifyElemMemb(Cert_i'^{PCA}, \pi_i', d_i) = 1) \wedge (Cert_i'^{PCA} \notin L_i))$

Fig. 5. Security experiment for generating membership witness to maliciously join communication as an authorized user.

$Game^{MHT}_{Forge}(\mathcal{A})$

1: $(d_i', L_j, \pi_{i,j}') \leftarrow \mathcal{A}()$
2: **return** 1
3: $if((VerAppendOnly(pk_{TLM}, d_i, i, d_j, \pi_{i,j}) = 1) \wedge (VerAppendOnly(pk_{TLM}, d_i', i, d_j, \pi_{i,j}') = 1)$
4: $\wedge\ (d_i \neq d_i'))$

Fig. 6. Security experiment for forging MHT to maliciously join communication as an authorized user.

certificate $Cert_i^{PCA\prime}$ during the handshake to $V1$. The $\mathcal{A}$ could not convince $V1$ because $V1$ verifies the certificate as well as executes VerElemMemb function which checks $e(W_{Cert_k^{PCA},L}, g^{Cert_k^{PCA}}, g^s) \stackrel{?}{=} e(Acc(L), g)$ shown in Algorithm 1 and accepts messages if it is registered, otherwise rejects. Since the certificate is not registered on the TLM so $VerElemMemb(Cert_i^{PCA}, \pi_i, d_i) = 0$ and the party $V1$ discards the messages received from $\mathcal{A}$. Hence, corrupting only PCA is not enough to conduct the attack against vehicle, which is formally proved in Theorem 1.

Theorem 1. *If q-strong DH assumption holds, then, a PPT adversary $\mathcal{A}$ cannot act as a legitimate user of the system by maliciously acquiring a fake but valid pseudonym certificate from a subverted PCA. More precisely, if $\mathcal{A}$ wins the attack game $Game^{Proof}_{Forge}(\mathcal{A})$ by outputting $(Cert_i'^{PCA}, L_i, d_i, \pi')$, then $\mathcal{A}$ finds a fake membership proof of a non-member pseudonym of L_i with respect to $Acc_s(L_i)$ with probability at most $O(\frac{1}{p})$.*

Proof. Let PCA issues a set of valid but fake pseudonym certificate $Cert_i'^{PCA}$ for $\mathcal{A}$. Since the pseudonym must be logged into the TLM to be accepted by vehicles and clients (see Algorithm 1), $\mathcal{A}$ must win security game given in Fig. 5 to participate in the communication system as an authorized user. Suppose $\mathcal{A}$ wins $Game^{Proof}_{Forge}$ by producing a fake membership evidence $\pi_i' = W'_{Cert_i'^{PCA},L_i}$ with respect to $Acc_s(L_i)$ and with non-negligible probability such that $VerifyCrdlMem(Cert_i'^{PCA}, \pi_i', d_i) = 1$ and $Cert_i'^{PCA} \notin L_i$. Let $P = Cert_i'^{PCA}$ (only for brevity), then, $W'_{P,L_i} = Acc_s(L_i) = g^{f_{L_i}(s)}$, where $f_{L_i}(s) = \sum_{i=0}^{|L|}(e_i.s^i)$, with e_i is a known coefficient that relies on the members of L_i such that $0 \leq i \leq |L|$. Since $P \notin L_i$ that is, $(P + s)$ does not divide $f_{L_i}(s)$. Hence, applying polynomial division while having L_i and P, $\mathcal{A}$ can compute a non-zero integer e and a polynomial $q(s)$ of degree $|L_i| - 1$ which results in $f_{L_i}(s) = e + q(s).(P + s)$. Therefore, $W'_{P,L_i} = g^{q(s)}.g^{\frac{e}{(P+s)}}$, which results in $g^{\frac{1}{(P+s)}} = [W'_{P,L_i}.[g^{q(s)}]^{-1}]^{e^{-1}}$ efficient computation using TLM public-key, which contradict q-strong DH assumption. □

$\mathcal{A}$ **compromises TLM's secret key.** If we assume that TLM is not compromised, then all the above mentioned attacks can successfully be resisted since all of the $\mathcal{A}$'s malicious actions would become transparent and publicly visible. Compromising only TLM is not sufficient to join the infrastructure illegitimately and act as an authorized party in the communication. Nevertheless, in worst-case where $\mathcal{A}$ has corrupted a PCA and the TLM, then $\mathcal{A}$ could act as a legitimate client say vehicle V to $V1$ by creating fake pseudonym and proof. In such a case, the $\mathcal{A}$ would need to create two different signed versions of the TLM's log, which can easily be detected and caught by the validator as it signs off the TLM PPs and also RSUs and vehicles occasionally verify TLM's public parameters with the validator.

$Game^{TLM-Ver}_{Hidden}(\mathcal{A})$

1: $(Cert_i'^{PCA}, d_i', L_i', pk_{VL}) \leftarrow \mathcal{A}()$
2: **return** 1
3: $if((VerDigest(L_i', d_i') = 1) \wedge (verify(d_i', pk_{VL})) = 1) \wedge (d_i \neq d_i'))$

Fig. 7. Security experiment for creating hidden version of TLM database to maliciously participate in communication as an authorized user.

Theorem 2. *If H (has function) is collision-resistant and signature scheme is (euf-cma) unforgeable, then, a PPT adversary $\mathcal{A}$ cannot act as legitimate user of the system by subverting PCA and TLM. More precisely, if $\mathcal{A}$ wins the attack game $Game^{MHT}_{Forge}(\mathcal{A})$ or the attack experiment $Game^{TLM-Ver}_{Hidden}(\mathcal{A})$, then $\mathcal{A}$ either finds a single MHT digest (root hash) under two different lists of elements with non-negligible probability or forge the signature with non-negligible probability.*

Proof. Let $Cert_i'^{PCA}$ is maliciously issued pseudonym certificate by a dishonest PCA and logged by a subverted TLM (see Algorithm 1). The intruder must win either of the security games defined in the below two cases.

Case 1: If $\mathcal{A}$ wins $Game^{MHT}_{Forge}$ given in Fig. 6 by outputting $(d_i', L_j, \pi_{i,j}')$, then $TLM^{\mathcal{A}()}$, which runs $\mathcal{A}$ and outputs $(d_i', L_j, \pi_{i,j}')$ can successfully forge MHT. Suppose $\mathcal{A}$ wins $Game^{MHT}_{Forge}$ by generating same append-only proof with respect to two different lists L_i with digest d_i and L_i' with digest d_i' at version i with non-negligible probability such that $VerAppendOnly(pk_{TLM}, d_i, i, d_j, \pi_{i,j}) = 1$, $VerAppendOnly(pk_{TLM}, d_i', i, d_j, \pi_{i,j}') = 1$ and $d_i \neq d_i'$. This directly follows from the H (collision-resistance). Assume that the vehicles and other parties receive two digests at version i $d_i \neq d_i'$ with respect to lists L_i and L_i', and the version j digest d_j such that $(VerAppendOnly(pk_{TLM}, d_i, i, d_j, \pi_{i,j}) = 1$ and $VerAppendOnly(pk_{TLM}, d_i', i, d_j, \pi_{i,j}') = 1$ hold. This implies that either $L_i = L_i'$ or the adversary generates the same d_j under two different lists of elements. Since $d_i \neq d_i'$ so $L_i \neq L_i'$, this implies that $\mathcal{A}$ generates collision for MHT, which by lemma 1 is collision in H.

