Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

Future generation Security for smart and connected Cars - FuSeCar

Deliverable D5.1: Accountable and secure vehicular communication protocols

WP5: Accountability and transparency for vehicular communications protocols and architectures

Authors:
Mattia Trabucco[2], Luca Ferretti[2], Giovanni Gambigliani Zoccoli[1], Mirco Marchetti[1], and Mauro Andreolini[2]
{name.surname}@unimore.it

[1]Department of Engineering "Enzo Ferrari"
[2]Department of Physics, Informatics and Mathematics
University of Modena and Reggio Emilia

Current revision: R1.0
Delivery date: November 24th, 2025

Revision history

Authors	Changes	Date	Revision
Mattia Trabucco, Giovanni Gambigliani Zoccoli, Luca Ferretti, Mirco Marchetti	Creation of the document, tentative structure, daft of Section 2 and 3	June 9th, 2025	R0.1
Mattia Trabucco, Luca Ferretti	Draft of Section 4 and 5	September 18th, 2025	R0.2
Mattia Trabucco, Luca Ferretti	First draft of complete document, minor fixes	November 24th, 2025	R1.0

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

Contents

1 Introduction

This deliverable is a technical report including the result of Work Package WP5: Accountability and transparency for vehicular communications protocols and architectures. This WP has one main outcome: propose a novel approach to support accountability for Vehicle-to-Everything communications. In this context, accountability ensures that critical actions, such as sending a safety message, can be non-repudiably attributed to a specific entity (e.g., a vehicle or an infrastructure node) by a trusted authority.

Vehicle-to-Everything communications are key enablers for Intelligent Transportation Systems, but their design includes many challenges due to the need of balancing security and privacy with tight latency-requirements of safety-related features, resource constraints of vehicular wireless communication protocols, and economic costs of infrastructures and, in particular, of vehicles. Privacy relates to the peculiar need of guaranteeing conditional anonymity to vehicles, that is, preventing a vehicle from being tracked by malicious parties, only as long as it behaves honestly. As soon as a vehicle behaves maliciously, e.g., by sending false information, it should be possible to efficiently revoke it from the network, thus invalidating its privacy guarantees.

In this document, we propose a novel method that uses *hierarchical deterministic homomorphic key derivation schemes* (HKD) to derive the key pairs used for signing V2X messages that do not require the inclusion of pseudo-random indexing identifiers (e.g., *linkage values* in SCMS [3]) included within the certificates exchanged in each V2X message. The messages exchanged by using this technique are smaller than those of standards based on explicit certificates, addressing one of the main bottlenecks of V2X communications. Network revocation costs are constant with regard to the number of pseudonyms assigned to vehicles and independent from traffic workloads, and no computational overhead is introduced for securing communications when compared to standards based on explicit certificates. By leveraging the properties of HKD schemes, our method allows to derive a large number of pseudonym key pairs from a single master key, thus reducing the storage requirements on both the vehicle and authority side and allowing efficient certificate issuance and revocation. We analyze the costs of our proposal both asymptotically and analytically when instantiated with the NIST p-256 elliptic curve recommended by standards [11, 3] and show that it is more efficient than standards based on explicit certificates in terms of network overhead, while still being affordable in terms of computational costs for key management operations.

2 Background

2.1 State of the art

As completely anonymous authentication schemes are very expensive and not affordable in the context of vehicular networks, the typical design choice is to adopt a PKI-based approach based on pseudonyms [15]. Vehicles are provisioned with a set of certificates issued by a Certificate Authority (CA) which are similar to those used in the Web PKI, as they are bound to the vehicle and the CA public keys (either by explicitly including them, or via implicit mathematical properties), and optional metadata such as the validity period of the certificate itself. However, certificates issued in the Vehicular PKI (VPKI) omit the real identity of the vehicle or other user-identifiable information, and are instead considered pseudonyms of the vehicle. Each vehicle can authenticate its messages with different secret keys, thus a message is supposed to be unlinkable to the real vehicle identity and to other messages authenticated by the same vehicle with a different certificate. This allows a vehicle to maintain its privacy while still being able to authenticate messages within the network. In the major standards for VPKI (i.e., the US Security Credential Management System [3] and the European ETSI ITS standard [11]) each vehicle is provisioned with a set of pseudonym certificates and corresponding key pairs that are valid for a specific time period (e.g., one week). Vehicles select one of the pseudonym certificates from its set to authenticate outgoing messages, and rotates at regular intervals the pseudonym certificate to ensure unlinkability between messages sent over the time period (e.g., every 10 minutes or after sending a certain number of messages) [2].

Revocation of certificates (and, more in general, of cryptographic keys) is quite always complex and possibly expensive [16]. Since pseudonym certificates do not include any vehicle identity information, the revocation of all the certificates associated with a misbehaving vehicle is not straightforward if no other additional mechanisms are implemented. Strawman approaches where the CA broadcasts large Certificate Revocation Lists (CRLs) including all the pseudonym certificates incur in linear costs with regard to the number of pseudonyms, and thus do not scale well. Standards avoid such high costs by using pseudo-random indexing identifiers (e.g., *linkage values* in SCMS [3]) included within the certificates exchanged in each V2X message, to enable authorities to reference the pseudonym certificates associated with a misbehaving vehicle by just releasing a single secret information. While the benefit is the distribution of small CRLs, the drawback is the need to send the indexing value within all V2X messages. Although such a value is currently quite small (9-Bytes in SCMS [3]), it still represents a significant disadvantage for V2X communications where network bandwidth is the major bottleneck, and, due to its dependence on hash functions' collision resistance, resizing may be needed in the future to support higher traffic workloads, in terms of number of vehicles and revocations, and privacy guarantees, in terms of number of pseudonyms assigned to each vehicle.

2.2 Notation

We denote as $\mathbb{G}$ an additive cyclic group of prime order q built over an elliptic curve, where the Elliptic-Curve Discrete Logarithm Problem (ECDLP) is hard with regard to a security parameter λ. Let $\mathcal{O}$ be the neutral element of $\mathbb{G}$ and $B \in \mathbb{G}$ be the generator of $\mathbb{G}$. We slightly abuse notation and adopt $+$ and $\cdot$ both for operations on scalars belonging to $\mathbb{Z}_q$ and to elliptic curve points belonging to $\mathbb{G}$. Thus, $X + Y$ denotes the point addition operation for any $X, Y \in \mathbb{G}$, $s \cdot X$ denotes the point scalar operation for any $s \in \mathbb{Z}_q$ and $X \in \mathbb{G}$, and $a + b$ and $a \cdot b$ denote scalar addition and multiplication modulo q for any two $a, b \in \mathbb{Z}_q$. We assume that all parameters are public and agreed by all participants. For ease of notation, we omit them from inputs of routines and assume them as implicit.

