

Revisiting the ETSI ITS Lifecycle with Certificateless Authorization Based on Group Signatures

Riccardo Gennaro*, Stefano Berlato[†], Alessandro Tomasi[†], Silvio Ranise^{†‡} and Florian Hahn*

* *University of Twente, Enschede, The Netherlands*

Email: r.gennaro@student.utwente.nl, f.w.hahn@utwente.nl

[†] *Fondazione Bruno Kessler, Trento, Italy*

Email: {sberlato,altomasi,ranise}@fbk.eu

[‡] *University of Trento, Trento, Italy*

Abstract—The current standards for Intelligent Transport Systems (ITSs) by the European Telecommunications Standards Institute (ETSI) rely heavily on Public Key Infrastructures (PKIs) and pseudonym-based digital certificates to provide message authenticity and user privacy in vehicular communications. Although effective, this approach introduces substantial complexity due to heavy certificate management and network overhead, particularly in cases of dense traffic. To simplify certificate management without sacrificing interoperability, this paper proposes a standard-compatible redesign of (part of) ETSI’s authentication and authorization lifecycle that replaces pseudonym-based certificates with Group Signatures (GSs). Our redesign preserves the separation of duties between Enrollment Authorities (EAs) and Authorization Authorities (AAs), balancing authenticity, unlinkability, non-repudiation, and limited anonymity. Also, we implement a proof-of-concept within the open-source C2C—Common platform using IBM’s `libgroupsig` library. Our benchmarks show that our redesign introduces substantial yet improvable signing and verification overheads — 3.7× and 10×, respectively — while maintaining comparable message sizes. Finally, we discuss a further ETSI-compatible extension with Attribute-Based Encryption (ABE) to introduce fine-grained access control aligned with ETSI’s permission codes and C-Roads use cases.

Index Terms—Group Signatures, ETSI, ITS, VANETs

1. Introduction

The rising volume of road traffic poses major challenges for transportation systems, including congestion, safety risks, and environmental impact. To address these challenges, Intelligent Transport System (ITS) applications have been devised to offer ITS services like collision avoidance, real-time traffic management, and emergency notifications. However, ITS services need strong security guarantees against disruptions and manipulations to ensure safety. In fact, ITS services face a wide range of adversaries, commonly categorized as passive or active [2], [33], and as external or internal [1]. *Passive* attacks, like eavesdropping and traffic analysis, aim to intercept communications with techniques such as packet sniffing

and radio-frequency scanning. Conversely, *active* attacks aim to manipulate or inject falsified data — examples include replay, impersonation, and message modification — and can lead to severe safety consequences. *External* attacks come from outside the ITS service boundary, while *internal* attacks are carried out by legitimate but malicious, or compromised, network participants.

Security and Privacy Requirements. Balancing security and performance has always been a challenge in ITS services [10]. Although performance requirements often vary depending on the specific ITS service, the following security and privacy requirements are almost always present to ensure ITS service safety [33]:

- *Authenticity* (which entails also *integrity*). Every message exchanged among vehicles and roadside infrastructure must be authenticated to prevent impersonation and identify forged or manipulated messages.
- *Confidentiality*. Some ITS services — e.g., those transmitting sensitive data or accessible via a fee or subscription — may require message confidentiality.
- *Unlinkability*. Adversaries and honest-but-curious authorities may track vehicle activities within and across ITS services. Hence, messages should be unlinkable to vehicles — to a certain extent (see below).
- *Limited anonymity*. Authorized parties — e.g., police officers — must be able to identify which vehicle sent a particular message under certain conditions (e.g., when investigating accidents or for law enforcement).
- *Non-repudiation*. It must not be possible to deny the transmission of a message by a given vehicle.

Standardization Efforts. Standards have been recognized as enablers to foster harmonization and interoperability of ITS services [5]. During the last decade, multiple standards bodies — such as the European Telecommunications Standards Institute (ETSI) in Europe and the Institute of Electrical and Electronics Engineers (IEEE) in the United States — have defined frameworks for the development of safe and secure ITS services.¹ In particular, current ETSI-compliant systems, e.g., the Cooperative ITS European Initiative (C-Roads) [9], rely heavily on Public Key Infrastructures (PKIs) and pseudonym-based

This work was partially supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union (EU) - NextGenerationEU.