Case 2: In this scenario, the TLM maintains two different databases–one database hidden from the validator containing the fake credential $(Cert_i'^{PCA})$ and another without (authentic). Since the digest and public parameters of TLM are countersigned by the validator (see Section 4.3.3), the $TLM^{\mathcal{A}()}$, which runs $\mathcal{A}$ must win security experiment given in Fig. 7 to convince clients accept the malicious pseudonym. Suppose $\mathcal{A}$ wins the game $Game^{TLM-Ver}_{Hidden}(\mathcal{A})$ by producing $(Cert_i'^{PCA}, d_i', L_i')$ such that $VerDigest(L_i', d_i') = 1)$ and $(verify(d_i', pk_{VL})) = 1)$ hold, then $TLM^{\mathcal{A}()}$ which runs $\mathcal{A}()$ can successfully forge the employed signature (EUF-CMA) scheme. For this we create another adversary (signature forger) $\mathcal{F}$, and the corresponding game which is given in Fig. 8.

$\mathcal{F}$ simulates $Exp^{euf-cma}_{SIG}(\mathcal{F})$ for $\mathcal{A}$, using it's $Osign$ for generating the signature. First of all, $\mathcal{F}$ initializes an empty list. It then starts simulating experiment $Exp^{euf-cma}_{SIG}(\mathcal{F})$ for $\mathcal{A}$, providing the

$Exp_{SIG}^{euf-cma}(\mathcal{F})$	$Osign(m)$
1: $(pk, sk) \xleftarrow{\$} KeyGen()$	1: $\sigma \xleftarrow{\$} sign(sk, m)$
2: L $\leftarrow$ {}	2: L $\leftarrow$ L $\cup$ {m}
3: $(m, \sigma) \xleftarrow{\$} \mathcal{F}^{Osign(pk)}$	3: return (m, σ)
4: return 1	
5: $if(verify(m, pk) = 1) \wedge (m \notin L))$	

Fig. 8. Security experiment for forging the EUF-CMA (existential unforgeability under chosen message attacks) signature scheme.

validator public-key pk_{VL} from experiment $Exp_{SIG}^{euf-cma}(\mathcal{F})$ as an input for $\mathcal{A}$. It invokes its signing oracle Osign whenever required to obtain a signature in the execution of simulations and maintains a list containing the queried values to the signing oracle. Whenever $Game_{Hidden}^{TLM-Ver}(\mathcal{A})$ halts, $\mathcal{F}$ wins and generates the fake but valid signature forgery, thus contradicting the security of the signature (EUF-CMA secure) scheme. □

Sybil attack prevention. A malicious vehicle's OBU might try to get multiple sets of pseudonyms from PCA to impersonate multiple OBUs. Such attacks are thoroughly mitigated validation of MEC by RA and the validator before approving pseudonym provision and logging to PCA and TLM, respectively. When an RA receives CSR signed from a vehicle along with an enrollment certificate, the RA first checks whether the MEC is not revoked, and this is the only CSR by the vehicle during the specified interval of time. After initial validation, the RA forwards the enrollment certificate to the validator, where the validator repeats the status checking and multiple acquisitions by the vehicle (see Section 4.3.2). The vehicle can get a logged set of pseudonym certificates if it passes both checks of RA and the validator.

Moreover, each vehicle is granted a pre-defined number of pseudonyms per week, all enrollment and pseudonym certificates are logged in TLM, and anyone can monitor both sets. It also makes it easy for any monitoring party to detect Sybil attacks by comparing the total number of valid pseudonyms to the total number of legitimate vehicles. PCA also issues pseudonyms with disjointed lifetime; in short, no car can acquire more than one fake but valid set of pseudonyms at any moment; hence, Sybil attack is thoroughly prevented.

Long-Term Unlinkability. To offer privacy, the real identity should not be linked to the pseudonyms set with that vehicle in the long-term. Shuffling over UBK expanded keys, encryption of packages to vehicles, and pseudonyms guarantee long-run unlinkability in ACMS. RA enables vehicles to get pseudonyms from PCA without revealing its real identity, encryption of linkage values to PCA and encryption of packages to vehicles prevents RA to know the set of pseudonyms belonging to the vehicle while also pseudonyms prevent other vehicles from knowing that the signed messages are from the which vehicle.

Proof Using Tamarin Prover. We have validated the security properties of the proposed V-PKI scheme using machine-checked formal verification tool Tamarin Prover [26]. It is a formal verification tool based on the symbolic representation of security protocols that offers unbounded verification and falsification. During modeling this scheme, we have abstracted some concepts as done in [17,19,18], and only modeled security-related properties. The Tamarin Prover has built-in support for the Dolev-Yao adversary model where the adversary is powerful enough such that it can inject, intercept, modify as well as delete communication messages. It uses first-order logic language to specify security properties in which (All, Ex) works as quantifiers and (|,==>, &, not) acts as logical connectors. The security goals in Tamarin Prover are modeled as lemma in which #i denotes the occurrence of an event at the time i. The lemma given below starts with variables (connid V V1 VL,...) and for these variables values there should be at timepoint i1 an action trace $Gen_Key(...)$ which generates authentic key. The vehicle V has acquired a pseudonym with public-key "oldkey" by requesting it from PCA. The lemma models the core security goal, where the goal holds if the V communicates with another party say vehicle V1 but the attacker does not know the key represented as KU(key).

```
lemma validate_secure_operations:
“
(
All connid V V1 VL oldKey key #i1.
        ( Gen_Key(V, oldKey, ‘authentic’) @ #i1
        & RequestPseudoCert(V, oldKey) @ #i2
        & ReceivedPseudoCert(V, oldKey) @ #i3)
        & Communication(connid,V,V1, validate, key) @ #i4)
)
        ==>
        (
        not (Ex #i5. KU(key) @ i5)
        )
“
```

Analysis. In the protocol verification, we identified expected attacks by compromising the PCA. We found that the proposed framework can guard vehicles against attacks in case of PCA compromise. We can conclude that an adversary capable of controlling either PCA, TLM and the validator or PCA and RA can breach security and privacy respectively while leaving designing such V-PKI that can prevent such attacks for our future work.

6. Performance evaluation

For evaluating the performance of ACMS, first, we conduct a comprehensive comparison of proposed scheme revocation mechanism with CRL and then compare ACMS with CRL-based V-PKI [10] and SA-KMP [36] schemes. More specifically, the performance evaluation process is broken down into three parts. The first part examines the revocation cost, latency, communication overhead, and scalability of ACMS, SCMS [10], and SA-KMP [36] on the basis of analytical simulation. The second part evaluates the average message (packet) loss ratio and end-to-end delay of aforementioned V-PKI proposals on the basis of ns2[6] and SUMO[7] [58]. At last, we also provide a comparison with some leading proposals through various performance metrics and also elaborate the computational and space complexity of TLM.

