A privacy-preserving method for temporarily linking/revoking pseudonym certificates in vehicular networks

Marcos A. Simplicio Jr.[1], Eduardo Lopes Cominetti[1], Harsh Kupwade Patil[2], Jefferson E. Ricardini[1], Leonardo T. D. Ferraz[1] and Marcos Vinicius M. Silva[1]

[1] Escola Politécnica, Universidade de São Paulo, Brazil.
{mjunior,ecominetti,joliveira,lferraz,mvsilva}@larc.usp.br
[2] LG Electronics, USA. harsh.patil@lge.com

Abstract. Vehicular communication (V2X) technologies are expected to become increasingly common in the future. Although they enable improvements on transportation safety and efficiency, the large scale deployment of V2X requires addressing some challenges. In particular, to prevent abuse by drivers and by the system itself, V2X architectures must: (1) ensure the authenticity of messages, which is usually accomplished by means of digital certification; and (2) preserve the privacy of honest users, so owners of non-revoked certificates cannot be easily identified and tracked by eavesdroppers. A promising design to address these requirements is the Security Credential Management System (SCMS), which is currently among the main candidates for protecting V2X communications in the United States. Even though SCMS provides efficient, scalable and privacy-preserving mechanisms for managing V2X-oriented certificates, in this article we show that its certificate revocation process can be further enhanced. Namely, we present two birthday attacks against SCMS's revocation process, both of which degrade the system's security as time passes and more certificates are revoked. We then describe an alternative design to prevent such security degradation with minimal computational overhead. In complement to these security gains, we also describe a mechanism for improving the flexibility of the revocation procedure, allowing certificates (as well as their owner's privacy) to be temporarily revoked in an efficient manner. This should be useful, for example, to implement suspension mechanisms or to aid in investigations by law-enforcement authorities.

Keywords: Vehicular communications, security, Security Credential Management System (SCMS), pseudonym certificates, revocable privacy, linkability

1 Introduction

The past decade has witnessed a surge in digital technologies embedded in physical objects, leading to what is today known as Internet of Things (IoT) [1]. This trend has also reached the automotive industry, which has shown a growing interest in exploring interaction models such as Vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I) and Vehicle-to-Pedestrian (V2P), collectively referred to as Vehicle-to-Everything (V2X) communications [2, 3]. After all, V2X enables several applications aimed at improving transportation safety, efficiency, and human to machine interaction [4]. For example, information classified as Basic Safety Messages (BSMs) – which include velocity, direction and brake status – can help drivers keep a safe distance from other vehicles while maintaining a suitable speed. Actually,

onboard systems can evaluate such information and, if an accident appears to be imminent, the vehicle itself can provide (semi-)automatic responses to prevent it in a timely manner. These prospects are among the motivations behind a recent publication by the United States Department of Transportation (USDOT) [5] mandating that vehicles should be capable of exchanging BSMs with each other. This can be accomplished via well-established communication technologies such as Wireless Access in a Vehicular Environment (WAVE), designed to be used within the Dedicated Short Range Communications (DSRC) spectrum [6], as well as by recent standardization efforts such as Long Term Evolution (LTE)-based V2X [3].

Albeit promising, the large scale deployment of V2X technologies also requires addressing some challenges, especially security and privacy concerns [7, 8]. More precisely, V2X architectures are expected to (1) ensure that messages exchanged between vehicles are legitimate, banning misbehaving users, while (2) preserving the anonymity of honest users, so their movements cannot be easily tracked by other vehicles or by the system itself.

One common approach for fulfilling these requirements is to create a Vehicular Public-Key Infrastructure (VPKI) [8, 9]. In this case, the system's security is provided by issuing digital certificates to vehicles, so they can sign messages sent. Such certificates can, however, be revoked if the system detects some misbehavior, such as the transmission of invalid messages signed with it (e.g., due to a malfunction or for malicious purposes). Vehicles are then expected to verify the authenticity of received messages, acting upon them only if they were signed by a non-revoked peer.

Ensuring the vehicles' privacy, in turn, requires the vehicles' certificates to be devoid of any information that identifies their owners. Otherwise, vehicles can be tracked by eavesdroppers monitoring when and where messages signed by the target are broadcast. One promising approach to alleviate such issues involves the usage of pseudonym certificates [10, 11], in which random-like strings play the role of identifiers. By signing different messages with distinct pseudonym certificates, those messages cannot be linked to the same vehicle, thus preserving the sender's privacy. Pseudonym certificates are usually short lived (e.g., valid for one week), which contributes to their owner's privacy and also facilitates revocation. The usage of traditional, long-term credentials is then reserved for situations in which a vehicle must be identified, such as proving that it is authorized to obtain new pseudonym certificates. The frequent renewal of pseudonym certificates should be avoided, though, because vehicles are not expected to constantly have access to a reliable network connection. Hence, vehicles are usually provisioned with batches of pseudonym certificates covering current and also future time periods, thus enabling the vehicle to operate for a long time (e.g., years) [12, 13].

Among the many pseudonym-based security solutions for V2X (see [10] for a survey), one of the most prominent is the Security Credential Management System (SCMS) [12, 14]. Actually, it is currently considered to be one of the leading candidate designs for protecting vehicular communications in the United States [14]. In SCMS, a Registration Authority (RA) creates batches of public keys from a single request, in the so-called butterfly key expansion process. The RA then shuffles keys belonging to different users before individually sending them to the Pseudonym Certificate Authority (PCA). The PCA, in turn, creates valid certificates for those keys and encrypts them before delivering the results to the requesting vehicle. The system is designed in such a manner that, unless RA and PCA collude, they are unable to link a pseudonym certificate to its owner, nor to learn whether or not two pseudonym certificates belong to the same vehicle. Specifically, the PCA does not identify the owner of each vehicle's certificate, whereas the RA, who delivers certificates to a vehicle, does not learn their contents. In case of abuse, however, the privacy of the misbehaving vehicle is annulled and all of its pseudonym certificates are revoked by placing a small piece of information in a Certificate Revocation List (CRL).

Motivated by this growing practical interest in SCMS, some recent studies unveiled

opportunities for enhancing its design, proposing novel or alternative procedures for tackling identified issues [13, 15]. In this article, we contribute to this research effort by presenting improvements to SCMS's certificate revocation and linkage approach, enhancing its security and flexibility. Specifically, we show that SCMS's revocation procedure is prone to attacks built upon the birthday paradox to degrade the system's security over time. We then propose an alternative method that addresses this issue with minimal overhead. In addition, while the original SCMS focused basically on the permanent revocation of devices, we describe an efficient method that enables vehicle revocation and linkage for a limited time. Such capability is useful, for example, when vehicles need to be temporarily suspended, or when aiding in investigations by law enforcement authorities.

The rest of this article is organized as follows. Section 2 lists the main symbols and notation employed in the document. Section 3 discusses related works, giving a broad view of the state-of-the-art on V2X security. Section 4 describes with some detail the SCMS protocol. Section 5 presents some shortcomings of SCMS's key linkage and revocation process, which motivates the improvements introduced in Section 6. The security and performance of our proposal is then analyzed, respectively, in Sections 7 and 8. Finally, Section 9 concludes the discussion and suggests some ideas for future works.

2 General notation

For convenience, we list in Table 1 the symbols and general notation adopted in this article.

When cryptographic algorithms are mentioned, we assume standardized schemes are adopted, providing a security level of at least 128 bits. In particular, the following algorithms are considered safe for use in modern systems: symmetric encryption (i.e., using secret keys) is performed with the AES block cipher [16] whereas asymmetric encryption (i.e., using public/private key pairs) is done with ECIES [17]; the hashing algorithm may be either SHA-2 [18] or SHA-3 [19]; and digital signatures are generated and verified with ECDSA [20] or EdDSA [21, 22].