We denote as $[a, b]$ the set of integers $[a, a + 1, \ldots, b]$, and as $[a]$ the set of integers $[1, a]$.

We denote as $\sigma \leftarrow \mathsf{sign}(sk, m)$ the digital signature σ on message m with secret key sk, and as $\{\mathsf{accept}, \mathsf{reject}\} \leftarrow \mathsf{verify}(PK, m, \sigma)$ the routine for verifying a signature σ on message m with public key PK. We observe that

Algorithm 1 BIP32 HKD specification using abstract algebra

```
keygen() :
  sk ←$ Z_q
  PK ← sk · B
  return ⟨sk, PK⟩

pderive(PK, info) :
  PK_info ← PK + KDF(PK, info) · B
  return PK_info

sderive(sk, PK, info) :
  sk_info ← sk + KDF(PK, info)
  return sk_info
```

sign may be probabilistic or deterministic, which may depend on the concrete specification, without affecting the validity of our proposal.

2.3 Hierarchical Deterministic Homomorphic Key Derivation and BIP32

Hierarchical Deterministic Homomorphic Key Derivation (HKD) schemes have recently become popular in the context of blockchains for managing so-called *deterministic wallets*. We consider an abstract version of the most popular scheme defined within BIP32 [21] for non-hardened addresses. An HKD scheme is composed of three routines: *key generation* (keygen), for generating a freshly new key pair which acts as the root of the key hierarchy; *public key derivation* (pderive) and *secret key derivation* (sderive), for computing derived public and secret keys, respectively, with regard to some information $info$, which is a bitstring which acts as the scope of the key derivation, as for more typical symmetric key derivation schemes such as HKDF [13]. In Algorithm 1 we show specifications of the three routines, where $\mathsf{KDF} : \{0,1\}^* \to \mathbb{Z}_q$ denotes a standard key derivation function for symmetric keys [4] (e.g., implemented through HKDF in BIP32). For modeling correctness, we denote as $\{\mathsf{accept}, \mathsf{reject}\} \leftarrow \mathsf{check}(sk', PK')$ a routine which is able to verify whether PK' is a legitimate public key for secret key sk'. Let $\langle sk, PK\rangle \leftarrow\!\!\$\ \mathsf{keygen}()$ be the master key pair: correctness of HKD holds if, given $sk' \leftarrow \mathsf{sderive}(sk, PK, info)$ and $PK' \leftarrow \mathsf{pderive}(PK, info)$, $\mathsf{check}(sk', PK')$ accepts with probability 1. Intuitively, security requires the *unlinkability* of keys, i.e., secret and public keys generated from the same master key pair and different $info$ are computationally indistinguishable from freshly generated keys as long as the master key pair is not leaked. Other security guarantees are implied for unlinkability, among which the most important is the unforgeability of master keys and other derived keys by only knowing some derived key. However, it is important to remark that not all HKD specifications may generate derived keys which are *universally composable* with any other cryptographic scheme, in the sense of not introducing some type of vulnerability when used as if they were freshly generated random keys. In particular, while BIP32 is secure when used with ECDSA digital signatures [7], it may not be secure when straightforwardly combined with other types of cryptographic systems. Note that while other universally composable HKD schemes exist, their design prevent their adoption in the context of efficient communications. Combining subsequent calls to pderive and sderive allows to build a hierarchical tree of key pairs which can be used for managing multiple keys or construct deterministic wallets. The hierarchical approach starts from a root, the master key pair $\langle sk, PK\rangle$. By evaluating sderive and pderive for different $info$ values, one can obtain a number of level-1 derived nodes. Since each derived node can be used as a master key for further derivation, one can build a sub-tree of derived keys by recursively applying sderive and pderive to a derived node. We assume that, for each level i of the hierarchy, there is only one special $info$ value, that we call *master derivation label* (`mdl`), used to derive the child master key pair for the subsequent level $i+1$.

Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

3 System and threat model

We consider a Vehicular credential management system based on a Public Key Infrastructure (VPKI) composed of a Certificate Authority (CA) and multiple vehicles which act as senders and/or receivers within the communication network[1]. Each vehicle communicates within the network by using many *pseudonyms*, and without ever using its own identity information. The privacy of the vehicle is preserved by using different pseudonym key pairs over time, as it is assumed that it is difficult to link two different pseudonyms to the same vehicle. To this aim, a vehicle must obtain a set of *pseudonym certificates* from the CA, which are cryptographic attestations, each binding a pseudonym (and possibly additional metadata) to cryptographic material used for authenticating messages. For a good security and performance trade-off, the CA divides time into time periods such that each pseudonym is valid only within a specific time period, and each vehicle can obtain a maximum fixed number $\mathtt{N}$of pseudonym certificates for each time period. We identify time periods through enumeration using index $t = 0, 1, 2, \ldots$ to denote the t-th time period, and τ the current time period. We assume that CA and vehicles share a synchronized clock[2]. Moreover, the CA may define a parameter $\mathtt{T} \in \mathbb{N}$, such that a vehicle can request sets of pseudonym certificates for up to $(\mathtt{T} - 1)$ future time periods[3]. Thus, a vehicle can obtain a maximum of $\mathtt{N} \cdot \mathtt{T}$ pseudonym certificates.

A CA may revoke all pseudonym certificates associated with a vehicle by distributing *revocation material* (rm), that is received by vehicles to build and maintain *Certificate Revocation Lists* ($\langle crl \rangle$). Note that, in comparison to well-known standard Web PKI where a $\langle crl \rangle$ is directly downloaded from a CA, for network efficiency a vehicle may need some extra processing to build each crl records from each rm and build $\langle crl \rangle$. Also note that a vehicle may need to update its $\langle crl \rangle$ for each new time period, to discard old records or build new ones. The CA owns a key pair $\langle sk_{CA}, PK_{CA} \rangle$, where sk_{CA} is the secret key for authenticating certificates and PK_{CA} is the public key used for verifying authenticity of certificates. Thus, the CA may determine the real identity of a vehicle from its pseudonym certificate even if the vehicle has not been revoked [11, Section 6.2.3 and 6.2.4].