Table 3 shows the notation and typical values used in the analytical evaluation. Values for V_n, V_r, $T_{Cert_i^{PCA}}^{expiry}$, and PS_{size} are taken from [34–36] while the values for T_{ECDSA}^{sign}, T_{ECDSA}^{verify} are computed and averaged over 10,000 of runs in Java using our personal computer with core i7 CPU @1.8 GHz, running Ubuntu 18.04 with 8GB RAM. Similarly, the cost of a single bilinear pairing operation is calculated using library[8] with the same configuration machine and averaged over 10,000 of runs. Moreover, we presume standard algorithms for hashing, bilinear pairing, and digital signatures. Notably, we assume that the following standard algorithms are applied: hashing is performed with SHA256 hash function; Ate pairing with 256 curve over Barreto-Naehrig (BN) is used for bilinear pairing, and ECDSA with secp256k1 curve (with signature size 64 bytes) is employed as a digital signature algorithm.

[6] The Network Simulator–ns-2: http://www.isi.edu/.
[7] Simulation of Urban MObility: http://sumo.sourceforge.net/.
[8] https://github.com/herumi/ate-pairing.

Table 3
Parameters adapted for analytical simulation.

Parameters	Description	Value
V_n	Number of vehicles	10^2 *to* 10^6
V_r	Estimated percentage of vehicles revoked per day	10 %
q	Estimated number of vehicle status queries per day	10^6
l_h	Hash function digest length	32 bytes
l_{sig}	Digital signature length	64 bytes
l_{lnk}	Linkage value length	9 bytes
T^{sign}_{ECDSA}	Time of a digital signature generation	1.04 ms
T^{verify}_{ECDSA}	Time of a signature verification	1.64 ms
$T^{expiry}_{Cert_i^{PCA}}$	Time to check expiry of the certificate	0 ms
PS_{size}	Size of pseudonym certificate	126 bytes
$T_{Pairing}$	Cost of a single bilinear pairing operation	1.23 ms

Table 4
Revocation overhead in bytes.

V_n	V_r	CRL_{UpCost}	$ACMS_{UpCost}$	CRL_{QuCost}	$ACMS_{QuCost}$
10^2	10	204	67	2.04×10^8	3.2×10^6
10^3	10^2	1014	89	1.014×10^9	3.2×10^6
10^4	10^3	9114	311	9.114×10^9	3.2×10^6
10^5	10^4	9×10^4	2530	9.011×10^{10}	3.2×10^6
10^6	10^5	9×10^5	24722	9.001×10^{11}	3.2×10^6

6.1. Communication cost of revocation

We estimate the daily update and query cost of CRL and ACMS. The CRL update overhead is $CRL_{UpCost} = CRL_{header} + (l_{lnk} \times V_r) + l_{sig}$, since the revoking authority (CA) sends the entire CRL to the directory (distribution point) in each update, where the CRL header CRL_{header} is 50 bytes in size. The ACMS update cost is $ACMS_{UpCost} = \frac{V_r \times l_{lnk}}{365} + l_{sig}$ since validator sends list of different length each day to TLM. Similarly, CRL daily query cost is $CRL_{QuCost} = V_n \times CRL_{size}$ since for every query the CRL distributer sends the entire CRL to the requesting client. While the proposed scheme sends a proof of 32[9] bytes to answer the user's query, totaling $ACMS_{QuCost} = V_n \times 32$. As shown in Table 4, ACMS costs are much lower than CRL overhead both in authority-to-directory and directory-to-clients communication. Again, vehicles do not need to save any revocation information in the proposed scheme while in case of CRL vehicles need to grant memory equal to the CRL size to save revocation information.

6.2. Processing and handshake latency

Stepping forward, we compute the processing latency induced by CRL and ACMS on verifiers. Since SCMS is based on CRL, that is SCMS inherits the delay induced by the CRL. Furthermore, we only calculate computational latency and ignore transmission delay as well as queueing delay. In CRL-based proposals, the processing latency equals to searching time of pseudonym certificate serial number in the CRL which is O(n), where n depends on the number of revoked vehicles and the processing time increases linearly with the increase in the number of revoked vehicles. Fortunately, the proposed scheme eliminates this scalability concern of V-PKI by introducing a new data structure (accumulation tree) which has a constant proof size and constant processing time that is 32 bytes and two bilinear pairing operations. We presume the processing latency of CRL as that benchmark in [35,36], whereas a single bilinear pairing operation takes 1.23 ms. Hence, proof verification takes 2.46 ms because proof of membership verification involves two bilinear pairing operations. Table 5 shows that ACMS has a constant processing latency independent of the number of revoked vehicles while the CRL latency increases with the increasing number of revoked vehicles.

Table 5
Processing latency.

V_r	T^{search}_{CRL}	T^{proof}_{ACMS}
10	0.023 ms	2.46 ms
10^2	0.071 ms	2.46 ms
10^3	0.691 ms	2.46 ms
10^4	7.910 ms	2.46 ms
10^5	97.297 ms	2.46 ms

We now proceed to estimate the handshake processing time of SCMS and ACMS while the values for SA-KMP are taken from the article [36]. The handshake in SCMS consists of time to sign a message, time to check a pseudonym certificate revocation status in a CRL, three times verifying the signature (once the signed message, PCA signature verification on pseudonym certificate, and signature verification on a CRL) and validating the pseudonym certificate expiry date and the time is expressed as.

$$T^{SCMS}_{Process} = T^{sign}_{ECDSA} + T^{expiry}_{Cert_i^{PCA}} + T^{search}_{CRL} + T^{verify}_{ECDSA} \times 3 \quad (4)$$

However, if the CRL is already saved and the signature is verified then the above equation becomes as.

$$T^{SCMS}_{Process} = T^{sign}_{ECDSA} + T^{expiry}_{Cert_i^{PCA}} + T^{search}_{CRL} + T^{verify}_{ECDSA} \times 2 \quad (5)$$

Equation (6) depicts the handshake processing time in ACMS which includes all the steps in equation (5) except the searching of CRL, where proof of membership of pseudonym certificate in TLM is verified instead of searching it in the CRL.

$$T^{ACMS}_{Process} = T^{sign}_{ECDSA} + T^{expiry}_{Cert_i^{PCA}} + T^{proof}_{ACMS} + T^{verify}_{ECDSA} \times 2 \quad (6)$$

Applying the values in Table 5 and Table 3, we obtain the handshake latency of 101.617 ms and 6.78 ms for SCMS and ACMS respectively, and the handshake value for SA-KMP is 12.309 ms.

6.3. Scalability

In this subsection, we assess the scalability of ACMS, SCMS, and SA-KMP and compare their suitability for vehicular communication in terms of RSU service ratio, latency, and numbers of authentication. It is often the prime priority besides the security and privacy

[9] Since G size for ATE pairing with security level 128 over Barreto-Naehrig (BN) is 256 bits.

that the new scheme must be scaled in terms of support for a vast amount of relying vehicles. First, we compute RSU service ratio to demonstrate that ACMS is more scalable. In order to simulate a more practical scenario we borrow the analysis approach and some parameters from [59] which are given below.

- The speed of vehicles (denoted as s) is 10 m/s to 50 m/s and probability of each vehicle to issue a request is $\rho = 0.8$.
- RSU has a communication range (denoted as R_{range}) of 300 m.
- The vehicle density (vehicles/km^2) ranges from 10 to 1000 and is denoted as d.