3 Related works

Following the emergence of V2X, many proposals have appeared in the literature aiming to fulfill its security and privacy requirements. Notably, approaches based on pseudonym certificates seem to be promising, and are being considered in standardization efforts [23, 24]. Existing schemes rely on a variety of security primitives, including asymmetric, identity-based, and symmetric cryptography, as well as on group signatures (for a survey, see [10]). To give an overview of the state-of-the-art in the area, in what follows we discuss some recent works, focusing on how they handle a vehicle's revocation.

The pseudonym scheme with user-controlled anonymity (PUCA) [25] was designed to ensure that the end users' privacy is preserved even when multiple system entities collude. PUCA assumes that each vehicle is equipped with a trusted module, which manages long-term enrollment certificates, cryptographic keys and pseudonyms. This module is also responsible for handling revocation: if a vehicle misbehaves, the system broadcasts an order of self-revocation (OSR) to the corresponding pseudonym; the trusted module responsible for that pseudonym is then expected to halt its operation. With this approach, the users' privacy is preserved even in case of misbehavior, since there is no need to identify the real users behind revoked pseudonyms. One potential issue, however, is that malicious users might somehow discard OSR messages addressed to them or tamper with the trusted module, thus avoiding revocation. In addition, PUCA was designed to prevent pseudonym certificates from being linked together, arguing that traditional investigation methods should be employed in case of misconduct, without any aid from the V2X system. This

Table 1: General notation and symbols

Symbol	Meaning
G	The generator point for an elliptic curve group $\mathbb{G}$
sig	A digital signature
$cert$	A digital certificate
U,$\mathcal{U}$	Public signature keys (stylized $\mathcal{U}$: reserved for PCA)
u,u	Private keys corresponding to U and $\mathcal{U}$ (respectively)
S	Public caterpillar key for signature
s	Private caterpillar key for signature
E	Public caterpillar key for encryption
e	Private caterpillar key for encryption
$\hat{S}$	Public cocoon key for signature
$\hat{s}$	Private cocoon key for signature
$\hat{E}$	Public cocoon key for encryption
$\hat{e}$	Private cocoon key for encryption
β	Number of cocoon keys in a batch of certificates
la_id	ID of a Linkage Authority (LA)
ℓ	Number of LAs (typically two)
`ls`	A linkage seed
`lh`	A linkage hook
`plv`	A pre-linkage value
`lv`	A linkage value
τ	Number of time periods covered by batch of certificates
σ	Number of certificates valid in each time period
$Enc(\mathcal{K}, str)$	Encryption of bitstring str with key $\mathcal{K}$
$Hash(str)$	Hash of bitstring str
$str_1 \parallel str_2$	Concatenation of bitstrings str_1 and str_2

might be reasonable if V2X itself did not lead to new threats, but unfortunately that is not the case: after all, malicious users can abuse the V2X capabilities to deliberately cause accidents or facilitate robbery (e.g., by reporting an accident ahead, attackers could induce target vehicles to suddenly break or take a detour). Therefore, it is sensible for the system to include mechanisms that handle issues it creates, which is why revocable privacy is a common requirement in other V2X solutions (e.g., [12]).

As an alternative to relying on self-revocation, some protocols use Bloom filters to reduce the size of CRLs [26, 27]. A Bloom filter is a probabilistic data structure that has a compression gain over the size of the list, at the cost of false-positives (but no false-negatives) on the pertinence test. Although potentially useful for reducing CRL sizes when a many certificates need to be revoked, this approach is likely to be inefficient when applied to the revocation of a few vehicles. This happens because the filter's size depends on the maximum number of certificate revocations it supports, so the CRL maintains its full size despite the actual number of revoked certificates. Given the large number of pseudonym certificates per vehicle, this technique is potentially more efficient than placing in the CRL every single certificate belonging to a revoked vehicle. However, Bloom filters

are likely less efficient than the approach adopted in solutions such as SCMS, in which all certificates from a single vehicle can be revoked altogether with a single CRL entry.

Another interesting solution aimed at certificate revocation in the V2X scenario is the Issue First Activate Later (IFAL) scheme [28]. Basically, IFAL provides mechanisms based on "activation codes", bitstrings that are required for the computation of any pseudonym certificate's private key. As long as a revoked vehicle does not receive the activation codes for pseudonym certificates obtained prior to revocation, it is prevented from using those certificates even if they are still valid. Non-revoked vehicles can then periodically request the activation codes for their own certificates, whereas requests from revoked vehicles are ignored. This results in a reduction of the CRLs' sizes, since entries corresponding to that vehicle (or to its pseudonyms) do not need to remain in the CRL. A similar-purpose solution, the Binary Hash Tree based Certificate Access Management (BCAM) [13], achieves an equivalent reduction on the size of CRLs while introducing a more efficient process for distributing activation codes. Namely, activation codes can be computed from a small piece of information broadcast by a Certificate Access Manager (CAM), so vehicles are not required to explicitly request them. Since IFAL and BCAM aim to reduce the number of identifiers placed in CRLs, without affecting how these identifiers are constructed or organized, they can actually be seen as complementary to the mechanisms hereby presented.

Finally, the Security Credential Management System (SCMS), originally proposed in [12] and later extended in [14], is one of the main proposals in the literature for dealing with revocable privacy while preventing non-colluding system entities from tracking devices. These security properties are expected to hold in the so-called "honest-but-curious" threat model: even though the system's entities follow the correct protocols when issuing and revoking pseudonym certificates, they might engage in passive attacks, trying to use the information acquired during the protocols' execution to their advantage (e.g., to track vehicles) [8]. Since SCMS is one of the leading candidates for protecting V2X security in the US [12, 14], and it is also used as the basis for our proposal, we analyze its design more closely in Section 4.

4 The Security Credential Management System (SCMS)

In this section, we describe SCMS in more detail. Along the discussion, we focus on the description given in [12] rather than in [14]. This is motivated by two main reasons: (1) the notation in the former is more concise; and (2) the modifications introduced in the latter have no effect on our high-level description, nor influence the attacks and improvements hereby discussed. Nonetheless, for completeness, we briefly highlight the differences between [12] and [14] whenever pertinent.

In SCMS, each vehicle receives two types of certificates: an enrollment certificate, which has a long expiration time (e.g., years) and identifies legitimate devices; and multiple pseudonym certificates, each having a short validity (e.g., a few days), so $\sigma \geqslant 1$ certificates of this type are valid simultaneously. For privacy reasons, vehicles are expected to frequently change the pseudonym certificate employed for signing messages, thus avoiding tracking by eavesdroppers. However, the value of σ should be limited by the system to avoid "sybil-like" attacks [29], in which one vehicle pretends to be a platoon by signing multiple messages with different pseudonyms [30, 31]. For example, if traffic lights give a higher priority to congested roads, such a fake platoon might receive preferential treatment even when driving on lightly loaded roads.

SCMS's design is such that batches of pseudonym certificates can be efficiently distributed to vehicles, as well as revoked in case of misbehavior by their owners. The basic architecture that gives support to these capabilities involves the following entities (see Figure 1 for a complete architecture and [12] for the description of all of its elements):

- Pseudonym Certificate Authority (PCA): issues pseudonym certificates to vehicles, upon request by RAs.
- Registration Authority (RA): handles the vehicles' requests for batches of pseudonym certificates. Such requests are signed with the requesting vehicle's enrollment certificate, so they can be validated by the RA. Each legitimate request by a vehicle generates multiple individual requests to the PCA, and requests associated to different vehicles are shuffled together so the PCA cannot correlate (groups of) requests to a same owner.
- Linkage Authority (LA): generates pseudorandom bitstrings that are inserted into pseudonym certificates to enable their efficient revocation. By revealing the pseudorandomness seeds, multiple certificates issued to a same vehicle can be identified from a single entry placed on a certificate revocation list (CRL). SCMS uses two LAs, even though its architecture is flexible enough to support additional LAs.
- Misbehavior Authority (MA): monitors the system aiming to identify misbehaving vehicles, handling their revocation whenever necessary.