1. We focus on ETSI’s standardization efforts for communication-level authentication and authorization; IEEE’s standards — and others, e.g., those of the Society of Automotive Engineers (SAE) for intra-vehicle cybersecurity [13] — are out of scope.

authentication and authorization to secure ITS services. ETSI standards (which we discuss in more detail in Section 2.3) prescribe each vehicle to be given a batch of rotating digital certificates. Consequently, vehicles can periodically rotate the certificate used for authenticating outgoing messages to achieve a pseudonymity guarantee. Then, rather than adding revoked certificates to one or more Certificate Revocation Lists (CRLs) — which would quickly become too long — certificates are simply short-lived and are not renewed once expired; new certificates are issued only to registered and active vehicles. However, while achieving the security and privacy requirements previously discussed, ETSI’s approach complicates certificate management due to the high certificate rotation frequency — which depends on the granularity of the time validity window. Also, network traffic increases as frequent transmissions of fresh certificates are required. Moreover, not only is this issue exacerbated in the case of dense traffic, but also a trade-off is introduced: while a narrow validity window creates high renegotiation-related traffic, a wide window enables adversaries to correlate certificates, compromising unlinkability.

Solution and Contributions. We propose a standard-compatible redesign of the pseudonym-based authentication and authorization mechanisms currently standardized by ETSI for ITS services. In essence, we suggest replacing large batches of pseudonyms with a hybrid construction combining Group Signatures (GSs) for unlinkability and Attribute-Based Encryption (ABE) for fine-grained service authorization — solving the aforementioned issues while still achieving the desired security and privacy requirements. Our redesign explicitly maps ETSI entities to GS and ABE authorities, and preserves ETSI message formats so that our proposal can be deployed incrementally.

In the following, after discussing background information and related work in Sections 2 and 3 (and before concluding the paper with final remarks and future work in Section 8), we make the following contributions:

- *Scheme selection.* The analysis of the security model and performance of multiple GS schemes for the requirements of ITS services (Section 4).
- *Design.* A revisited ETSI lifecycle with detailed message exchange sequence diagrams showing the integration of GSs and a clear mapping of responsibilities for the GS authorities (Section 5).
- *Implementation and Preliminary Evaluation.* A proof-of-concept implementation of the revisited lifecycle within C2C-Common — an open-source implementation of ETSI TS 102 941 [18] and 103 097 [14] — together with a preliminary performance evaluation and comparison with respect to the original lifecycle (Section 6).
- *Further Improvements.* A discussion on how ABE may fit into the revisited lifecycle with detailed message exchange sequence diagrams (Section 7).

2. Background

Below, we briefly recall the functioning of a PKI (Section 2.1) and the main definitions and schemes for GS (Section 2.2). Then, we outline the various ETSI standards

for ITS, including an overview of ETSI’s security architecture, authentication and authorization lifecycle, certificate formats, and communication patterns (Section 2.3).

2.1. Public Key Infrastructure

A PKI supports the management of digital certificates (or simply certificates) over key pairs used for, e.g., secure key establishment and digital signatures. A certificate binds a public key to the holder of the corresponding private key, along with further subject attributes — e.g., domain name, validity period. Moreover, a PKI maintains and provides information about the status of unexpired or revoked certificates [30]. Two fundamental components of PKIs are Certification Authorities (CAs) and Registration Authorities (RAs): CAs create, issue, distribute, and revoke certificates, while RAs verify — on behalf of CAs — the accuracy of the attributes of PKI users before issuing the corresponding certificates [30]. Since anyone can verify the authenticity and validity of certificates, the use of certificate-bound key pairs ensures authenticity [30]. Finally, CAs can establish trust relationships by signing — in essence, endorsing — each others’ certificates, creating a so-called *chain of trust* establishing a hierarchy of CAs and, possibly, span across different security domains [30].

2.2. Group Signatures

GSs schemes [11] allow members of a group to produce digital signatures — called *group signatures* — such that the signer’s identity remains hidden while the validity of their group membership can be publicly verified. Generally, a GS scheme is a digital signature scheme in which (i) members of the group can produce GSs with individual secret keys; (ii) GSs are verifiable with a single public key; (iii) a special member — the *group manager* — can “open” a GS to reveal the signer’s identity, enabling non-repudiation [11]. The precise definition of a GS scheme varies according to its *security model*. A famous model for GSs was proposed by Bellare, Micciancio, and Warinschi (BMW) [3] and comprises four algorithms:

- $Setup(1^\lambda, 1^n) \rightarrow (gpk, gmsk, gsk_{(1,\dots,n)})$. Given a security parameter 1^λ and the group size 1^n , the group manager generates the group public key gpk , the group manager’s secret key $gmsk$, and a (private) signing key gsk_i for each group member $i \in [n]$.
- $Sign(m, gsk_i) \rightarrow \sigma$. A group member i signs a message m using gsk_i to produce a GS σ .
- $Verify(m, \sigma, gpk) \rightarrow \{accept, reject\}$. Anyone can verify a σ for a m using gpk . A σ generated by $Sign$ over a (not tampered) message is always accepted, i.e., $Pr[Verify(gpk, m, Sign(m, gsk_i)) = accept] = 1$. The verification of a GS does not reveal the identity of the signer.
- $Open(\sigma, gmsk) \rightarrow i$. The group manager can use $gmsk$ to reveal the identity i of the signer of σ .