We model the persistent storage of each vehicle by distinguishing four types of data:

- $\langle crt \rangle$, the set of pseudonym certificates that is valid in the current time period;
- $\langle crl \rangle$, the list of all pseudonym certificates that have been revoked by the CA in the current time period;
- crm (*certificate refresh material*), all the due data to refresh and/or prepare $\langle crt \rangle$ for the next time period;
- rrm (*revocation refresh material*), all the due data to refresh and/or prepare $\langle crl \rangle$ for the next time period.

We consider passive and active network adversaries for communications among vehicles, while for ease of presentation we assume trusted communications between vehicles and the CA. We assume that PK_{CA} is known by all vehicles and that the distribution of public keys associated with vehicles depends on protocols specifications.

[1] While the efforts of protecting the privacy of a participant in a vehicular network are typically focused on vehicles, the same principles can be applied to other participants, such as infrastructure nodes or even pedestrians. For simplicity, throughout the document we will refer to vehicles as the only participant of the network that requires pseudonym certificates.

[2] Note that, like standards for vehicular communications, we are assuming a *low-precision* clock synchronization as typical for Web communication networks, since a typical period lasts a week [2].

[3] This happens within SCMS [3], but may not be defined in other standards. In this latter case, our analyses still apply by setting $\mathtt{T} = 1$.

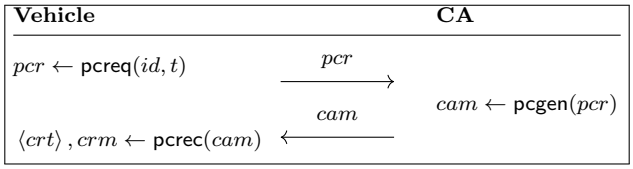

Figure 1: Pseudonym certificates release

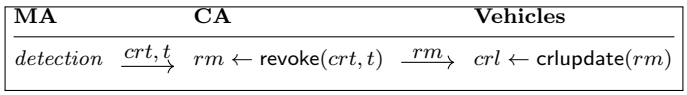

Figure 2: Revocation

4 PPV protocol framework

We propose an abstract protocol framework for describing the management of certificates by a PKI in the context of Pseudonym-based authenticated Vehicular communications, which we denote as PPV. PPV is composed of three sub-protocols: *pseudonym certificate release*, where a vehicle requests and obtains a set of pseudonym certificates from the CA; *revocation*, where a CA receives a notification about a vehicle misbehavior and revokes all the pseudonym certificates associated with the vehicle; *refresh*, where a vehicle updates its certificates and revocation lists for the following time period. Note that, while we only focus on the main features of VPKI management, for which we propose novel contributions, other sub-protocols may be included to cover other features of existing VPKI standards (e.g., the initial registration of vehicles through an Enrollment Authority [11]).

Pseudonym certificates release, shown in Figure 1, is composed of three routines:

- *pseudonym certificates request* (pcreq): the vehicle creates a homonym data structure (that we distinguish from the routine name by denoting it as *pcr*) from the vehicle identifier *id* and time span information Δ_t (i.e., the number of total subsequent time periods, *overall covered time span*, for which the vehicle is requesting pseudonym certificates, including the current time period τ);
- *pseudonym certificates generation* (pcgen): the CA creates a *certificate approval material* (*cam*) comprising data which assess the approval of the input *pcr* by the CA, but which may not include complete pseudonym certificates due to the need for further processing by the vehicle (for security and/or performance reasons);
- *pseudonym certificates reception* (pcrec): the vehicle processes *cam* and produces a set of pseudonym certificates $\langle crt \rangle$, which can be used by the vehicle to authenticate messages within the current time period τ, and certificate refresh material *crm*, which is used during refresh to generate $\langle crt \rangle$ for the following $(\Delta_t - 1)$ time periods;

Revocation, shown in Figure 2, is composed of two routines:

- *revoke* (revoke): the CA creates *revocation material* (*rm*) upon reception of the pseudonym certificate *crt* of a misbehaving vehicle at some time period t (which we show as the output of an abstract *detection* event observed by the Misbehavior Authority, MA);
- *CRL update* (crlupdate): the vehicle processes *rm* to create a *crl* record containing all the pseudonym certificates associated with a misbehaving vehicle.

Note that all *crl* output data obtained from multiple executions of the *Revocation* protocol are appended to the $\langle crl \rangle$ data structure, which is maintained within the persistent storage of the vehicle, as anticipated in Section 3.

Refresh is composed of two routines, both executed by a vehicle:

- *certificate refresh* ($\{\Vdash \langle crt \rangle'\} \leftarrow \mathsf{crtrefresh}(\langle crt \rangle, crm, t)$) updates the current set of pseudonym certificates $\langle crt \rangle$ for the next time period t, thus creating $\langle crt \rangle'$. The routine may fail and output $\perp$ if crm is empty, i.e., there is no certificate refresh material, which was released by the CA, to create a new set of pseudonym certificates for the next time period.
- *revocation list refresh* ($\{\not\Vdash \langle crl \rangle'\} \leftarrow \mathsf{crlrefresh}(\langle crl \rangle, rrm, t)$) updates the current $\langle crl \rangle$ for the next time period t, possibly deleting crl associated with pseudonym certificates released more than $\mathtt{T}$ time periods ago. The routine may fail and output $\perp$ if rrm is empty, i.e., there are no revoked vehicles in the future time period.