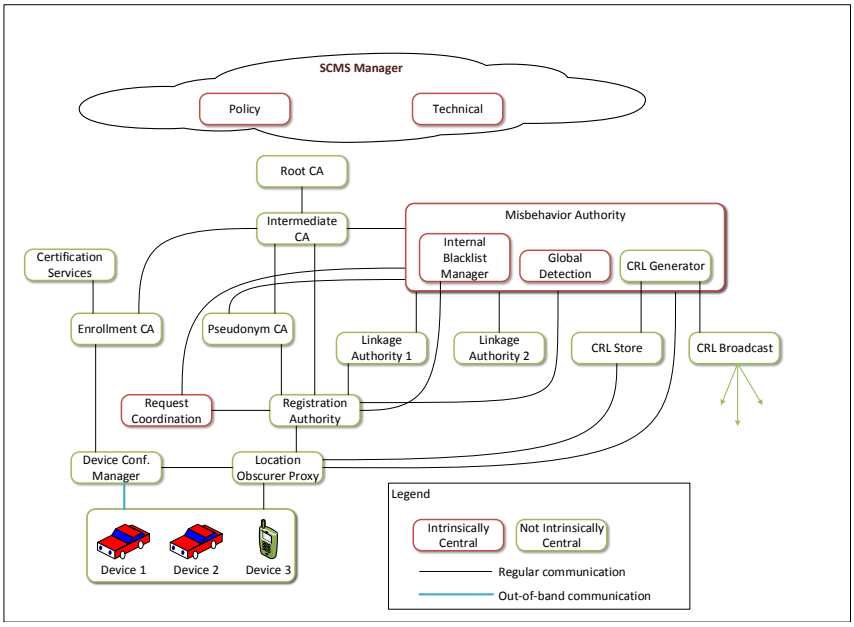

Figure 1: Overview of SCMS's architecture. Source:[12].

These entities are involved in the two main procedures provided by SCMS: the "butterfly key expansion", by means of which pseudonym certificates are issued to vehicles; and "key linkage", which allows the revocation of those certificates in case of misbehavior. Both procedures are detailed in the following subsections.

4.1 Butterfly key expansion

SCMS's butterfly key expansion process allows vehicles to obtain arbitrarily large batches of (short-lived) pseudonym certificates by means of a single, small-sized request. It involves the following steps, which are illustrated in Figure 2. First, the vehicle generates two pairs of *caterpillar* private/public keys, $(s, S = s \cdot G)$ and $(e, E = e \cdot G)$, for randomly picked s and e. In addition, the vehicle also instantiates two pseudorandom functions (PRF), f_s

and f_e; there are small differences on how this instantiation is done in [12] and in [14], but we hereby omit the details because they have no impact on the attacks hereby presented. Finally, the vehicle then sends f_s and f_e to the Registration Authority (RA), together with the public caterpillar keys S and E.

The RA, in turn, uses S and f_s to generate β public *cocoon* signature keys $\hat{S}_i = S + f_s(i) \cdot G$, where $0 \leqslant i < \beta$ for an arbitrary value of β. Similarly, E and f_e are employed in the computation of β public cocoon encryption keys $\hat{E}_i = E + f_e(i) \cdot G$. Pairs of cocoon keys $(\hat{S}_i, \hat{E}_i)$ corresponding to different vehicles are then shuffled together by the RA before being individually sent to the Pseudonym Certificate Authority (PCA).

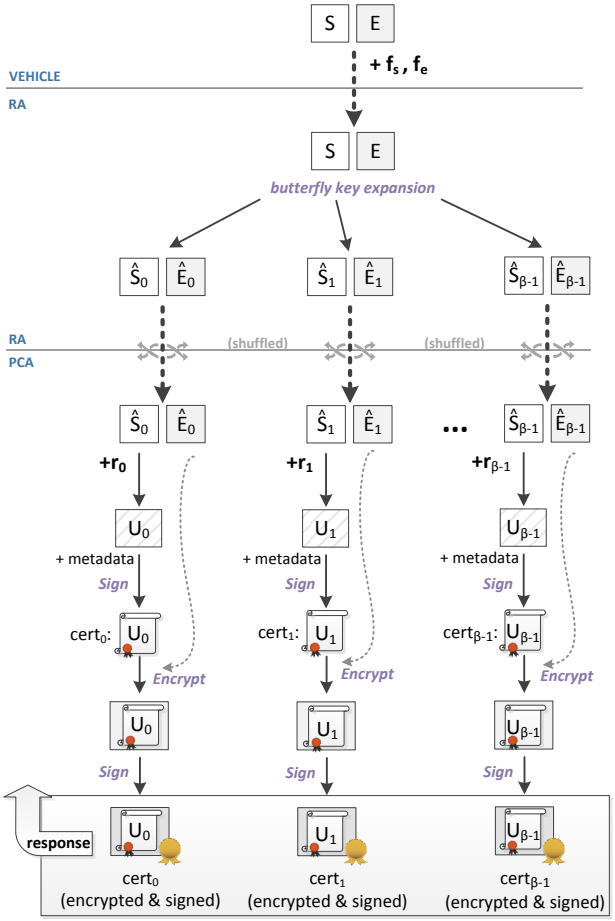

Figure 2: SCMS's butterfly key expansion and certificate generation.

Upon reception of a $(\hat{S}_i, \hat{E}_i)$ pair from the RA, the PCA can create either explicit or implicit certificates [32]. In the case of explicit certification, the vehicle's public signature key is computed as $\mathrm{U}_i = \hat{S}_i + r_i \cdot G$, for a random value r_i. The resulting U_i is then inserted into a certificate $cert_i$ together with the required metadata `meta` (e.g., the corresponding validity period and linkage values, as described later in Section 4.2). Finally, the PCA digitally signs $cert_i = (\mathrm{U}_i, \texttt{meta})$ with its own private key u, uses $\hat{E}_i$ to encrypt both the signed certificate and the value of r_i, and once again signs the result before relaying it to the RA. As a result, only the requesting vehicle can decrypt the PCA's response with the private decryption key $e + f_e(i)$. By doing so, the vehicle learns the public signature key U_i, and then computes its corresponding private signature key $\mathrm{u}_i = s + r_i + f_s(i)$.

For implicitly certified keys, this process is slightly different. First, the PCA computes a credential $V_i = \hat{S}_i + r_i \cdot G$, again for a random r_i. The PCA then creates the implicit certificate $cert_i = (V_i, \texttt{meta})$, computes its hash $h_i = Hash(cert_i)$, and signs it to obtain $sig_i = h_i \cdot r_i + u$. The pair $(cert_i, sig_i)$, after being encrypted with $\hat{E}_i$ and signed with u, is sent back to the vehicle via the RA. Finally, the vehicle decrypts the PCA's response to recover $(cert_i, sig_i)$ and computes $h_i = Hash(cert_i)$. The vehicle then sets its own private signature key as $\mathrm{u}_i = h_i \cdot (s + f_s(i)) + sig_i$, whereas the corresponding public signature key is computed as $\mathrm{U}_i = \mathrm{u}_i \cdot G$. The validity of the (U_i, V_i) pair can then be implicitly verified by any vehicle, simply by checking that $\mathrm{U}_i = h_i \cdot V_i + \mathcal{U}$, where $\mathcal{U}$ is the PCA's public signature key.