An example of GS scheme based on BMW is BBS04 by Boneh et al. [7]. However, BMW supports static groups only, where members are defined during setup and cannot be changed.² Dynamic groups were introduced with the

2. Actually, BBS04 can support dynamic groups, but its security has been demonstrated in the static BMW model only, and not in dynamic models. Thus, dynamic BBS04 has no suitable security guarantees.

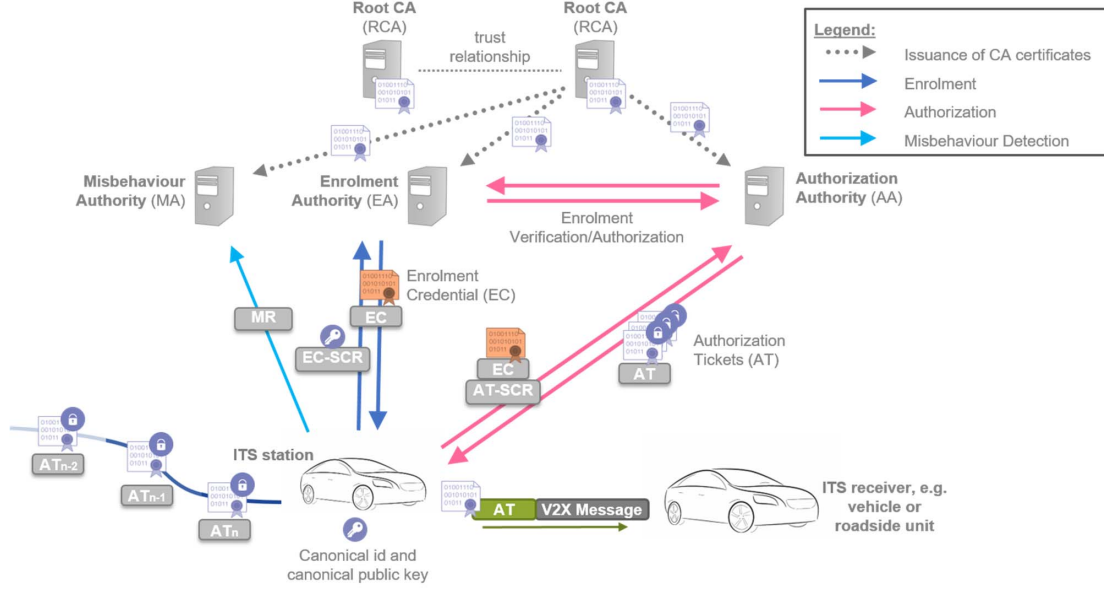


Figure 1. ETSI ITS security architecture from [16]. The figure depicts a single PKI with one EA, AA, and MA, as well as the interactions with an ITS-S. A representation of the ETSI ITS security architecture with multiple PKIs — as well as the CPOC and the TLM — can be found in [26].

BSZ model [4] — e.g., implemented by KLAP20 [25] — which adds the algorithms *Issue*, *Join*, and *Judge* and distinguishes between the entity that can add members (*issuer*) and the entity which can open GSs (*opener*):³

- $Setup(1^\lambda) \rightarrow (gpk, gsk_{iss}, gsk_{open})$. A trusted party generates the group public key gpk , the issuer’s secret key gsk_{iss} , and the opener’s secret key gsk_{open} .
- $Sign(m, gsk_i) \rightarrow \sigma$ and $Verify(m, \sigma, gpk) \rightarrow \{accept, reject\}$. As in BMW.
- $Issue(gpk, pubk_i, gsk_{iss}) \rightarrow (reg_i, cred_i)$. The issuer produces the registration information reg_i and credential $cred_i$ for the user i using gsk_{iss} , gpk , and the user’s public key $pubk_i$.⁴
- $Join(gpk, pubk_i, privk_i, cred_i) \rightarrow gsk_i$. The user i interacts with the group manager to derive its signing key gsk_i using gpk , $cred_i$, $pubk_i$, and the private key $privk_i$ associated to $pubk_i$.
- $Open(gpk, gsk_{open}, reg, m, \sigma) \rightarrow (i, \pi)$. The opener can use gsk_{open} — and the issuer’s register reg containing all registration information — to reveal the identity i of the signer from a σ over a message m . Also, $Open$ returns a proof π that the opener executed the $Open$ algorithm correctly.
- $Judge(gpk, i, pubk_i, m, \sigma, \pi) \rightarrow \{accept, reject\}$. Anyone can verify that the group member i did create σ over m by using the proof π .