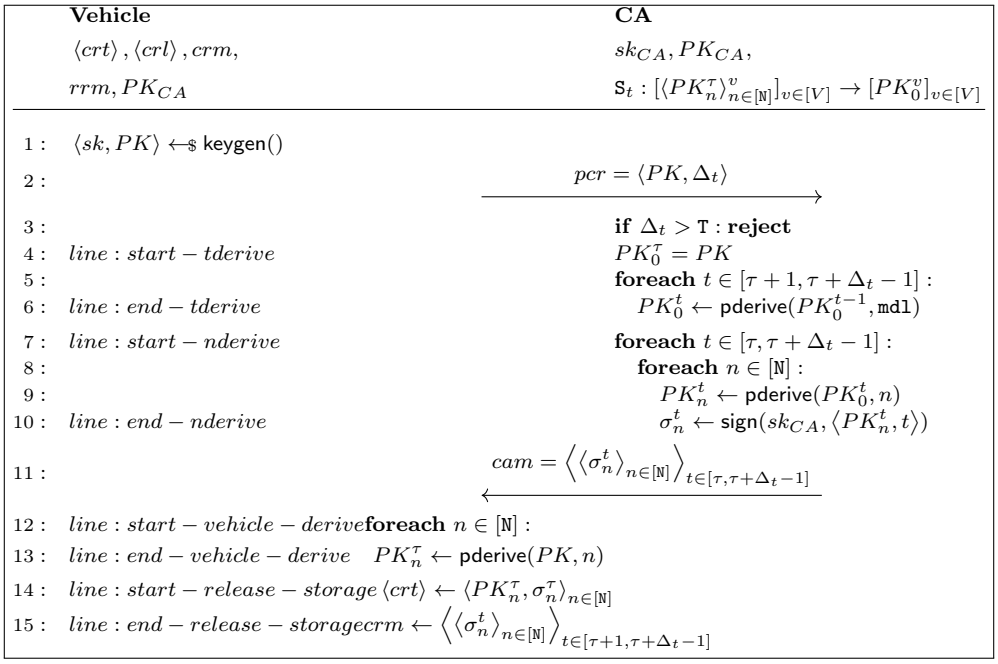

Figure 3: Specification of *pseudonym certificates release*

5 Network-efficient HKD-based PPV protocol specification

We describe a network-efficient PPV protocol specification (Section 4) based on homomorphic key derivation (HKD) functions (Section 2). We describe certificates release in Section 5.1, revocation in Section 5.2, and refresh in Section 5.3. We conclude with some final remarks on design choices and potential variants in Section 5.4.

We recall from Section 3 that a vehicle can request a set of N pseudonym certificates for each time period, for a total of T time periods, starting from the current time period τ and up to $\mathtt{T}-1$ time periods in the future.

5.1 Pseudonym certificates release

Figure 3 shows the specification of the *pseudonym certificates release* protocol including three routines: pcreq, pcgen, and pcrec. We assume that the vehicle is requesting pseudonym certificates for the first time at some time period τ, thus $\langle crt\rangle$, $\langle crl\rangle$, crm and rrm are empty at the beginning of the protocol. The vehicle generates a master key pair ($\langle sk, PK\rangle \leftarrow\$$ keygen()), and creates a pseudonym certificates request containing the master public key PK and the number of future time periods for which the vehicle is requesting pseudonym certificates Δ_t, where $\Delta_t \leq \mathtt{T}$. If the vehicle is requesting pseudonym certificates for more than one time period ($\Delta_t > 1$), the CA must derive the master public key PK_0^t for each time period $t \in [\tau+1, \tau+\Delta_t-1]$ by using pderive, where $info$ is set to the reserved value mdl (Lines **??-??**). A graphical example of key derivation operated by the pseudonym certificates release protocol is shown in Figure 4. Then, for each time period $t \in [\tau, \tau+\Delta_t-1]$ (where $PK_0^\tau = PK$), the CA derives Npseudonym public keys PK_n^t from PK_0^t using pderive, where $info = n, \forall n \in [\mathtt{N}]$, and signs them (together with the related time period t for which the keys are valid) (Lines **??-??**). For network efficiency, only digital signatures are sent back to the vehicle within *certificate approval material cam*. The vehicle receives *cam* and computes pseudonym public keys for the current time period (Lines **??-??**), postponing computation of public keys for following time periods during *refresh*. Thus, certificates for the current time period ($\langle PK_n^\tau, \sigma_n^\tau\rangle_{\forall n\in[\mathtt{N}]}$) are saved in $\langle crt\rangle$, while other signatures ($\langle\sigma_n^t\rangle_{t\in[\tau+1,\tau+\Delta_t-1],n\in\mathtt{N}}$) are stored in crm (Lines **??-??**).

Remarks. Note that, for computational efficiency at revocation time, our protocol requires the CA to

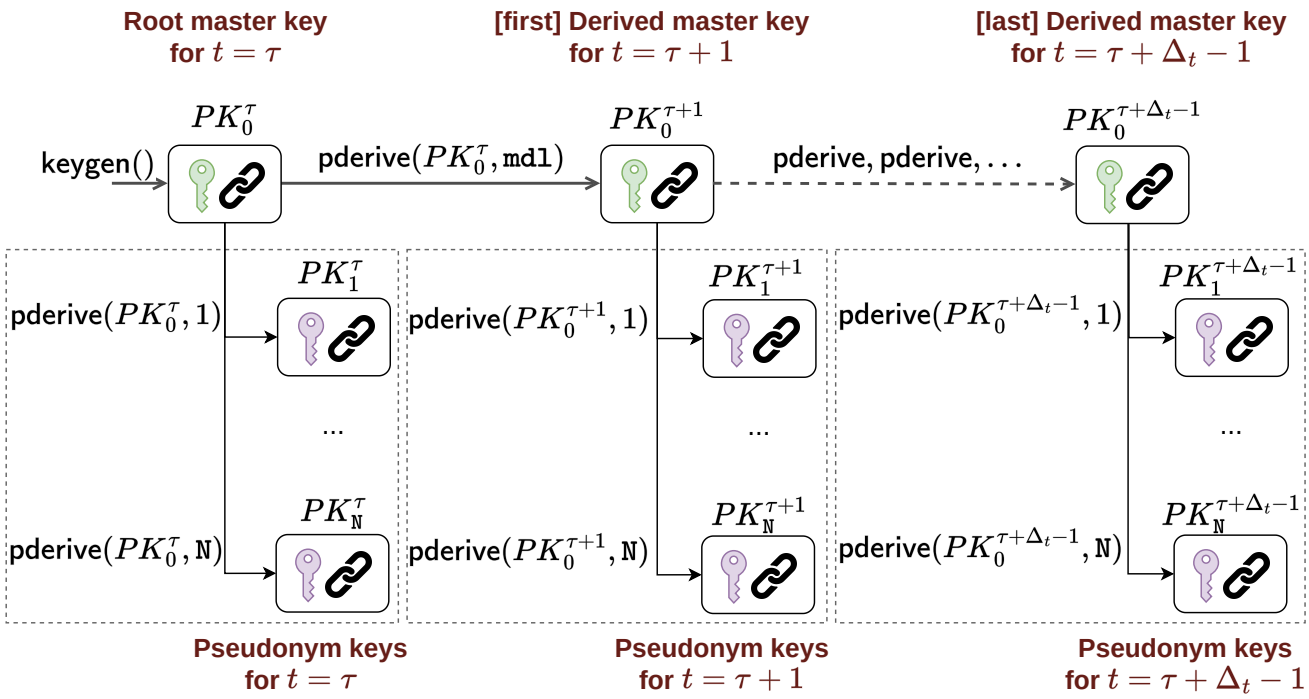

Figure 4: Derivation of public keys spanning Δ_t time periods.