Independently of the type of certificate adopted, this process preserves the vehicles' privacy as long as there is no collusion between RA and PCA. After all, by shuffling the public cocoon keys from different vehicles together, the RA prevents the PCA from linking groups of keys to a same device. Unlinkability of public keys towards the RA, in turn, is ensured because RAs never learn the value of U_i or $cert_i$ from the PCA's encrypted response. In addition, by signing its response, the PCA prevents the RA from performing a Man-in-the-Middle (MitM) attack. More precisely, without this signature, the RA could perform a MitM attack as follows:

1. Instead of sending $\hat{E}_i$ to the PCA, the RA provides a forged cocoon encryption key $\hat{E}_i^* = z \cdot G$, for an arbitrarily chosen z;
2. Upon reception of the PCA's response, the RA can decrypt it using the z decryption key, thus learning the value of $cert_i$;
3. The RA then re-encrypts the PCA's response using the correct $\hat{E}_i$, relaying the result to the vehicle;
4. The vehicle, unaware of the attack, decrypts the received response as usual, computing a public/private signature key pair and the corresponding pseudonym-based certificate $cert_i$;
5. Whenever the vehicle presents $cert_i$ in a V2X communication, the RA can link that $cert_i$ to the original request, thus identifying the corresponding vehicle via its enrollment certificate.

The extra signature made by the PCA prevents such MitM attack attempt, though, because the RA would not be able to provide a valid signature for the re-encrypted package generated in the attack's step 3.

4.2 Key linkage

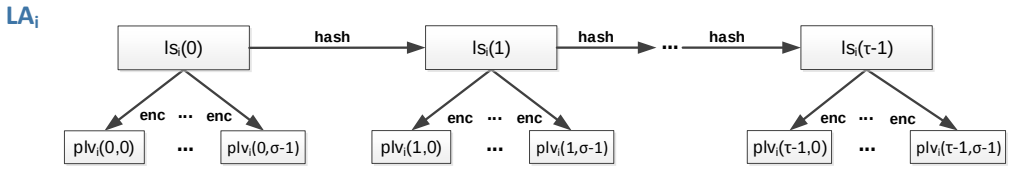

Figure 3: SCMS's key linkage tree: LA_i generates the linkage seeds (`ls`) and pre-linkage values (`plv`) employed for certificate revocation/linkage.

The revocation process in SCMS was designed to avoid creating large certificate revocation lists (CRLs), in which many pseudonym certificates from the same vehicle would have to be listed. Namely, by placing a small amount of information placed in a

CRL, SCMS allows multiple certificates created from a same caterpillar key can be linked together when the corresponding vehicle is revoked. This is accomplished by means of linkage values $\texttt{lv}$, which are included as part of the pseudonym certificates' metadata, as described in what follows

Suppose the RA needs to create a batch of pseudonym certificates covering τ time periods, with σ pseudonym certificates valid per period, so the batch size is $\beta = \tau \cdot \sigma$. In this case, the RA chooses $\ell \geqslant 2$ Linkage Authorities (LAs), and request from each of them β pre-linkage values $\texttt{plv}_i(t, c)$, where $0 \leqslant t < \tau$ and $0 \leqslant c < \sigma$.

In response to the RA's request, LA_i picks a random 128-bit linkage seed $\texttt{ls}_i(0)$. Then, it iteratively computes a τ-long hash chain [33] of the form $\texttt{ls}_i(t) = \mathit{Hash}(la\_id_i \,\|\, \texttt{ls}_i(t-1))$, where la_id_i is LA_i's identifier and $1 \leqslant t < \tau$. Each linkage seed $\texttt{ls}_i(t)$ obtained in this manner is then employed in the computation of σ pre-linkage values $\texttt{plv}_i(t, c) = \mathit{Enc}(\texttt{ls}_i(t), la\_id_i \,\|\, c)$. In [14], the encryption operation uses the Davies-Meyer construction [34], so the cipher's input is XORed with its output to create a one-way function; however, since this modification is not relevant for our discussion, we hereby omit the extra XOR. Finally, every $\texttt{plv}_i(t, c)$ is truncated to a suitable length, individually encrypted and authenticated[1] using a key shared between PCA and LA_i, and sent to the RA.

After receiving the LA's response, the RA simply includes this (encrypted) information in the pseudonym certificate request sent to the PCA, so $\texttt{plv}_i(t, c)$ (for $1 \leqslant i \leqslant \ell$) accompanies the c-th public cocoon keys corresponding to time period t. The PCA, after decrypting the pre-linkage values and verifying their authenticity, computes the linkage value $\texttt{lv}(t, c)$ by XORing those pre-linkage values together. In the usual case, which consists of two LAs, the linkage value for the c-th certificate and time period t is then computed as $\texttt{lv}(t, c) = \texttt{plv}_1(t, c) \oplus \texttt{plv}_2(t, c)$.

As a result of this process, whenever the Misbehavior Authority (MA) notices that a given pseudonym certificate was involved in some malicious event, certificates from the same owner can be revoked altogether. This requires the collaboration of the PCA, RA, and LAs, as follows. First, the PCA associates the $\texttt{lv}$ informed by the MA to the original pseudonym certificate request from the RA. The PCA then sends this information, as well as the corresponding pre-linkage values $\texttt{plv}_i(t, c)$, to the RA. Subsequently, the RA does the following: (1) identifies the vehicle behind the original request, placing its enrollment certificate in a blacklist to prevent it from obtaining new pseudonym certificates; and (2) asks LA_i to identify the linkage seed $\texttt{ls}_i(0)$ from which $\texttt{plv}_i(t, c)$ was computed. Each LA_i responds with $\texttt{ls}_i(t_s)$, where the time period t_s corresponds to the moment when the revocation should start being valid. For example, t_s might be the current time period, or the time period when the misbehavior was first detected. The linkage seeds $\texttt{ls}_i(t_s)$ provided by the LAs are then placed in a public CRL, allowing any entity to compute $\texttt{lv}(t, c)$ for time periods $t \geqslant t_s$, identifying which certificates correspond to this CRL entry. Consequently, this mechanism provides forward privacy: the misbehaving vehicle's certificates for *current* and *future* time periods are revoked, and messages signed with them can be traced back to that device; messages signed in *past* time periods cannot be linked, though, preserving the device's privacy prior to the detection of the malicious activity.

In terms of complexity, this revocation process is such that, if the system involves ℓ LAs, each revoked device contributes with ℓ pre-linkage values to the CRL. Hence, the CRL size grows linearly with the number of revoked devices, not with the number of revoked certificates. Actually, if SCMS is combined with activation codes as suggested in BCAM [13], the size of CRLs is expected to becomes even smaller. This reduction is useful not only for saving bandwidth, but also because the larger the number of entries in a CRL, the higher the processing overheads for checking a certificate's revocation status.

[1] Even though authentication and freshness are not explicitly mentioned in [12, 14], these properties are important to prevent a dishonest RA from delivering to the PCA forged or reused pre-linkage values without contacting LA_i; otherwise, the RA might craft values of $\texttt{plv}$ to gain the ability of tracking vehicles.

More precisely, for each CRL entry published at time period t_s, the verification of whether it covers a given certificate involves basically the computation of two components:

a) $\texttt{ls}_i(t_c)$: it takes $\ell \cdot (t_c - t_s)$ hashes to compute $\texttt{ls}_i(t_c)$ from $\texttt{ls}_i(t_s)$, where $1 \leqslant i \leqslant \ell$ and t_c is the time period when the verification is performed. This cost may be reduced by means of pre-computation, i.e., if the device always keeps the updated version of the linkage seeds, $\texttt{ls}_i(t_c)$, besides the original ones provided in the CRL. Nevertheless, to cope with the lack of a system-wide time synchronization [28], devices may actually need to keep a slightly older linkage seed in memory; for example, by keeping $\texttt{ls}_i(t_c - \epsilon)$ for a small ϵ, it is possible to compute $\texttt{ls}_i(t_c)$ with only ϵ hashes.

b) $\texttt{plv}_i(t_c, c)$: it takes ℓ encryptions to compute $\texttt{plv}_i(t_c, c)$ from $\texttt{ls}_i(t_c)$ if the value of c for the certificate under verification is known; this is the case, for example, if the value of c is part of that certificate's metadata. Otherwise, the total cost would be up to $\ell \cdot \sigma$ encryptions, since the certificate under analysis may be any out of σ that are valid in the current time period; with enough memory, however, the latency of this process can be reduced via the pre-computation of a look-up table with all σ possible entries for each $\texttt{ls}_i(t_c)$ in the CRL. On the one hand, besides providing better performance, the first approach facilitates the construction of solutions resilient to the aforementioned sybil-like attacks; this can be accomplished by counting as valid only messages signed with certificates for a fixed value of c. On the other hand, this ability may also be abused to allow vehicle tracking if one or several applications decide to only accept a specific c; meanwhile, a bit of privacy is lost because different certificates known to have the same value for σ are also deemed to belong to different vehicles. Therefore, mandating the disclosure of c in pseudonym certificates is likely to become controversial and, in practice, it would probably be avoided in favor of look-up tables.