Let (V_X) be the number of requesting vehicles, then (V_X) follows the Binomial distribution $\mathcal{B}(d, \rho)$ with respect to the ρ, and we have.

$$P\{V_X = v\} = \binom{d}{v} \rho^v (1-p)^{d-v}, v = 0, 1, 2, ..., d \tag{7}$$

and the average number of service requests can be modeled as the mathematical expectation

$$E\{V_X\} = \sum_{v=0}^{d} \binom{d}{v} \rho^v (1-p)^{d-v} = d.\rho \tag{8}$$

So $S_{req} = E(V_x) = d.\rho$ and the maximum servicing capacity of RSU can be modeled as

$$S_{max} = \frac{R_{range}}{s.T_{Process}} \tag{9}$$

and $T_{process} \in \{T^{SCMS}_{Process} = 101.617,\ T^{ACMS}_{Process} = 4.916,\ T^{SA-KMP}_{Process} = 12.309\}$.

Then we calculate the amount of actual processed service request (denoted as S_{act}) as

$$S_{act} = \begin{cases} S_{req}, & if\ S_{req} < S_{max}, \\ S_{max} & otherwise \end{cases} \tag{10}$$

Therefore, serving ratio (represented as S_{ratio}) is defined as

$$S_{ratio} = \frac{S_{act}}{S_{req}} \tag{11}$$

Then, the S_{ratio} can be computed as

$$S_{ratio} = \begin{cases} 1, & if\ \frac{R_{range}}{s.T_{Process}.d.\rho} \geq 1, \\ \frac{R_{range}}{s.T_{Process}.d.\rho} & otherwise \end{cases} \tag{12}$$

Fig. 9a illustrates the RSU servicing ratio of the three schemes. From the result, it is clear that ACMS has better performance than both schemes and that RSUs can efficiently handle requests in most cases. Fig. 9b shows the successful number of authentications per second when ACMS, SCMS, and SA-KMP are applied for secure communication. The graph clearly represents that the capacity of SCMS to authenticate vehicles drops significantly with the increasing number of revoked vehicles (i.e., large size of CRL). On the other hand, both ACMS and SA-KMP are independent of revoked vehicles; ACMS also successfully eliminates these scalability issues of traditional CRL-based V-PKI schemes. Fig. 9c compares the latency incurred during the communication among vehicles in SCMS, ACMS, and SA-KMP. The graph depicts that SCMS delay increases sharply when the number of revoked cars reaches 10,000. This sharp jump in the latency is because of a handful of entries in a CRL that increases linearly with the number of revoked vehicles causing considerable delay to search for the revoked serial number. On the other side, the increase in the number of revoked vehicles has no impact on latency in V2V as well as V2X communication in the proposed credential management system and SA-KMP scheme. Therefore, we can conclude that ACMS is more practical and scalable than both schemes under consideration.

Table 6
Simulation parameters.

Parameters	Value
Simulation area	$2000m \times 2000m$
Simulation time	500 s
Number of vehicles	500
Communication range	300 m
Max. speed	25 m/s
Payload size	200 bytes
MAC protocol	802.11p
Channel bandwidth	6 Mb/s

6.4. Message loss and end-to-end delay

To further test the feasibility of ACMS, we conducted simulation for each scheme using ns-2.35 and SUMO-1.3.1. The simulation was done on a machine with core i7-4790 CPU at 3.6 GHz, running Ubuntu 18.04 with 16 GB RAM. The simulation scenario was generated for Beijing city, which was selected using OpenStreetMap.[10] The simulation is performed for safety applications (e.g., EEBL) where each OBU needs to broadcast BMS every 100~300 ms as per Dedicated Short Range Communication (DSRC) standard [60,61]. The parameters adapted for ACMS, SCMS, and SA-KMP simulation are given in Table 6 and the average message loss ratio (MLR) and average end-to-end delay are used as performance metrics.

Fig. 10a and Fig. 10b compare the average MLR when each vehicle broadcasts every 100 ms and 300 ms respectively. It can be observed that the average MRL increases directly with verification time and traffic density for the V-PKI schemes under consideration. Also, using ACMS for message authentication decreases the MLR compared to using SCMS and SA-KMP in both cases. It can be seen from Fig. 10a and Fig. 10b that SCMS with large CRL size (10^5) has the highest MLR because searching large size CRL takes longer time which results in excessive packet loss. It can be concluded that CRL is the primary bottleneck of CRL-based V-PKI schemes such as SCMS.

Lastly, Fig. 10c shows the average end-to-end delay, which includes the time to sign, transmit and verify. It is crucial to guarantee low end-to-end delay that can satisfy the latency requirement of safety applications in vehicular communication [62–64]. It can be seen that increase in traffic load (number of vehicles in communication range) causes increase in the average end-to-end latency. The average latency for ACMS and SK-KMP can satisfy the maximum allowable limit of delay while SCMS can fulfill the latency requirement if the CRL size is less than 10^5.

6.5. Comprehensive comparison of approaches

Table 7 compares ACMS with some leading CRL-based V-PKI schemes such as SCMS [10], IEEE VPKI [34], SECMACE [9], and registry-based schemes such as PKR [35], SA-KMP [36] using various metrics. For the "Storage cost in terms of storing public-keys" metric, the proposed and CRL-based V-PKI schemes are more efficient as compared to the registry-based schemes because in these proposals vehicles require to save only their pseudonym certificates and the memory granted by each vehicle equals to $number\ of\ pseudonym\ certificates \times PS_{size}$. Table 7 shows that the proposed and CRL-based schemes has less than 1 MB storage even when a vehicle saves pseudonym certificates for a year (i.e., 1040

[10] https://www.openstreetmap.org.

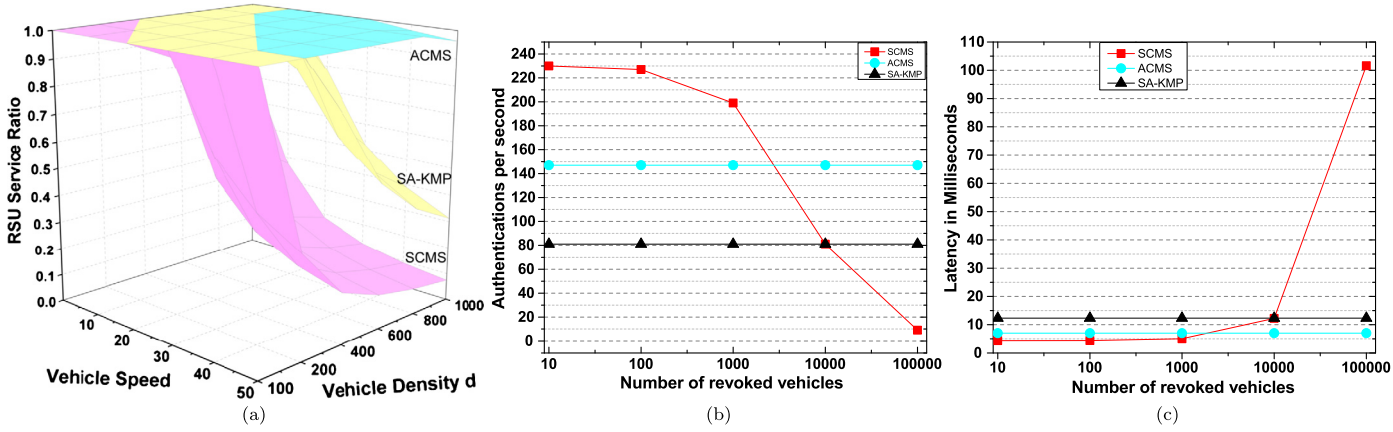

Fig. 9. Plots showing the Scalability in terms of service ratio, authentication, and latency. Number of revoked vehicles (V_r) is 10% of total number of vehicles (V_n), which is equal to CRL size of SCMS.