maintain all the master public keys PK that each registered vehicle has sent during the pseudonym certificate release using a reverse hash map $\texttt{S}_t$. At time period t, $\texttt{S}_t$ maps each pseudonym public key PK_n^t to the corresponding master public key PK_0^t, for each vehicle $v \in [V]$ (where V is the total number of vehicles registered in the VPKI). In Figure 3 (and Figure 5) we denote this reverse hash map as a function $\texttt{S}_t : [\langle PK_n^\tau \rangle_{n\in[\texttt{N}]}^v]_{v\in[V]} \to [PK_0^v]_{v\in[V]}$, slightly abusing notation when writing PK_0^v to denote the master public key of the v-th vehicle. Also, in our specification we omit verification of the vehicle's identity id by the CA, which is orthogonal to the protocol (e.g., checking inclusion within a database of registered vehicles). Moreover, in order to minimize the size of messages sent over the network, the CA does not send the derived pseudonym public keys PK_n^t to the vehicle, but only the signatures σ_n^t and let the vehicle derive the pseudonym public keys on its own. This is possible because of the deterministic nature of the HKD scheme, but variants may let the CA also send public keys for easing computation at the vehicle side. Finally, note that at the end of the pseudonym certificates release protocol we let the vehicle only derive public keys, and not secret keys. This is due to the very low computation cost of the sderive routine, which includes the computation of a KDF function (that is, the execution of a few hash functions or block ciphers, depending on the specification [5]) and of a scalar addition operation, which is negligible compared to message signing. Thus, secret key derivation can be executed at runtime before sending the message, and vehicles can avoid storing (up to) Nsecret keys (see Section 6), which is a clear advantage with regard to standard VPKI protocols.

5.2 Pseudonym certificates revocation

Figure 5 shows details of the specification for *pseudonym certificates revocation*. Let crt be the certificate observed by the Misbehavior Authority (MA) sent within some malicious message at time period t, the CA only needs to publish a revocation material (rm) containing the master public key PK_0^t of the misbehaving vehicle, where t is the time period in which the misbehavior was detected, to revoke all its pseudonym certificates. The CA may also include an additional information if the revocation spans multiple time periods, i.e., if the

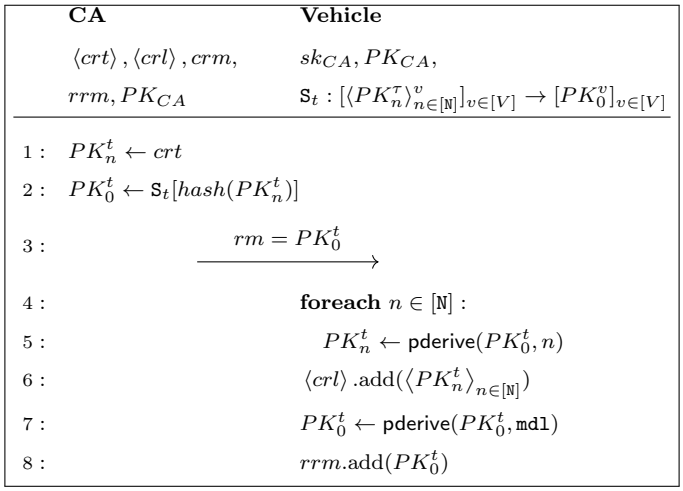

Figure 5: Specification of *pseudonym certificates revocation*.

misbehaving vehicle had requested pseudonym certificates for more than one time period, the revocation material must be valid also for those future time periods. The CA may also prevent the vehicle from requesting new pseudonyms in the future [3, 11]. A vehicle that receives rm can then derive all the pseudonym public keys associated with the misbehaving vehicle using the pderive routine for all $n \in \mathtt{N}$ with the master public key PK_0^t contained in the CRL. The vehicle stores all the derived pseudonym public keys for time period t in $\langle crl \rangle$, and stores in rrm the master public key PK_0^{t+1} for the next time period $t+1$, if required. When a vehicle receives a message, it can check if the message is signed with a pseudonym public key that is listed in $\langle crl \rangle$ and, possibly, discard the message.

Remarks. Note that, in Figure 5 (and Figure 6) we denote as .add() the addition of an element to a set without removing previous elements, if any. A critical advantage of adopting an HKD scheme is its support for efficient certificate revocation. Rather than referencing pseudonym certificates through global opaque identifiers (e.g., Serial Number in X.509 certificates [22] or linkage values in SCMS [3]), our protocol allows to derive all the pseudonyms of a vehicle by knowing only the master public key. This allows the CA to revoke all pseudonym certificates associated with a vehicle at some time period t (and for future time periods, if any) by simply publishing its master public key PK_0^t, without the need to send all the pseudonym certificates keys or identifiers. More importantly, V2X messages do not need to include any identifier, because at revocation time all vehicles can use pderive with a fixed set of known predefined indexes ($n \in [\mathtt{N}]$).