5 Two shortcomings of SCMS's key linkage process

SCMS allows basically the permanent revocation of users, via the disclosure of the linkage seed for the current time period. This can be seen in Figure 3, which shows a graphical representation of the dependencies between linkage seeds and pre-linkage values in the linkage tree: as indicated by the directed arrows, the disclosure of linkage value $\texttt{ls}_i(t_s)$ allows anyone to compute the pre-linkage values associated to it, $\texttt{plv}_i(t_s, \cdot)$, as well as linkage values for subsequent time periods and their corresponding pre-linkage values.

Even though permanent revocation is indeed a critical use case for key linkage in V2X communications, in some situations it is also important that a small set of messages exchanged among vehicles can be traced to their origin. For example, when handling traffic accidents, the messages sent by the vehicles involved in the event may be useful for law enforcement authorities, so they can understand its causes (and, possibly, identify culprits) [30]. Similarly, a hijacked car might have its privacy temporarily suspended. This would allow nearby vehicles and roadside units to identify all messages sent from it to a same user and, thus, track its movement. In the original SCMS, this could be accomplished if the identifiers for all certificates belonging to that time period are placed in a CRL. For improved performance, however, it would be interesting to enable this temporary tracking by means of a single CRL-like entry, similarly to what happens with permanent revocations.

A second issue with SCMS's key linkage and revocation procedure is that it is prone to attacks built upon the birthday paradox to degrade the system's security over time, allowing the recovery of linkage seeds that have not been placed in CRLs. This issue appears both during the computation of linkage seeds and of pre-linkage values derived from them, as explained in the following sub-sections.

5.1 Birthday attacks against pre-linkage values

In SCMS, multiple pre-linkage values are computed via the encryption of a same plaintext, under different k-bit long keys. Namely, for all users, the c-th pre-linkage value valid on a given time period t, $\texttt{plv}_i(t, c)$ is computed by LA_i as $Enc(\texttt{ls}_i(t), la\_id_i \,\|\, c)$, using the linkage seed $\texttt{ls}_i(t)$ as encryption key.

This procedure allows the construction of a key recovery attack typical of a multi-key setting [35, 36], as follows. First, the attacker picks 2^n distinct keys $\texttt{ls}_i^j$, where $0 \leqslant j < 2^n$. Then, the attacker performs 2^n encryptions to build a table of the form $(\texttt{plv}_i^j, \texttt{ls}_i^j)$, where $\texttt{plv}_i^j = Enc(\texttt{ls}_i^j, la\_id_i \,\|\, c)$ for a target la_id_i and a fixed $0 \leqslant c < \sigma$. According to the birthday paradox, if the attacker can gather a total of 2^m pre-linkage values computed by LA_i for a same index c, at least one of those pre-linkage values will match a $\texttt{plv}_i^j$ in the attacker's table with a high probability as long as $m + n \geqslant k$ [35]. Except in the very unlikely case of equivalent keys, whenever there is a match for $(\texttt{plv}_i^j, \texttt{ls}_i^j)$ it is safe to assume that $\texttt{ls}_i^j$ corresponds to the linkage seed employed for the computation of that pre-linkage value.

Since the 2^m pre-linkage values employed in the attack can refer to different vehicles and time periods, as long as they receive the same index in that time period and come from a same LA_i, the security of the system for a given choice of k degrades as time passes and LA_i serves more devices. To give some concrete numbers, suppose that 1 million pre-linkage values equally indexed by a given LA are disclosed in one year of the system's operation, meaning that $m \approx 20$. That would be the case, for example, if 20,000 vehicles are revoked in the first week of the year, so the corresponding pre-linkage values for the subsequent ≈ 50 weeks become available. In that case, even if that LA adopts a modern security level of $k = 128$ bits, in the span of one year the system's security against such collision attacks drops to $k - m = 108$ bits. Albeit still high enough to prevent most practical attacks, this security level is below the current recommendation of having at least 112 bits of security [37]. In practice, this may end up limiting the lifespan of LAs for a given security threshold, in particular because the recovery of a given $\texttt{ls}_i(t_s)$ allows the computation of any subsequent $\texttt{ls}_i(t)$ for $t > t_s$. Therefore, the effects of such key-recovery could be quite serious to the affected vehicles' privacy if they ever occur.

Fortunately, SCMS is not completely defenseless against this attack, for at least two reasons. The first is that, by design, only the PCA has access to the raw pre-linkage values not included in CRLs, whereas the device's certificates contain only linkage values (i.e., the XOR of two or more pre-linkage values). Hence, even though the PCA is able perform the aforementioned attacks, external entities are in principle prevented from doing so. The second is that the cipher's output is actually truncated for the computation of $\texttt{plv}_i(t, c)$ (namely, in [12], the authors suggest using the 8 most significant bytes of AES). This should produce many partial matches on the attacker's table, leading to multiple candidates for the correct linkage seed $\texttt{ls}_i(t)$. Nevertheless, these candidates could still be filtered with a certain probability if the attacker has access to additional pre-linkage values related to the same linkage seed. For example, from $\texttt{plv}_i(t, c)$ and $\texttt{plv}_i(t + 1, c)$, the attacker obtains, respectively, one set of candidates for $\texttt{ls}_i(t)$ and one for $\texttt{ls}_i(t + 1)$; incorrect candidates can then be filtered out if they do not satisfy $\texttt{ls}_i(t+1) = Hash(la\_id_i \,\|\, \texttt{ls}_i(t))$. Alternatively, if the pre-linkage values obtained are $\texttt{plv}_i(t, c)$ and $\texttt{plv}_i(t, c')$, with $c' \neq c$, two tables can be built: one of the form $(Enc(\texttt{ls}_i^j, la\_id_i \,\|\, c), \texttt{ls}_i^j)$ and the other of the form $(Enc(\texttt{ls}_i^j, la\_id_i \,\|\, c'), \texttt{ls}_i^j)$, for the same group of 2^n keys $\texttt{ls}_i^j$; each table will lead to a different set of candidates for $\texttt{ls}_i(t)$, and candidates not appearing in both sets can be eliminated.

5.2 Birthday attacks against linkage seeds

A second attack that can be perpetrated against SCMS, aimed specifically at its forward privacy property, relies on the fact that the k-bit long linkage seeds are computed via iterative hashing, using a fixed prefix for each LA, i.e., $\texttt{ls}_i(t) = \mathit{Hash}(la\_id_i \,\|\, \texttt{ls}_i(t-1))$. More precisely, to discover $\texttt{ls}_i(t < t_s)$ from a given $\texttt{ls}_i(t_s)$ placed in a CRL, an attacker can proceed as follows. First, the attacker picks a random $\texttt{ls}_i^0$ and then uses it as the anchor for a hash chain of the form $\texttt{ls}_i^j = \mathit{Hash}(la\_id_i \,\|\, \texttt{ls}_i^{j-1})$, until 2^n hashes are performed for the target la_id_i. For simplicity, we assume that no collision occurs during this process, i.e., that $\texttt{ls}_i^j \neq \texttt{ls}_i^{j'}$ for all $j \neq j'$; nevertheless, this simplification comes without loss of generality because, whenever there is a collision, the attacker could simply save the current chain, pick a new anchor distinct from any previously computed $\texttt{ls}_i^j$, and then start a new chain from it.