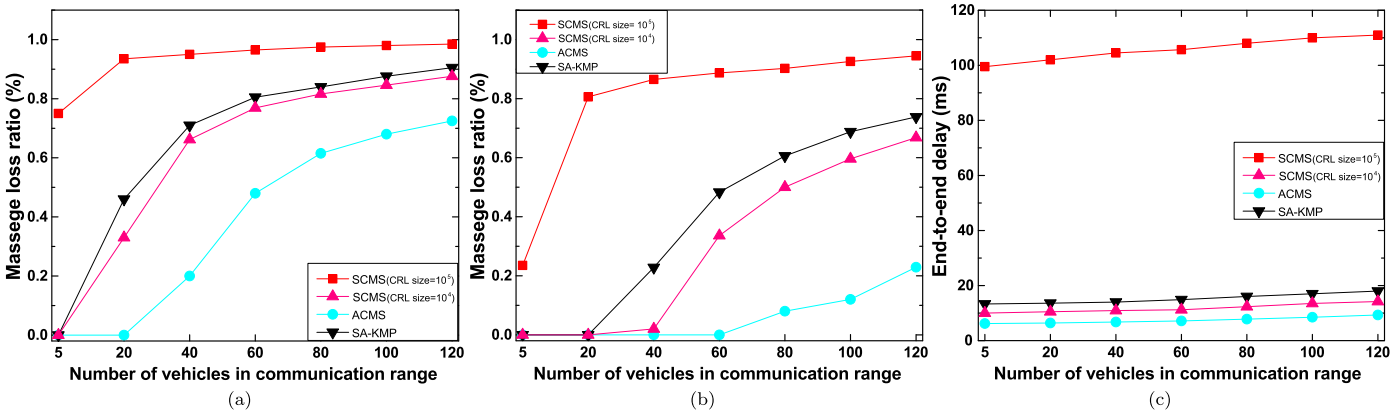

Fig. 10. Plots showing message loss ratios and end-to-end delay.

Table 7
Comprehensive comparison of state-of-the-art V-PKI proposals.

Parameters	ACMS	CRL-based schemes			Registry-based schemes	
		IEEE V − PKI [34]	*SCMS* [10]	*SECMACE* [9]	*PKR* [35]	*SA − KMP* [36]
Storage cost in terms of storing public-keys per year	0.131 MB	0.131 MB	0.131 MB	0.131 MB	33 MB	101.7 MB
Storage cost in terms of storing public-keys per weak	2.46 KB	2.46 KB	2.46 KB	2.46 KB	33 MB	101.7 MB
Induced latency	4.916 ms	101.617 ms	101.617 ms	101.617 ms	13.462 ms	12.309 ms
Max. number of authentication support	203	230	230	230	74	81
Min. number of authentication support	203	9	9	9	74	81

pseudonym certificates). On the other hand registry-based techniques need to store all vehicles keys and the space occupied is equal to $(V_n \times (key\ size + vehicle\ Id\ size))$ in PKR. While in SA-KMP, the memory granted by each vehicle for maintaining vehicles and RSUs keys are $(V_n \times (key\ size + vehicle\ Id\ size + sig_l))$ and $(no.of\ RSUs \times (key\ size + RSUs\ Id\ size))$ respectively.

We also analyze latency, and the numerical results show that the latency of CRL-based V-PKI schemes is several orders of the magnitude greater than the delay induced by the proposed V-PKI framework and registry-based techniques. Moreover, the latency induced by the proposed work is lower than that of the registry-based schemes. Finally, we investigate the maximum number of authentication per second when an enormous amount of vehicles are revoked in a less verbose way. Regarding authentication, any technique that has higher latency is less scalable in terms of it. For example, CRL-based schemes are less scalable because of huge latency with the increasing number of vehicles being revoked. CRL-based V-PKI approaches achieve maximum number of authentication when the CRL has minimum entries while the authentication support drop with a large CRL.

6.6. Enrollment certificate issuance

To demonstrate the feasibility of our proposed multi-signature enrollment certificate issuance, we built a typical scenario. The scenario consists of 1) an RSU that requests an enrollment certificate (MEC) from EAs, 2) a set of two EAs that issues enrollment certificate, 3) a validator, and 4) a TLM. Because the EAs have a similar role to standard CAs, so they are implemented by using OpenSSL in our requirement. This experiment is conducted on mini-network with five computers, and setup is connected to the Internet via 10 Mbit/s WiFi. Similarly, an instance of a TLM is implemented by modifying an already existing Merkle tree[11] implementation in Python. To get a certificate, an RSU sends a

[11] https://github.com/jvsteiner/merkletree.

Table 8
Computational complexity of ACMS data structures.

Metrics	*MHT*	*AT*
Member insertion Cost	$O(log_2(n))$	$O(c)$
Proof generation cost	$O(log_2(n))$	$O(log_2(c))$
Proof verification cost	$O(log_2(n))$	$O(1)$
Proof size	$O(log_2(n))$	$O(1)$

certificate request to EA1 and EA2. EA1 and EA2 endorse the certificate for the RSU upon receiving the request. The whole process takes around 100 milliseconds, averaged over 100 runs. We used standard X.509 certificates to handle MEC as in [65].

6.7. Transparency log maintainer

In this section, we highlight the significance of the TLM proofs requiring space/time proportional to O(log n) and O(1) rather than O(n), by calculating some typical values.

Storage. We suppose TLM is required to store one billion (10^9) pseudonym certificates that are registered with the server over a year. This requires storing all 10^9 pseudonyms, which are roughly in the order of $10^9 \times P_{size}\ bytes \approx 126GB$, which grows over time. Again the size of proof of a Merkle tree depends on the number of pseudonyms and the hash function used in computing the TLM database while the Accumulation tree has a constant proof size. In ACMS, SHA256 is used as a hash function, and the number of pseudonyms is 10^9, while each entry contains a batch of 2739,727 pseudonym certificates; the proof consists of around 20 hash values, together with a signed tree root. This is about 1KB data in case of MHT while AT has constant membership and non-membership proof where the size of membership is $W_{gpk_k,X} \in G$ is 256 bits, and the size of non-membership is double of membership proof i.e., 512 bytes respectively.