5.3 Certificate refresh

Figure 6 describes the specification of the *refresh* protocol which includes crtrefresh and crlrefresh. A vehicle executes crtrefresh if, at the end of the current time period τ, it has at least one set of signatures stored in crm. In other terms, if it successfully obtained pseudonym certificates during some time period t using as input $\Delta_t \geq (\tau - t)$. Otherwise, it must request a new set of explicit pseudonym certificates from the CA. The crtrefresh procedure allows the vehicle to derive a new master key pair $((sk_0^{\tau+1}, PK_0^{\tau+1}))$ for the next time period $\tau+1$ (i.e., invoking sderive and pderive using the current master key pair and the master derivation label mdl). Then, the vehicle derives all the pseudonym public keys $PK_n^{\tau+1}$ for the next time period, and generates the corresponding explicit certificates, to be stored in $\langle crt \rangle$, by using the signatures from crm. The unused signatures for time periods $t > \tau + 1$ are kept stored in crm for future invocations of crtrefresh. Whether the vehicle has refreshed $\langle crt \rangle$ from a previous set of signatures stored in crm or from a new invocation of the pseudonym certificate release protocol, it must also update its set of revoked pseudonym public keys for the

Vehicle (crtrefresh *procedure*)

1 : $sk_0^{\tau},\ PK_0^{\tau}$
2 : $sk_0^{\tau+1} \leftarrow \mathsf{sderive}(sk_0^{\tau}, PK_0^{\tau}, \texttt{mdl})$
3 : $PK_0^{\tau+1} \leftarrow \mathsf{pderive}(PK_0^{\tau}, \texttt{mdl})$
4 : **foreach** $n \in [\texttt{N}]$:
5 : $\quad PK_n^{\tau+1} \leftarrow \mathsf{pderive}(PK_0^{\tau+1}, n)$
6 : $\quad \sigma_n^{\tau+1} \leftarrow crm$
7 : $\langle crt \rangle \leftarrow \langle PK_n^{\tau+1}, \sigma_n^{\tau+1} \rangle_{n \in [\texttt{N}]}$

Vehicle (crlrefresh *procedure*)

1 : **foreach** $PK_0^{\tau+1} \in rrm$:
2 : $\quad$ **foreach** $n \in [\texttt{N}]$:
3 : $\quad\quad PK_n^{\tau+1} \leftarrow \mathsf{pderive}(PK_0^{\tau+1}, n)$
4 : $\quad \langle crl \rangle$ .add$(\langle PK_n^{\tau+1} \rangle_{n \in [\texttt{N}]})$
5 : $\quad rrm$.add$(\mathsf{pderive}(PK_0^{\tau+1}, \texttt{mdl}))$

Figure 6: Specification of *refresh*.

next time period. The crlrefresh procedure allows the vehicle to update $\langle crl \rangle$ as follows: for each revoked master public key (for $\tau + 1$) stored in rrm the vehicle derives all the revoked pseudonym public keys $PK_n^{\tau+1}$ for the next time period, and stores them in $\langle crl \rangle$. The crlrefresh procedure ends by saving in rrm the master public keys for future time period $t > \tau + 1$ (i.e., invoking pderive using the current master public key and the master derivation label `mdl`).

5.4 Final remarks

The proposed architecture reduces the storage requirements when compared to traditional VPKI systems. Vehicles only need to store a single master private key and a set of explicit certificates that are smaller than the pseudonym certificates containing global linkage identifiers [3, 12]. The scalability of the architecture is also improved, as the VPKI authority only needs to store the master public keys for each registered vehicle, rather than maintaining a large database of individual pseudonym certificates or revocation identifiers. Section 6 provides a detailed analysis of the storage and communication overhead of the proposed architecture.

While implicit certificates (e.g., ECQV [17] or SIMPL [1]) offer superior performance in terms of bandwidth efficiency, they rely on elliptic curve cryptography and do not currently extend to post-quantum settings [18], while, in contrast, explicit certificates are compatible with post-quantum signing schemes. We leave the integration of implicit certificates as a future work. Nevertheless, the use of HKD schemes still provide benefits in terms of scalable pseudonym generation and efficient revocation, even when combined with explicit certificates in post-quantum scenarios.

Protocol	Network cost
Release	$O(\mathtt{N} \cdot \mathtt{T})$
Revocation	$O(1)$
Refresh	-

Table 1: Asymptotic costs for network

Protocol	Procedure	Computation cost	Made by
Release	pcreq	$O(1)$	Vehicle
	pcrec	$O(\mathtt{N})$	
	pcgen	$O(\mathtt{N} \cdot \mathtt{T})$	CA
Revocation	revoke	$O(1)$ or $O(\mathtt{N})$	
	crlupdate	$O(\mathtt{N})$	Vehicle
Refresh	crtrefresh	$O(\mathtt{N})$	
	crlrefresh	$O(\mathtt{N} \cdot \mathtt{T})$	

Table 2: Asymptotic costs for computation

6 Cost evaluation

We discuss costs of the proposed HKD-based PPV protocol specification (see Section 5) and compare them with existing standards both asymptotically (Section 6.1) and analytically when instantiated with standard security parameters (Section 6.2).

6.1 Asymptotic analysis

We analyze asymptotic costs in terms of network and computation costs for the Release, Revocation and Refresh protocols, and in terms of storage size used by the vehicle for maintaining $\langle crt \rangle$, $\langle crl \rangle$, crm and rrm.

Table 1 shows asymptotic network costs, which we want to minimize because they are the most critical bottleneck in vehicular networks, especially when considering large number of vehicles. During Release, vehicles send a *pseudonym certificates request* to the CA including the master public key of the vehicle, the CA replies with all the signatures for each pseudonym certificate $\mathtt{N}$ in each time period $\mathtt{T}$, thus Release network cost is $O(\mathtt{N} \cdot \mathtt{T})$. During Revocation, the CA sends a single master public key to all the vehicles, which is used to revoke all the pseudonym certificates with public keys that have been derived from that master public key, regardless of their number, thus Revocation network cost is $O(1)$. Refresh does not require any network communication, as vehicles can update their local storage with the new pseudonym certificates and revocation lists without interacting with CAs.

While network costs are more critical in vehicular communications, it is important that computational costs are still affordable and kept below a certain threshold to comply with allowed latencies of safety-related features. Our proposal does not modify verification algorithms and thus safety-related features are not influenced. When dynamically deriving private keys, signing costs are only increased by a negligible amount (see remarks of Section 5.1). As for the CA, we assume that has enough computational resources to perform all the cryptographic operations required to manage the pseudonym certificates, and thus we do not consider the CA as bounded by the same constraints as the vehicles (the same applies for network communication, where we assume that the

Type of data	Storage	Storage cost
Pseudonym certificates valid in τ	$\langle crt \rangle$	$O(\mathtt{N})$
Certificates revoked by the CA in τ	$\langle crl \rangle$	$O(\mathtt{N} \cdot \mathtt{T})$
Certificate refresh material	crm	$O(\mathtt{N} \cdot \mathtt{T})$
Revocation refresh material	rrm	$O(\mathtt{N} \cdot \mathtt{T})$

Table 3: Asymptotic costs for storage

CA is not constrained by the same limitations as the vehicles).