Once again due to the birthday paradox, an attacker that gathers 2^m linkage seeds computed by LA_i has a high probability to find a match between at least one of those linkage seeds and some previously computed $\texttt{ls}_i^j$ if $m+n \geqslant k$. If a match occurs for $\texttt{ls}_i(t_s)$ and $\texttt{ls}_i^j$, then a previous linkage seed $\texttt{ls}_i(t_s - \epsilon)$ will also match $\texttt{ls}_i^{j-\epsilon}$. Assuming $\texttt{ls}_i^{j-\epsilon}$ is actually the pre-image of $\texttt{ls}_i(t_s - \epsilon + 1)$, and not a second pre-image, this would allow the attacker to associate non-revoked certificates to a same device and, thus, violate the system's forward privacy.

This attack can be performed both by internal and external entities, using pre-computed hash-chains for selected LAs; after all, it requires only access to linkage seeds from (public) CRLs and to the LAs' identifiers. Since the 2^m linkage seeds employed for this purpose can refer to any device and time period, once again the system's security degrades as time passes and certificates from devices served by a same LA_i are included in CRLs. Therefore, for security reasons, the lifespan of a given LA_i may become limited by the choice of k and by the number of devices revoked with the participation of LA_i.

6 A more flexible and secure revocation/linkage process

In this section, we present two independent improvements to SCMS, addressing the issues identified in Section 5. Since the proposed solutions are independent (i.e., they can be adopted altogether or as stand-alone improvements), we describe them separately in what follows.

6.1 Enabling temporary linkability and revocation: linkage hooks

Aiming to enable the temporary revocation of vehicles, while still maintaining SCMS's overall approach, we propose a slight modification to the construction of linkage trees. In the proposed scheme, illustrated in Figure 4, we create one extra link to the linkage tree. This is accomplished with the insertion of a *linkage hook* $\texttt{lh}_i(t)$ between any linkage seed $\texttt{ls}_i(t)$ and its corresponding pre-linkage values $\texttt{plv}_i(t, \cdot)$. As a result, the disclosure of $\texttt{lh}_i(t_s)$ allows the recovery of every $\texttt{plv}_i(t_s, \cdot)$, but not of any $\texttt{plv}_i(t, \cdot)$ for $t \neq t_s$. Hence, this simple modification is enough to grant the system the ability to link and/or revoke all certificates from a given time period.

As a side note, it is worth mentioning that the same concept can be further extended to address other use cases, such as a scenario in which only a part of the certificates from a given time period need to be linked/revoked. For example, suppose that the system's certificates must have υ different purposes, so they display distinct "key usage" fields (like in regular X.509 certificates [38]) even though they share the same validity period. This feature might be the useful, e.g., for protecting the identity of official vehicles: whenever they do not need (or want) to be identified, they could use their regular pseudonym

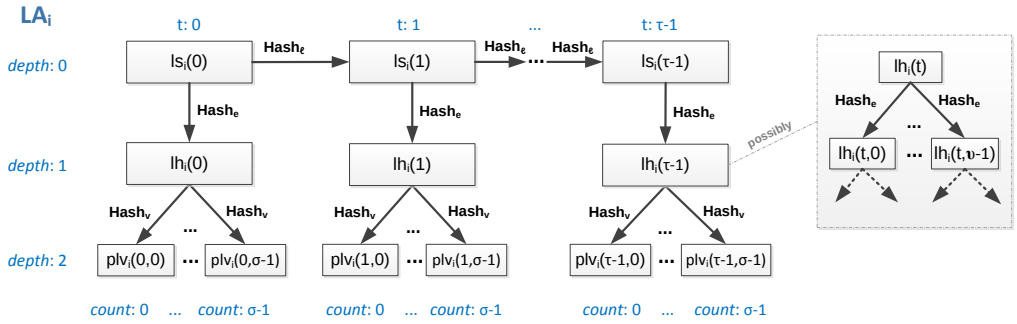

Figure 4: Proposed certificate linkage tree: adding linkage hooks and security strings.

certificates, identical to those issued to other vehicles; however, when they need to prove their status as official vehicles (e.g., aiming to get traffic priority), they would sign their messages with special-purpose certificates.

In this scenario, one possible approach for allowing the independent revocation/linkage of such different-purpose certificates is to create distinct linkage trees, one for each key usage. Then, the certificates sharing the same purpose could be revoked altogether as usual, by inserting ℓ linkage hooks (for temporary revocation) or seeds (for a permanent revocation) in a CRL. However, revoking all certificates belonging to a vehicle would lead to υ times more data placed in CRLs. Conversely, a more efficient revocation can be obtained by adding one extra level to the linkage tree, with $\mathtt{lh}_i(t)$ linking the multiple $\mathtt{lh}_i(t,0) \ldots \mathtt{lh}_i(t,\upsilon-1)$. As a result, if all certificates from a given time period need to be linked/revoked, then $\mathtt{lh}_i(t)$ would be disclosed as usual; if, however, only certificates of a certain type needed to be linked/revoked, then the disclosure of the corresponding linkage hook would suffice.

6.2 Protecting linkage trees against birthday attacks: security strings

Whereas the structural change resulting from linkage hooks creates a more flexible key linkage/revocation process, it does not address the security issues identified in Sections 5.1 and 5.2. Therefore, for better security, we also propose a slightly different process for the derivation of pre-linkage values. Namely, instead of using a block cipher, the computation of linkage seeds, linkage hooks and pre-linkage values relies simply on hash functions whose inputs include a "security string", i.e., a different suffix for each hash function invocation. As originally discussed by Leighton and Micali [39] in the context of hash-based signatures [40], such security strings limit the attackers' ability to use the birthday paradox in their favor, effectively thwarting the aforementioned birthday attacks.

When applying the improvement from Section 6.1, the security string I can be built taking into account the tree-like structure shown in Figure 4. Hence, each node receives a different value of I based on the linkage tree's identifier and on its location inside that tree. For this purpose, the fields shown in Table 2 can be employed, leading to $I = la\_id \,\|\, tree\_id \,\|\, t \,\|\, count \,\|\, depth$.

Using this security string, the linkage seeds, linkage hooks and pre-linkage values are computed by LA_i as follows:

- Linkage seeds: $\mathtt{ls}_i(0)$ is picked at random, and

$$\mathtt{ls}_i(t) = Hash(\mathtt{ls}_i(t-1) \,\|\, la\_id \,\|\, tree\_id \,\|\, t-1 \,\|\, 0 \,\|\, 0)$$

Table 2: Components of the security strings.

Field	Suggested length (bits)	Description
depth	8	Node's depth in tree (all linkage seeds are at depth 0, as shown in Figure 4)
count	8	Node's index in the time period and depth (starting at 0, as shown in Figure 4)
t	24	Time period to which the node is associated
tree_id	56	Tree identifier (unique per tree from a same LA)
la_id	32	LA's identifier

- Linkage hooks: they are all computed as

$$\texttt{lh}_i(t) = \mathit{Hash}(\texttt{ls}_i(t) \,\|\, \mathit{la\_id} \,\|\, \mathit{tree\_id} \,\|\, t \,\|\, 0 \,\|\, 1)$$

- Pre-linkage values: they are all computed as

$$\texttt{plv}_i(t, c) = \mathit{Hash}(\texttt{lh}_i(t) \,\|\, \mathit{la\_id} \,\|\, \mathit{tree\_id} \,\|\, t \,\|\, c \,\|\, 2)$$

Finally, we emphasize that this approach is generic enough to accommodate further changes to the linkage tree's structure, simply by modifying the composition of the security strings. For example, supopose that more intermediate linkage hooks are added to give support to additional key usages, as discussed in Section 6.1. In this case, the security string's *count* parameter would simply be adjusted according with the linkage hook's position in the tree.