Two notable further models were proposed in the literature, CLS [22] and UCL [12]: CLS allows for linking GSs without revealing the signer’s identity, and both UCL and CLS remove the *Open* algorithm and let linkability to be user-controlled. Finally, Pointcheval and Sanders propose a GS scheme (PS16) operating under the BSZ model (with joined issuer and opener) and producing shorter signatures and efficient *Sign* and *Verify* algorithms [31].

3. As in [32], members have both a public and private signing key.

4. While in [32] *Issue* outputs only the registration information reg_i , the *Issue* algorithm of the BSZ GS scheme discussed in Section 4 also outputs a credential $cred_i$ that is sent to the user to be included in gsk_i .

2.3. ETSI Standards

We now outline the various ETSI standards concerning the security of ITSs from complementary viewpoints: architectural, lifecycle, certificate formats, and communication patterns. In detail, we consider the following ETSI technical specifications (ETSI TS): 102 940 [16], 102 941 [18], 103 097 [14], 103 900 [21], 103 831 [15], and 103 759 [17].

Security Architecture Overview. As shown in Figure 1, the security architecture for ITS services proposed by ETSI in 102 940 [16] — and further described in the Certificate Policy for Cooperative Intelligent Transport System (C-ITS) document [26] — comprises multiple independent PKIs. Each PKI can be operated by, e.g., a national organization, a commercial entity, or a European institution. Interoperability among the PKIs is provided by a Central Point Of Contact (CPOC) and a Trust List Manager (TLM), which, essentially, facilitate the establishment of trust relationships between PKIs. ETSI TS 102 940 [16] and 102 941 [18] state that, besides a Root Certificate Authority (RCA), each PKI comprises also one or more Enrollment Authorities (EAs), Authorization Authorities (AAs), and Misbehaviour Authorities (MAs), each being independent and having jurisdiction on a specific regional or national geographical area. Within such an area, the EA provides long-term identities to requesting vehicles and Road Side Units (RSUs) — collectively referred to as ITS-Stations (ITS-Ss) — in the form of certificates called *Enrollment Certificates (ECs)*. A vehicle can request its first EC by presenting to the EA its canonical identifier — i.e., its Vehicle Identifying Number (VIN) and a tuple of preloaded key pairs provisioned and certified by the vehicle’s manufacturer — and subsequent ECs by presenting the previous EC. Also, the EA maintains the list of permissions assigned to an ITS-S. Differently, the AA provides pseudonyms to requesting ITS-Ss (that present a valid EC) in the form of short-lived

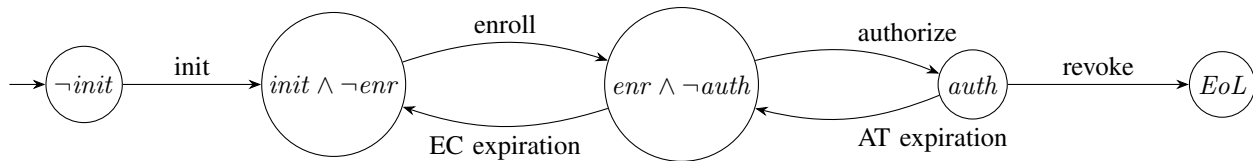


Figure 2. The ETSI ITS authentication and authorization lifecycle.

certificates called *Authorization Tickets (ATs)*. In detail, the Certificate Policy for C-ITS document [26] prescribes that the validity window for ATs can be up to a week. Moreover, ITS-Ss can preload ATs for up to three months in the future. Finally, an ITS-S may send a report (e.g., a notice of misbehavior of another ITS-S) to the MA that will investigate the report and decide whether to ignore it, simply warn the reported ITS-S, or request its revocation.

The use of short-lived ATs provides authenticity while hindering the tracking of ITS-Ss, since ITS-Ss can periodically rotate ATs while accessing ITS services, as mandated by ETSI TS 102 941 [18]; neither canonical identifiers nor ECs are suitable for use in ITS services. More specifically, the privacy areas⁵ defined in ETSI TS 102 940 [16] and 102 941 [18] include the protection of ITS registration data and the privacy of communications between ITS-Ss. Consequently, authentication and authorization are carried out in such a way that the canonical identifier of an ITS-S is known only by the EA. Furthermore, ETSI enforces a separation of duties between EA — responsible for enrollment and identity management — and AA — responsible for issuing and managing ATs.