Table 2 shows the asymptotic computation costs for certificate management. During Release, the vehicle performs a pseudonym certificate request (pcreq) that computes the master key pair, whose computation is independent of any protocol parameter except the security level, and thus its computation cost is $O(1)$. Then, CA performs the pseudonym certificate generation (pcgen) that requires up to $O(\mathtt{N} \cdot \mathtt{T})$ computation cost to generate all the pseudonym certificates for each time period requested by the vehicle. Note that we assume the worst case where the CA has to generate all the pseudonym certificates for the maximum allowed number of time periods $\mathtt{T}$. This assumption is justified by our main design goal of minimizing the network communication cost, thus reducing interactions among vehicles and CAs. Finally, the vehicle processes the certificate approval material (pcrec) to generate the $\mathtt{N}$pseudonym certificates for the current time period, thus requiring $O(\mathtt{N})$ computation cost. During Revocation protocol, the CA performs the revocation operation (revoke) that requires a constant computation cost of $O(1)$ to revoke a master public key, thanks to the reverse hash map (see Section 5.3). Different approaches where the CA does not maintain such a data structure, and thus has to perform more expensive search operations, could also be chosen to reduce the CA storage requirements, however we do not see any realistic scenario where a CA has such constrained storage. The vehicle then processes the certificate revocation list update (crlupdate) to update its local revocation list with the new revoked vehicle, thus requiring $O(\mathtt{N})$ computation cost to reconstruct all the pseudonym public keys of the revoked vehicle. During Refresh, the vehicle operates the certificate refresh routine (crtrefresh) and the revocation list refresh (crlrefresh) to update its local storage with new pseudonym certificates and revocation lists to be used in the next time period, which cost $O(\mathtt{N})$ and (up to) $O(\mathtt{N} \cdot \mathtt{T})$, respectively.

Table 3 shows storage asymptotic costs. Vehicles store all the pseudonym certificates valid in the current time period in $\langle crt \rangle$, that is, derived public keys and their associated signatures, thus storage cost is $O(\mathtt{N})$. Vehicles also store revocation list $\langle crl \rangle$, which contains $\mathtt{N}$pseudonym certificates for each vehicle that has been revoked by the CA in the current time period. However, since each vehicle can request pseudonym certificates for up to $\mathtt{T}$ time periods, the total storage required is $O(\mathtt{N} \cdot \mathtt{T})$ for each revoked vehicle. Furthermore, vehicles store certificate refresh material crm, which includes the signatures of pseudonym certificates that are valid in future time periods, thus its storage cost is also $O(\mathtt{N} \cdot \mathtt{T})$. Finally, vehicles store revocation refresh material rrm, which includes master public keys associated with the next time periods, thus storage cost is again $O(\mathtt{N} \cdot \mathtt{T})$.

6.2 Analytic evaluation for 128-bit security

We provide an analytic evaluation of our protocol and we compare it with the SCMS standard [3]. We consider an instantiation based on the ECDSA signature scheme [14] with the NIST p-256 elliptic curve [6], thus with a 128-bit security level, as recommended by current standards for V2X communications [19, 11].

Certificate size. The SCMS architecture [3] relies on opaque indexing identifiers called *linkage values*, which are included in each pseudonym certificate released by the CA (similar to the Serial Number in X.509 certificates [22]). Linkage values *globally* identify pseudonym certificates among all vehicles, except for a certain

probability of collision which depends on its size. The current recommended size for linkage values is 72 bits. The authors of SCMS suggest a flexible design of the linkage value to account for the increasing number of connected vehicles and potential weakness of the underlying cryptographic primitives in terms of collision resistance [3]. However, allowing such flexibility in real-world embedded systems may not be practical. Modifying the size of the linkage value after the system has been deployed would require a significant effort in terms of software and hardware updates, which is not feasible in many cases. $\mathtt{N}$ is defined by the VPKI authority at setup time, and allows a trade-off between privacy, security and scalability: the larger the value, the more pseudonym identities a vehicle can use within a time period, thus increasing anonymity (i.e., preventing identification and tracking based on a series of eavesdropped messages [20]), but also increasing storage requirements and risk of Sybil attacks [9]. We refer to the example from the discussion about the SCMS linkage value length in [3, Section V.C] to better understand the impact of this design choice. Consider 2.5×10^8 vehicles, each having $\mathtt{N} = 40$ pseudonym certificates per time period of one week, then in any time period there are $2.5 \times 10^8 \times 40 = 10^{10}$ pseudonym certificates in use. Revocation rate is assumed below 1%. In this setting, exploiting the birthday paradox [8], the chance of a collision between linkage values (i.e., the linkage value of a revoked vehicle gets assigned also to another vehicle in the same time period) is $\approx 1 - \exp(-((10^{10}/10^2) \times (10^{10}/10^2 - 1))/(2 \times 2^{72})) \approx 1.06 \times 10^{-6}$. While this probability is small, it is not *cryptographically* negligible [10], especially considering that the SCMS system is designed to be used for many years, and must account for the growth in the number of vehicles and pseudonym certificates over time. To obtain a collision probability that is *cryptographically* negligible, the size of the linkage value should be increased to at least 85 bits, which would result in a collision probability of $\approx 2^{-32} \approx 1.29 \times 10^{-10}$, further increasing the size of all certificates and thus the network congestion. In contrast, our protocol does not require inclusion of any identifier within a certificate, because certificate revocation is only based on cryptographic keys (Section 5.2). Since our approach aims to minimize the network communication cost, a reduction of 72 bits per certificate is significant, especially when considering the large amount of messages exchanged in a V2X network. Moreover, our approach is more *future-proof*, because its parameters are independent of workloads.