Computation. Insertion and update each on MHT involves about 20 operations, as discussed in the previous subsection, which will take negligible time. Similarly, verifying Merkle proof requires 20 hash operations, which takes a negligible time. On the other hand, the accumulation tree needs O(c) time for each operation where c is the number of valid pseudonym certificates, but verifying proof takes O(1), which involves two pairing operations. To evaluate the MHT cost, we implement it in Python with one million nodes; proof generation time averaged 240 µs, while verification time averaged 1.85 ms. Similarly, we also compute accumulation tree membership and non-membership witness verification time. Both involve two pairing operations while a single pairing operation using (https://github.com/herumi/ate-pairing) pairing library takes 3069054.57 clock cycles, which on our personal computer takes 1.23 ms. Hence the cost of verification of membership and non-membership is 2.46 ms. Note, although the accumulation tree has higher computation cost as compared to MHT proof verification, it guarantees constant proof size independent of the number of entries in TLM. The computational and space complexity of both trees used in ACMS are highlighted in Table 8.

7. Discussion

EA and PCA. ACMS reduces the trust placed in EAs and PCA and makes them firmly accountable to the public. EAs alone cannot convince RAs, PCAs, and validators to issue and publish pseudonym for a vehicle as in traditional VPKI where only an EA signed enrollment certificate is enough to convince RA and PCA to endorse pseudonyms for a vehicle. Entities with MEC signed by a threshold number of EAs are allowed to join the network by RAs, PCAs, and validators. Similarly, vehicles only accept pseudonyms signed by PCA and having an inclusion proof (logged in TLM). RCAs need to monitor TLMs and should actively cooperate with other parties (e.g., electors and traffic police department) to prevent false credentials registrations.

Organizational separation of trust servers. In order to preserve a vehicle's privacy against insider attackers, some components must not run by the same organization. If one organization would run PCA and RA, the organization could link the set of pseudonym certificates issued for any vehicle. Similarly, if one organization would run PCA and EA, the organization could disclose the real identity of a vehicle, and if one organization would run PCA, TLM, and validators, the organization could register fake pseudonyms.

Detection of successful attacks. ACMS offers stout security against attacks, specifically that a successful attack is only possible when at least a threshold number of trusted insiders are corrupted by the attacker. If such a compromise happens, the adversary can then create a fake pseudonym certificate along with fake inclusion proof. Thereupon, vehicles would start communication with the adversary based on the fake set of pseudonym certificates and proof. However, we believe that clients could expose such attacks because vehicles periodically validate their views of TLMs with RSUs and validators and through gossip with each other.

Blockchain-based Solution. Lately, different blockchain-based solutions for web PKI have been proposed, but none of them gain widespread adoption because they need a massive amount of storage. For example, in Certledger [66], the blockchain's size is 512 GB for approximately 10^8 certificates, and the blockchain's size would be 1260 GB for 10^9 number of pseudonyms. They also increase the handshake message size, such as Cerledger, the message size is 14 KB. Hence, we believe transparency log-based proposals are more suitable than blockchain-based solutions from our literature study of web PKI proposals. Although ACMS is built on a transparency log, its design is flexible and can be instantiated with most ledger technologies such as blockchain if it can satisfy the efficiency requirements of vehicular communication.

8. Conclusion

In this study, we introduced an Accountable Credential Management System for vehicular communication, with a particular emphasis on security and key transparency in V2V communication. This ACMS design balances the absolute authority of CAs (e.g., EA and PCA) and make them accountable by maintaining a transparent history of both enrollment and pseudonym certificates. Even though ACMS builds on previous works, in particular transparency log proposals [14–17] and SCMS [29,10], it leads to comparatively stronger security and cost-effective credential management system. ACMS achieves the security goals obtained by leading PKI proposals (e.g., AKI) and offers the level of privacy provided by today's leading V-PKI candidates such as SCMS. We conducted both informal and formal security verification of ACMS to verify its core security feature against an active adversary under the Dolev-Yao adversary model.

In terms of performance, one benefit of ACMS over SCMS refers to the elimination of CRL distribution. Through comprehensive simulation and experimental results, we showed that ACMS is highly scalable and practical. Also, the storage and cost analysis of TLM demonstrates that it is feasible for practical use. Additionally, ACMS provides a significant starting point towards designing a highly secure and accountable V-PKI framework. Future work would be defining an efficient method of how to add and revoke the certificate of a trust anchor and to implement our proposed V-PKI in the real-world to carry out detailed usability and performance analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work is partially supported by China National Key Research and Development Program No. 2016YFB0800301, and National Natural Science Foundation of China “NSFC” No. 61872041.