7 Security against birthday attacks

Against the approach hereby proposed, the birthday attacks described in Sections 5.1 and 5.2 would proceed as follows. First, the attacker builds a large table containing entries of the form $(h^j = \mathit{Hash}(\mathit{str}^j, I), \mathit{str}^j)$ for a fixed security string I and for arbitrary k-long bit strings str^j, where $0 \leqslant j < 2^n$ for some n. Then, if some k-long linkage seed $\texttt{ls}_i(t)$ matches h^j, the attacker is able to recover the corresponding pre-image str^j, which should match $\texttt{ls}_i(t-1)$ with high probability. A similar reasoning applies if the match occurs for a linkage hook $\texttt{lh}_i(t)$ or for a pre-linkage value $\texttt{plv}_i(t, c)$, whose pre-images reveal $\texttt{ls}_i(t)$ and $\texttt{lh}_i(t)$, respectively.

Like before, the birthday paradox dictates that finding at least one match with high probability requires 2^m tests for $m + n \geqslant k$. Since I is used only once in the system, however, the attacker can only perform one test per I, meaning that the attack would work in practice only for $n \approx k$. If k is chosen appropriately (e.g., $k = 128$), the construction of such table with 2^k entries becomes computationally unfeasible. In addition, if the value of *tree_id* is unpredictable by attackers, they would not be able to pre-compute (parts of) such look-up table before one node in the corresponding tree is revoked. In this case, *tree_id* provides additional security similarly to what is done by salts in the context of password hashing [41].

Finally, it is interesting to note that SCMS does use something similar to a security string in its design. Namely, it includes *la_id* in the derivation process employed for creating pre-linkage values and linkage seeds. This avoids security issues that might arise from collisions among data produced by different LAs. Therefore, the approach hereby proposed can be seen as an extension of the SCMS design, improving even further its resilience against birthday attacks.

8 Performance Considerations

The flexibility introduced by linkage hooks incurs only a small overhead when compared to the original SCMS scheme. Namely, in any time period, a single additional hash function call is required for verifying whether a device's certificates was permanently revoked: the one that allows the computation of the corresponding linkage hook. Checking whether a certificate was temporarily revoked with a security hook, on the other hand, takes as much effort as verifying if it was (permanently) revoked in SCMS.

The addition of a security string I to the computation of the linkage tree's nodes, in turn, has little impact on processing as long as the hash function's input fits its block size. For example, SHA-256 operates on 512-bit blocks, appending at least 65 bits to its input message (a bit '1' for padding, and a 64-bit length indicator) [18]; therefore, a single call to its underlying compression function is enough to process a 128-bit linkage seed, linkage hook or pre-linkage value even when it is combined with a 319-bit security string. Nevertheless, since parts of the security string may need to be published on the CRLs together with the corresponding linkage seeds or linkage hooks, its length should be limited to avoid unnecessary communication overheads. More precisely, the *depth* and *count* fields correspond to counters, so they do not need to be explicitly included in the CRL: it suffices to indicate whether the entry refers to a linkage seed or linkage hook to determine whether $depth = 0$ or $depth = 1$, respectively, and in both cases *count* starts at zero. The LA's identifier *la_id* and the time period t are likely to be explicitly shown, although multiple CRL entries for which these fields are identical may be grouped together under the same data structure, thus avoiding the need of repeating them. Finally, the *tree_id* field must be unique for each CRL entry from a same LA, so it should always appear explicitly. Such considerations are what motivated the parameter lengths suggested in Table 2, which result in 128-bit security strings.

9 Conclusion

The broad adoption of intelligent transportation systems (ITS) requires attention to two important aspects: data authentication, so invalid messages can be filtered out by vehicles; and (revocable) user privacy, so honest drivers cannot be tracked by their peers or by the system itself. A promising design to address these requirements is the Security Credential Management System (SCMS), which provides efficient, scalable and privacy-preserving mechanisms for issuing and revoking pseudonym certificates. For these characteristics, SCMS is currently part of standardization efforts [14], and has also been target of recent studies aiming to improve its design [13, 15].

In this article, our contribution on this research effort is twofold. First, we propose modifications to the structure of SCMS's key linkage tree aiming to address additional use cases. In particular, the proposed approach efficiently supports the temporary revocation and linkage of pseudonym certificates, which should be useful to implement suspension mechanisms or aid in investigations by law-enforcement authorities. Second, we describe two birthday attacks against SCMS's key linkage/revocation process, both of which able to affect the vehicles' privacy as time passes and more certificates are revoked. One way to tackle this issue would be to limit the lifespan of the corresponding Linkage Authorities (LAs), thus preventing the security level from dropping below a given threshold. Instead, the proposed approach based on security strings solves this issue by averting this security degradation altogether.

The two proposed enhancements are independent, whereas their combination yields a revocation process with improved security and flexibility when compared to the state-of-the-art. Moreover, this added robustness comes with minimal performance impacts, both in terms of processing and bandwidth usage.

As future work we plan to further investigate possible improvements to SCMS's key linkage process. In special, it would be interesting to create a method by which the index t of the different pseudonym certificates valid on a same time period could be identified after they are revoked, while concealing its value otherwise. This would facilitate the process of verifying whether or not a given certificate is revoked, since then it would be unnecessary to check a given certificate against every possible pre-linkage value derived from linkage seeds/hooks placed on a CRL. Furthermore, as mentioned in Section 4.2, SCMS's linkage process is such that PCAs must verify the authenticity and freshness of received pre-linkage values. Otherwise, a dishonest RA might gain the ability to track vehicles by performing a quite simple attack: it pretends to be a vehicle, creating cocoon keys whose private key is known, and obtains the corresponding (encrypted) pre-linkage values from LAs, as well as the corresponding certificates and their linkage-values from the PCA; then, by reusing the same (encrypted) pre-linkage values on an actual request, the resulting linkage-values that are enclosed in those certificates will be exactly those that were previously learned. The PCA can discard replayed pre-linkage values via nonce verification, for example, but this standard approach involves memory overheads to store previously-used nonces. Therefore, providing more efficient alternatives remains as an interesting topic for future work.

Acknowledgment

This work was supported by the Brazilian National Council for Scientific and Technological Development (CNPq) under grant 301198/2017-9, and by LG Electronics via the Foundation for the Technological Development of the Engineering Sciences (FDTE).

References

[1] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *IEEE Communications Surveys Tutorials*, vol. 16, no. 1, pp. 414–454, 2014.

[2] J. Harding, G. Powell, R. Yoon, J. Fikentscher, C. Doyle, D. Sade, M. Lukuc, J. Simons, and J. Wang, "Vehicle-to-vehicle communications: Readiness of V2V technology for application," National Highway Traffic Safety Administration, Washington, DC, USA, Tech. Rep. DOT HS 812 014, 2014.

[3] S. Chen, J. Hu, Y. Shi, Y. Peng, J. Fang, R. Zhao, and L. Zhao, "Vehicle-to-everything (v2x) services supported by LTE-based systems and 5G," *IEEE Communications Standards Magazine*, vol. 1, no. 2, pp. 70–76, 2017.

[4] P. Papadimitratos, A. L. Fortelle, K. Evenssen, R. Brignolo, and S. Cosenza, "Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation," *IEEE Communications Magazine*, vol. 47, no. 11, pp. 84–95, November 2009.

[5] NHTSA, "Federal Motor Vehicle Safety Standards; V2V Communication," National Highway Traffic Safety Administration, U.S. Department of Transportation (USDOT), Tech. Rep. 8, Jan 2017. [Online]. Available: https://www.federalregister.gov/documents/2017/01/12/2016-31059/federal-motor-vehicle-safety-standards-v2v-communications

[6] D. Jiang and L. Delgrossi, "IEEE 802.11p: Towards an international standard for wireless access in vehicular environments," in *IEEE Vehicular Technology Conference (VTC Spring)*, 2008, pp. 2036–2040.