Certificate Revocation List size. In SCMS, the revocation of a vehicle is performed by distributing a Certificate Revocation List (CRL) that allows vehicles to reconstruct all the linkage values of the pseudonym certificates issued to the revoked vehicle. Two Linkage Authorities (LAs)[4] are involved in the revocation process: the CRL includes two linkage value seeds (i.e., hash chains) and the identities of the two LAs. The total size of each CRL is $2 \times 128 + 2 \times 32 = 320$ bits, where 128 bits are the linkage seeds created by the LAs and 32 bits are identity strings associated with the LAs. When a vehicle receives a CRL, it can reconstruct all the linkage values of the certificates of the revoked vehicle. On the vehicle side this means that a single CRL expands to $\mathtt{N}$ linkage values, each of size 72 bits, for a total of $72 \cdot \mathtt{N}$ bits that must be stored to check the revocation status of each V2X message received. The authors of SCMS assume that all vehicles will provide at least enough storage for 10000 CRL entries (where a single CRL entry corresponds to a revoked vehicle), which translates to $320/8 \times 10000 = 400$ KB. But, when expanding each CRL entry, the total required storage space is $72\mathtt{N}/8 \times 10000 = 90\mathtt{N}$ KB, which, for $\mathtt{N} = 40$, results in 3600 KB. In our architecture, the revocation of a vehicle is performed by distributing a CRL that includes the master public key of that vehicle, which is of size 256 bits for the NIST p-256 elliptic curve [6]. When a vehicle receives a CRL, it can reconstruct all the pseudonym public keys issued to the revoked vehicle by deriving the set of pseudonym public keys from the master pseudonym public key. Assuming the same number of CRL entries, the total required size is $256/8 \times 10000 = 320$ KB, before expansion. When expanding each CRL entry, the total required storage space for $\mathtt{N} = 40$ is $256/8 \times 40 \times 10000 = 12800$ KB. This is $\sim$3.5$\times$ larger than the SCMS approach for explicit certificates, however, this is still acceptable in practice, as the increased storage space requirements are manageable for modern vehicles. This difference in storage

[4]This is due to the different VPKI architecture: in SCMS there are four different authorities, and the authority involved in certificate revocation is the Linkage Authority.

requirements holds also when compared to the SCMS approach for implicit certificates, as implicit certificates in SCMS use the same linkage value mechanism as explicit certificates, thus requiring the same CRL storage space for 10000 entries.

7 Conclusions

This document presented a novel protocol framework for accountability in vehicular networks, with a focus on efficient certificate management for pseudonym-based authenticated communications. Accountability is a critical requirement for ensuring the security and trustworthiness of Vehicle-to-Everything (V2X) communications, as it allows for the non-repudiable attribution of actions to specific entities, enabling effective revocation of malicious actors from the network. By leveraging hierarchical deterministic homomorphic key derivation schemes, we designed a protocol that significantly reduces network overhead while maintaining strong security and privacy guarantees. Our analyses showed that the proposed approach allows for efficient certificate issuance and revocation, with constant network costs for revocation operations, making it suitable for large-scale deployment in V2X communication systems. Future work will explore further optimizations and the integration of implicit certificates to enhance the efficiency of vehicular credential management even further.

References

[1] Paulo S. L. M. Barreto, Marcos A. Simplicio, Jefferson E. Ricardini, and Harsh Kupwade Patil. Schnorr-based implicit certification: Improving the security and efficiency of vehicular communications. *IEEE Trans. Computers*, 2021.

[2] Norbert Bißmeyer, Hagen Stübing, Elmar Schoch, Stefan Götz, Jan Peter Stotz, and Brigitte Lonc. A generic public key infrastructure for securing car-to-x communication. In *18th ITS World Congress, Orlando, USA*, 2011.

[3] Benedikt Brecht, Dean Therriault, André Weimerskirch, William Whyte, Virendra Kumar, Thorsten Hehn, and Roy Goudy. A security credential management system for V2X communications. *IEEE Trans. Intelligent Transportation Systems*, 2018.

[4] Lily Chen. Recommendation for key derivation using pseudorandom functions, 2024.

[5] Lily Chen and Lily Chen. *Recommendation for key derivation using pseudorandom functions*. US Department of Commerce, National Institute of Standards and Technology, 2024.

[6] Lily Chen, Dustin Moody, Karen Randall, Andrew Regenscheid, and Angela Robinson. Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters, 2023.

[7] Poulami Das, Sebastian Faust, and Julian Loss. A formal treatment of deterministic wallets. In *Proc. ACM CCS*, 2019.

[8] Harold Davenport. *The higher arithmetic: An introduction to the theory of numbers*. Cambridge University Press, 1999.

[9] John R Douceur. The sybil attack. In *Int'l. workshop peer-to-peer systems*, 2002.

[10] Morris Dworkin. Recommendation for block cipher modes of operation: Galois/counter mode (gcm) and gmac, 2007.

[11] ETSI TS 102 731 V2.0.0. Intelligent Transport Systems (ITS); Security; Security Services and Architecture, 2022.

[12] ETSI TS 103 097 V2.1.1. Intelligent Transport Systems (ITS); Security; Security header and certificate formats, 2021.

[13] Hugo Krawczyk and Pasi Eronen. HMAC-based Extract-and-Expand Key Derivation Function (HKDF). RFC 5869, 2010.

[14] National Institute of Standards, Technology (NIST), Lily Chen, Dustin Moody, Andrew Regenscheid, and Angela Robinson. Digital signature standard (dss), 2023.

[15] Jonathan Petit, Florian Schaub, Michael Feiri, and Frank Kargl. Pseudonym schemes in vehicular networks: A survey. *IEEE Comm. Surveys & Tutorials*, 2014.

[16] Maxim Raya, Panagiotis Papadimitratos, Imad Aad, Daniel Jungels, and Jean-Pierre Hubaux. Eviction of misbehaving and faulty nodes in vehicular networks. *IEEE journal selected areas in communications*, 2007.

[17] Certicom Research. SEC 4: Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV), 2004. Standards for Efficient Cryptography.

[18] Geoff Twardokus, Nina Bindel, Hanif Rahbari, and Sarah McCarthy. When cryptography needs a hand: Practical post-quantum authentication for v2v communications. In *NDSS Symp.*, 2024.

[19] V2X Core Technical Committee. *SAE J 2735 - V2X Communications Message Set Dictionary*, 2024.

[20] Björn Wiedersheim, Zhendong Ma, Frank Kargl, and Panos Papadimitratos. Privacy in inter-vehicular networks: Why simple pseudonym change is not enough. In *Int'l. Conf. wireless on-demand network systems and services*, 2010.

[21] Pieter Wuille. Bitcoin Improvement Proposal (BIP) 32: Hierarchical Deterministic Wallets, 2012.

[22] Peter E. Yee. Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 6818, 2013.