References

[1] I. García-Magariño, S. Sendra, R. Lacuesta, J. Lloret, Security in vehicles with iot by prioritization rules, vehicle certificates and trust management, IEEE Int. Things J. 6 (4) (2019) 5927–5934.
[2] K. Lin, C. Li, G. Fortino, J.J. Rodrigues, Vehicle route selection based on game evolution in social internet of vehicles, IEEE Int. Things J. 5 (4) (2018) 2423–2430.
[3] T.A. Butt, R. Iqbal, S.C. Shah, T. Umar, Social internet of vehicles: architecture and enabling technologies, Comput. Electr. Eng. 69 (2018) 68–84, https://doi.org/10.1016/j.compeleceng.2018.05.023, http://www.sciencedirect.com/science/article/pii/S0045790617336996.
[4] B. Jain, G. Brar, J. Malhotra, S. Rani, S.H. Ahmed, A cross layer protocol for traffic management in social internet of vehicles, Future Gener. Comput. Syst. 82 (2018) 707–714, https://doi.org/10.1016/j.future.2017.11.019, http://www.sciencedirect.com/science/article/pii/S0167739X17312694.
[5] J. Petit, F. Schaub, M. Feiri, F. Kargl, Pseudonym schemes in vehicular networks: a survey, IEEE Commun. Surv. Tutor. 17 (1) (2014) 228–255.
[6] I. ETSI, Intelligent transport systems (its); security; threat, vulnerability and risk analysis (tvra), Tech. rep., Technical report, ETSI TR 102 893, European Telecommunications Standards …, 2010.
[7] T. ETSI, Etsi ts 102 867 v1. 1.1-intelligent transport systems (its); security; stage 3 mapping for ieee 1609.2, Standard, TC ITS, 2012, p. 30.
[8] E. Annex, 4.1: why sign data instead of using a message authentication code, Available: standards.ieee.org/findstds/standard/1609.2-2013.html.
[9] M. Khodaei, H. Jin, P. Papadimitratos, Secmace: scalable and robust identity and credential management infrastructure in vehicular communication systems, IEEE Trans. Intell. Transp. Syst. 19 (5) (2018) 1430–1444.
[10] B. Brecht, D. Therriault, A. Weimerskirch, W. Whyte, V. Kumar, T. Hehn, R. Goudy, A security credential management system for v2x communications, IEEE Trans. Intell. Transp. Syst. 19 (99) (2018) 3850–3871, https://doi.org/10.1109/TITS.2018.2797529.
[11] M. Khodaei, H. Noroozi, P. Papadimitratos, Scaling pseudonymous authentication for large mobile systems, in: Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks, WiSec ‘19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 174–184, https://doi.org/10.1145/3317549.3323410.
[12] H. Birge-Lee, Y. Sun, A. Edmundson, J. Rexford, P. Mittal, Bamboozling certificate authorities with bgp, in: Proceedings of the 27th USENIX Conference on Security Symposium, SEC'18, USENIX Association, USA, 2018, pp. 833–849.
[13] M. Al-Bassam, S. Meiklejohn, Contour: a practical system for binary transparency, in: J. Garcia-Alfaro, J. Herrera-Joancomartí, G. Livraga, R. Rios (Eds.), Data Privacy Management, Cryptocurrencies and Blockchain Technology, Springer International Publishing, Cham, 2018, pp. 94–110.
[14] B. Laurie, A. Langley, E. Kasper, Certificate Transparency, RFC 6962, RFC Editor, June 2013.
[15] B. Laurie, A. Langley, E. Kasper, Certificate transparency, Commun. ACM 57 (10) (2014) 40–46, https://doi.org/10.1145/2659897.
[16] T.H.-J. Kim, L.-S. Huang, A. Perrig, C. Jackson, V. Gligor, Accountable key infrastructure (aki): a proposal for a public-key validation infrastructure, in: Proceedings of the 22nd International Conference on World Wide Web, WWW ‘13, ACM, Association for Computing Machinery, New York, NY, USA, 2013, pp. 679–690, https://doi.org/10.1145/2488388.2488448.
[17] D. Basin, C. Cremers, T.H.-J. Kim, A. Perrig, R. Sasse, P. Szalachowski, Design, analysis, and implementation of arpki: an attack-resilient public-key infrastructure, IEEE Trans. Dependable Secure Comput. 15 (3) (2016) 393–408, https://doi.org/10.1109/TDSC.2016.2601610.
[18] J. Yu, V. Cheval, M. Ryan, Dtki: a new formalized pki with verifiable trusted parties, Comput. J. 59 (11) (2016) 1695–1713, https://doi.org/10.1093/comjnl/bxw039.
[19] S. Khan, Z. Zhang, L. Zhu, M. Li, K. Safi, Q. Gul, X. Chen, Accountable and transparent tls certificate management: an alternate public-key infrastructure with verifiable trusted parties, Secur. Commun. Netw. 2018 (2018) 1–16, https://doi.org/10.1155/2018/8527010.
[20] L.S. Huang, A. Rice, E. Ellingsen, C. Jackson, Analyzing forged ssl certificates in the wild, in: 2014 IEEE Symposium on Security and Privacy, IEEE, 2014, pp. 83–97.
[21] M. Cui, Z. Cao, G. Xiong, How is the forged certificates in the wild: practice on large-scale ssl usage measurement and analysis, in: Y. Shi, H. Fu, Y. Tian, V.V. Krzhizhanovskaya, M.H. Lees, J. Dongarra, P.M.A. Sloot (Eds.), Computational Science – ICCS 2018, Springer, Springer International Publishing, Cham, 2018, pp. 654–667.
[22] X. Zheng, L. Pan, H. Chen, P. Wang, Investigating security vulnerabilities in modern vehicle systems, in: L. Batten, G. Li (Eds.), International Conference on Applications and Techniques in Information Security, Springer, Springer Singapore, Singapore, 2016, pp. 29–40.
[23] Z. El-Rewini, K. Sadatsharan, D.F. Selvaraj, S.J. Plathottam, P. Ranganathan, Cybersecurity challenges in vehicular communications, Veh. Commun. 23 (2020) 100214, https://doi.org/10.1016/j.vehcom.2019.100214, http://www.sciencedirect.com/science/article/pii/S221420961930261X.
[24] L.C. Hua, M.H. Anisi, P.L. Yee, M. Alam, Social networking-based cooperation mechanisms in vehicular ad-hoc network—a survey, Veh. Commun. 10 (2017) 57–73, https://doi.org/10.1016/j.vehcom.2017.11.001, http://www.sciencedirect.com/science/article/pii/S2214209617300992.
[25] S. Sharma, B. Kaushik, A survey on internet of vehicles: applications, security issues & solutions, Veh. Commun. 20 (2019) 100182, https://doi.org/10.1016/j.vehcom.2019.100182, http://www.sciencedirect.com/science/article/pii/S2214209619302293.
[26] S. Meier, B. Schmidt, C. Cremers, D. Basin, The tamarin prover for the symbolic analysis of security protocols, in: Proceedings of the 25th International Conference on Computer Aided Verification - Volume 8044, CAV 2013, Springer-Verlag, Berlin, Heidelberg, 2013, pp. 696–701.
[27] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, J. Hubaux, Secure vehicular communication systems: design and architecture, IEEE Commun. Mag. 46 (11) (2008) 100–109, https://doi.org/10.1109/MCOM.2008.4689252.
[28] T. Leinmüller, L. Buttyan, J.-P. Hubaux, F. Kargl, R. Kroh, P. Papadimitratos, M. Raya, E. Schoch, Sevecom-secure vehicle communication, Tech. rep., Sep. 2006.
[29] W. Whyte, A. Weimerskirch, V. Kumar, T. Hehn, A security credential management system for v2v communications, in: 2013 IEEE Vehicular Networking Conference, IEEE, 2013, pp. 1–8, https://doi.org/10.1109/VNC.2013.6737583.
[30] P. Consortium, et al., Preparing secure vehicle-to-x communication systems - preserve, http://www.preserve-project.eu. (Accessed 20 February 2019).
[31] Car-to-car communication consortium (c2c-cc), http://www.car-2-car.org. (Accessed 20 February 2019).
[32] Ieee standard for wireless access in vehicular environments - security services for application and management massages, accessed: 25/2/2019 (3/2016).
[33] E. T. S. Institute, Intelligent transport systems (its); vehicular communications; basic set of applications; definitions, ETSI Tech. TR-102-638.
[34] I.T.S. Committee, et al., Ieee trial-use standard for wireless access in vehicular environments-security services for applications and management messages, IEEE Vehicular Technol. Soc. Standard 1609 (2006) 01, https://doi.org/10.1109/IEEESTD.2006.243731.
[35] P.-Y. Shen, V. Liu, M. Tang, B. Caelli, An efficient public key management system: an application in vehicular ad hoc networks, in: P.B. Seddon, S. Gregor (Eds.), Proceedings of the 15th Pacific Asia Conference on Information Systems (PACIS), Queensland University of Technology/AIS Electronic Library (AISeL), Australia, 2011, pp. 1–15, https://eprints.qut.edu.au/43638/.
[36] H. Tan, M. Ma, H. Labiod, A. Boudguiga, J. Zhang, P.H.J. Chong, A secure and authenticated key management protocol (sa-kmp) for vehicular networks, IEEE Trans. Veh. Technol. 65 (12) (2016) 9570–9584, https://doi.org/10.1109/TVT.2016.2621354.
[37] N. Alexiou, M. Laganà, S. Gisdakis, M. Khodaei, P. Papadimitratos Vespa, Vehicular security and privacy-preserving architecture, in: Proceedings of the 2nd ACM Workshop on Hot Topics on Wireless Network Security and Privacy, HotWiSec ‘13, Association for Computing Machinery, New York, NY, USA, 2013, pp. 19–24, https://doi.org/10.1145/2463183.2463189.
[38] N. Bißmeyer, J. Petit, K.M. Bayarou, Copra: conditional pseudonym resolution algorithm in vanets, in: 2013 10th Annual Conference on Wireless on-Demand Network Systems and Services (WONS), 2013, pp. 9–16, https://doi.org/10.1109/WONS.2013.6578314.
[39] Vehicle safety communications security studies: technical design of the security credential management system, https://www.regulations.gov/document?D=NHTSA-2015-0060-0004, 7/2016. (Accessed 25 March 2019).
[40] M. Khodaei, P. Papadimitratos, Evaluating on-demand pseudonym acquisition policies in vehicular communication systems, in: Proceedings of the First International Workshop on Internet of Vehicles and Vehicles of Internet, IoV-VoI ‘16, Association for Computing Machinery, New York, NY, USA, 2016, pp. 7–12, https://doi.org/10.1145/2938681.2938684.
[41] M. Khodaei, H. Jin, P. Papadimitratos, Towards deploying a scalable robust vehicular identity and credential management infrastructure, in: 2014 IEEE Vehicular Networking Conference (VNC), IEEE, 2014, pp. 33–40, https://doi.org/10.1109/VNC.2014.7013306.