[7] F. Schaub, Z. Ma, and F. Kargl, "Privacy requirements in vehicular communication systems," in *Proc. of the International Conference on Computational Science and Engineering*, vol. 3. IEEE, 2009, pp. 139–145.

[8] M. Khodaei and P. Papadimitratos, "The key to intelligent transportation: Identity and credential management in vehicular communication systems," *IEEE Vehicular Technology Magazine*, vol. 10, no. 4, pp. 63–69, Dec 2015.

[9] P. Cincilla, O. Hicham, and B. Charles, "Vehicular PKI scalability-consistency tradeoffs in large scale distributed scenarios," in *IEEE Vehicular Networking Conference (VNC)*, Dec 2016, pp. 1–8.

[10] J. Petit, F. Schaub, M. Feiri, and F. Kargl, "Pseudonym schemes in vehicular networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 228–255, 2015.

[11] M. Khodaei, H. Jin, and P. Papadimitratos, "Towards deploying a scalable & robust vehicular identity and credential management infrastructure," in *2014 IEEE Vehicular Networking Conference (VNC)*, Dec 2014, pp. 33–40.

[12] W. Whyte, A. Weimerskirch, V. Kumar, and T. Hehn, "A security credential management system for V2V communications," in *IEEE Vehicular Networking Conference*, 2013, pp. 1–8.

[13] V. Kumar, J. Petit, and W. Whyte, "Binary hash tree based certificate access management for connected vehicles," in *Proc. of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, ser. WiSec'17. New York, NY, USA: ACM, 2017, pp. 145–155.

[14] CAMP, "Security credential management system proof–of–concept implementation – EE requirements and specifications supporting SCMS software release 1.1," Vehicle Safety Communications Consortium, Tech. Rep., may 2016. [Online]. Available: https://www.its.dot.gov/pilots/pdf/SCMS_POC_EE_Requirements.pdf

[15] M. Simplicio, E. Cominetti, H. K. Patil, J. Ricardini, and M. Silva, "The unified butterfly effect: Efficient security credential management system for vehicular communications," IACR eprint archive: https://eprint.iacr.org/2018/089.pdf, 2018.

[16] NIST, *Federal Information Processing Standard (FIPS 197) – Advanced Encryption Standard (AES)*, National Institute of Standards and Technology, U.S. Department of Commerce, National Institute of Standards and Technology, U.S. Department of Commerce. Gaithersburg, MD, USA, November 2001. [Online]. Available: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[17] IEEE, *IEEE Standard Specifications for Public-Key Cryptography – Amendment 1: Additional Techniques*, IEEE Computer Society, 2004.

[18] NIST, *Federal Information Processing Standard (FIPS 180-4) – Secure Hash Standard (SHS)*, National Institute of Standards and Technology, U.S. Department of Commerce, National Institute of Standards and Technology, U.S. Department of Commerce (NIST). Gaithersburg, MD, USA, August 2015, dOI:10.6028/NIST.FIPS.180-4.

[19] NIST, *Federal Information Processing Standard (FIPS 202) – SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions*, National Institute of Standards and Technology, U.S. Department of Commerce, National Institute of Standards and Technology, U.S. Department of Commerce. Gaithersburg, MD, USA, August 2015, dOI:10.6028/NIST.FIPS.202.

[20] ——, *Federal Information Processing Standard (FIPS 186-4) – Digital Signature Standard (DSS)*, National Institute of Standards and Technology, U.S. Department of Commerce, National Institute of Standards and Technology, U.S. Department of Commerce. Gaithersburg, MD, USA, July 2013. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

[21] D. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," *Journal of Cryptographic Engineering*, vol. 2, no. 2, pp. 77–89, Sep 2012, see also http://ed25519.cr.yp.to/eddsa-20150704.pdf.

[22] S. Josefsson and I. Liusvaara, "RFC 8032 – edwards-curve digital signature algorithm (EdDSA)," https://tools.ietf.org/html/rfc8032, January 2017.

[23] ETSI, "TR 102 941 – intelligent transport systems (ITS); security; trust and privacy management," European Telecommunications Standards Institute, Tech. Rep., Jun 2012.

[24] IEEE, "IEEE standard for wireless access in vehicular environments–security services for applications and management messages - amendment 1," *IEEE Std 1609.2a-2017 (Amendment to IEEE Std 1609.2-2016)*, pp. 1–123, Oct 2017.

[25] D. Förster, F. Kargl, and H. Löhr, "PUCA: A pseudonym scheme with strong privacy guarantees for vehicular ad-hoc networks," *Ad Hoc Networks*, vol. 37, pp. 122–132, 2016, special Issue on Advances in Vehicular Networks.

[26] M. Raya, P. Papadimitratos, I. Aad, D. Jungels, and J.-P. Hubaux, "Eviction of misbehaving and faulty nodes in vehicular networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 8, 2007.

[27] J. J. Haas, Y.-C. Hu, and K. P. Laberteaux, "Design and analysis of a lightweight certificate revocation mechanism for vanet," in *Proceedings of the sixth ACM international workshop on VehiculAr InterNETworking*. ACM, 2009, pp. 89–98.

[28] E. Verheul, "Activate later certificates for V2X – combining ITS efficiency with privacy," Cryptology ePrint Archive, Report 2016/1158, 2016. [Online]. Available: http://eprint.iacr.org/2016/1158

[29] J. Douceur, "The Sybil attack," in *Proc. of 1st International Workshop on Peer-to-Peer Systems (IPTPS)*. Springer, January 2002. [Online]. Available: https://www.microsoft.com/en-us/research/publication/the-sybil-attack/

[30] R. Moalla, B. Lonc, H.Labiod, and N. Simoni, "Risk analysis study of ITS communication architecture," in *3rd International Conference on The Network of the Future*, 2012, pp. 2036–2040.

[31] K. Alheeti, A. Gruebler, and K. McDonald-Maier, "An intrusion detection system against malicious attacks on the communication network of driverless cars," in *12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, Jan 2015, pp. 916–921.

[32] Certicom, "Sec 4 v1.0: Elliptic curve Qu-Vanstone implicit certificate scheme (ECQV)," Certicom Research, Tech. Rep., 2013, http://www.secg.org/sec4-1.0.pdf.

[33] L. Lamport, "Password authentication with insecure communication," *Commun. ACM*, vol. 24, no. 11, pp. 770–772, 1981.

[34] B. Preneel, *Davies–Meyer Hash Function*. Boston, MA: Springer US, 2005, pp. 136–136.

[35] E. Biham, "How to decrypt or even substitute DES-encrypted messages in 2^{28} steps," *Inf. Process. Lett.*, vol. 84, no. 3, pp. 117–124, nov 2002.

[36] N. Mouha and A. Luykx, "Multi-key security: The Even-Mansour construction revisited," in *Advances in Cryptology – CRYPTO 2015: 35th Annual Cryptology Conference*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 209–223.

[37] NIST, *Special Publication 800-131A Rev. 1 – Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*, National Institute of Standards and Technology, U.S. Department of Commerce, Gaithersburg, MD, USA, November 2015, dOI:10.6028/NIST.SP.800-131Ar1.

[38] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "RFC 5280 – Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) profile," RFC 5280 – https://tools.ietf.org/html/rfc5280#section-4.2.1.3, May 2008.

[39] F. Leighton and S. Micali, "Large provably fast and secure digital signature schemes based on secure hash functions," July 11 1995, uS Patent 5,432,852.

[40] D. McGrew, M. Curcio, and S. Fluhrer, "Hash-based signatures," Internet Engineering Task Force, Internet-Draft draft-mcgrew-hash-sigs-06, mar 2017, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/html/draft-mcgrew-hash-sigs-06

[41] E. Andrade, M. Simplicio, P. Barreto, and P. Santos, "Lyra2: efficient password hashing with high security against time-memory trade-offs," *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 3096–3108, 2016, See also: http://eprint.iacr.org/2015/136.