[42] U.s. department of transportation. safety pilot model deployment, https://safetypilot.umtri.umich.edu. (Accessed 25 May 2018).
[43] U. s. department of transportation-national highway traffic safety administration. u. s. dot federal motor vehicle safety standards; v2v comunications, https://www.federalregister.gov/documents/2017/01/12/201631059/federal-motor-vehicle-safety-standards-v2v-communications. (Accessed 29 May 2018).
[44] V. Kumar, J. Petit, W. Whyte, Binary hash tree based certificate access management for connected vehicles, in: Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks, WiSec '17, Association for Computing Machinery, New York, NY, USA, 2017, pp. 145–155, https://doi.org/10.1145/3098243.3098257.
[45] M.A.S. Junior, E.L. Cominetti, H.K. Patil, J. Ricardini, L. Ferraz, M.V. Silva, Privacy-preserving method for temporarily linking/revoking pseudonym certificates in vanets, in: 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), IEEE, 2018, pp. 1322–1329, https://doi.org/10.1109/TrustCom/BigDataSE.2018.00182.
[46] M.A. Simplicio Jr, E.L. Cominetti, H.K. Patil, J.E. Ricardini, M.V.M. Silva, Acpc: efficient revocation of pseudonym certificates using activation codes, Ad Hoc Netw. 90 (2019) 101708, https://doi.org/10.1016/j.adhoc.2018.07.007, http://www.sciencedirect.com/science/article/pii/S1570870518304761.
[47] R.C. Merkle, A certified digital signature, in: G. Brassard (Ed.), Advances in Cryptology — CRYPTO' 89 Proceedings, Springer, New York, New York, NY, 1990, pp. 218–238.
[48] B. Dowling, F. Günther, U. Herath, D. Stebila, Secure logging schemes and certificate transparency, in: I. Askoxylakis, S. Ioannidis, C. Katsikas, Sokratisand Meadows (Eds.), Computer Security – ESORICS 2016, Springer, Springer International Publishing, Cham, 2016, pp. 140–158.
[49] L. Nguyen, Accumulators from bilinear pairings and applications, in: A. Menezes (Ed.), Topics in Cryptology – CT-RSA 2005, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 275–292.
[50] I. Damgård, N. Triandopoulos, Supporting non-membership proofs with bilinear-map accumulators, IACR Cryptol. ePrint Archiv. 2008 (2008) 538.
[51] C. Papamanthou, R. Tamassia, N. Triandopoulos, Authenticated hash tables, in: Proceedings of the 15th ACM Conference on Computer and Communications Security, CCS '08, Association for Computing Machinery, New York, NY, USA, 2008, pp. 437–448, https://doi.org/10.1145/1455770.1455826.
[52] C. Papamanthou, R. Tamassia, N. Triandopoulos, Optimal verification of operations on dynamic sets, in: P. Rogaway (Ed.), Advances in Cryptology – CRYPTO 2011, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 91–110.
[53] S. Nakamoto, Bitcoin: a peer-to-peer electronic cash system.
[54] M.A. Simplicio, E.L. Cominetti, H.K. Patil, J.E. Ricardini, M.V.M. Silva, The unified butterfly effect: efficient security credential management system for vehicular communications, in: 2018 IEEE Vehicular Networking Conference (VNC), IEEE, 2018, pp. 1–8, https://doi.org/10.1109/VNC.2018.8628369.
[55] D. Dolev, A. Yao, On the security of public key protocols, IEEE Trans. Inf. Theory 29 (2) (2006) 198–208, https://doi.org/10.1109/TIT.1983.1056650.
[56] E. Syta, I. Tamas, D. Visher, D.I. Wolinsky, P. Jovanovic, L. Gasser, N. Gailly, I. Khoffi, B. Ford, Keeping authorities" honest or bust" with decentralized witness cosigning, in: Security and Privacy (SP), 2016 IEEE Symposium on, Ieee, 2016, pp. 526–545, https://doi.org/10.1109/SP.2016.38.
[57] A. Tomescu, V. Bhupatiraju, D. Papadopoulos, C. Papamanthou, N. Triandopoulos, S. Devadas, Transparency logs via append-only authenticated dictionaries, in: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS '19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 1299–1316, https://doi.org/10.1145/3319535.3345652.
[58] D. Krajzewicz, J. Erdmann, M. Behrisch, L. Bieker, Recent development and applications of SUMO - simulation of urban MObility, Int. J. Adv. Syst. Meas. 5 (3&4) (2012) 128–138, http://elib.dlr.de/80483/.
[59] R. Lu, X. Lin, H. Zhu, P.-H. Ho, X. Shen, Ecpp: efficient conditional privacy preservation protocol for secure vehicular communications, in: IEEE INFOCOM 2008-the 27th Conference on Computer Communications, IEEE, 2008, pp. 1229–1237, https://doi.org/10.1109/INFOCOM.2008.179.
[60] S. International, Dedicated short range communications (dsrc) message set dictionary, Tech. rep., DSRC Technical Committee, 2016.
[61] S. Banani, S. Gordon, S. Thiemjarus, S. Kittipiyakul, Verifying safety messages using relative-time and zone priority in vehicular ad hoc networks, Sensors 18 (4) (2018) 1195, https://doi.org/10.3390/s18041195.
[62] O. Chakroun, S. Cherkaoui, Overhead-free congestion control and data dissemination for 802.11p vanets, Veh. Commun. 1 (3) (2014) 123–133, https://doi.org/10.1016/j.vehcom.2014.05.003, http://www.sciencedirect.com/science/article/pii/S2214209614000205.
[63] J. Rezgui, S. Cherkaoui, O. Chakroun, Deterministic access for dsrc/802.11p vehicular safety communication, in: 2011 7th International Wireless Communications and Mobile Computing Conference, 2011, pp. 595–600.
[64] B.S. Gukhool, S. Cherkaoui, Ieee 802.11p modeling in ns-2, in: 2008 33rd IEEE Conference on Local Computer Networks (LCN), 2008, pp. 622–626.
[65] S.M. Farooq, S. Hussain, S. Kiran, T.S. Ustun, Certificate based security mechanisms in vehicular ad-hoc networks based on iec 61850 and ieee wave standards, Electronics 8 (1) (2019) 96.
[66] M.Y. Kubilay, M.S. Kiraz, H.A. Mantar, Certledger: a new pki model with certificate transparency based on blockchain, Comput. Secur. 85 (2019) 333–352, https://doi.org/10.1016/j.cose.2019.05.013, http://www.sciencedirect.com/science/article/pii/S0167404818313014.