Lifecycle Overview. The authentication and authorization lifecycle for ITS-Ss defined in ETSI TS 102 941 [18] is represented by the 5-state deterministic automaton shown in Figure 2. First, a *not initialized* ($\neg init$) ITS-S is initialized (*init*) by the manufacturer with a canonical identifier and contact information (e.g., certificates) for communicating with EAs and AAs. Then, an *initialized and not enrolled* ($init \wedge \neg enr$) ITS-S can enroll by interacting with an EA (*enroll*): the first enrollment requires the canonical identifier, while subsequent re-enrollments — which occur when an EC expires (*EC expiration*) — also require the previous EC. An *enrolled and not authorized* ($enr \wedge \neg auth$) ITS-S can require ATs by sending its EC to the AA (*authorize*) and become *authorized for service* (*auth*). An authorized ITS-S signs messages and accesses ITS services using its ATs — authorization is passively revoked as ATs expire (*AT expiration*). Finally, an ITS-S reaches the end state *End of Life (EoL)* when decommissioned or compromised (*revoke*).

Certificate Format Overview. ECs and ATs — seen as certificates — are composed of multiple fields stating both the identity and the permissions of their holder. The semantics of these fields is given in IEEE Std 1609.2 [24], and the certificate format (which contains additional requirements) is specified in ETSI TS 103 097 [14]. Specifically, the permissions stated in an AT prove to a receiving ITS-S that the sender has the right to

send a given type of message. These permissions are set in the field `appPermissions` and consist of tuples composed by *ITS Application Identifier (ITS-AID)* and *Service Specific Permission (SSP)* codes. In particular, ITS-AIDs identify ITS services — e.g., Cooperative Awareness Message (CAM) and Decentralized Environmental Notification Message (DENM) correspond to ITS-AID 36 and 37, respectively — while SSPs’ semantics differ according to the underlying ITS service. For instance, ETSI TS 103 900 [21] specifies that, for the CAM service, octet 1 of the SSP represents the role of the ITS-S (e.g., emergency vehicle) while octet 2 contains flags for specific roles indicating whether a ITS-S can send a given type of CAM (e.g., a lane closure notification). Instead, ETSI TS 103 831 [15] specifies that, for the DENM service, octets 1 to 3 represent the specific cause for which the message has been sent (e.g., bit 1 of octet 1 for the SSP in a given AT allows an ITS-S to sign an accident notification).

Communication Pattern Overview. ETSI TS 102 941 [18] defines three communication patterns — multicast, broadcast, and unicast — in wireless networks enabling communication among ITS-Ss; such networks are often also called Vehicular Ad hoc Networks (VANETs). Unlike safety-related broadcast services such as CAMs and DENMs, multicast and unicast services may be offered by multiple service providers and can be commercially sensitive (e.g., fleet management, paid real-time traffic analytics). As a result, different ITS services may present different security and privacy requirements. For instance, for *broadcast Security Associations (SAs)*, ETSI TS 102 941 [18] requires authentication, authorization, and integrity, but not confidentiality, while *Multicast and unicast SAs* also requires confidentiality, to be satisfied with symmetric cryptography [14]. In this case, the symmetric key is distributed in the multicast group using a public key encryption scheme. An ITS-S may join such services using its AT, possibly followed by further steps (e.g., registration to the service). Unicast key management may be service-specific or use standard key management systems such as network layer security with Internet Protocol Security (IPsec) following the IETF RFC documents listed in ETSI TS 102 941 [18]. Security at the transport layer can be provided using methods such as Transport Layer Security (TLS) [14], [18].

3. Related Work

GS-based Approaches. Guo et al. [23] were among the first to investigate the use of GSs for authenticating ITS-Ss messages in VANETs, and for allowing group managers to open GSs for non-repudiation. Following the work in [23], Short Group Signatures (SGSS) schemes [7] were proposed in [35], [36] to enable batch verification of safety

5. In ETSI terminology, a privacy area is an aspect or domain of the ITS security architecture designed to protect users’ and vehicles’ privacy.

	BBS04 [7]	PS16 [31]	GL19 [22]	KLAP20 [25]	DL21 [12]
Model	BMW	BSZ	CLS	BSZ	UCL
Sign	$3e_{\mathbb{G}_T}, 9e_{\mathbb{G}_1}$	$1e_{\mathbb{G}_T}, 2e_{\mathbb{G}_1}$	$16e_{\mathbb{G}_1}, 15e_{\mathbb{Z}_{n^2}}$	$4e_{\mathbb{G}_1}$	$14e_{\mathbb{G}_1}$
Verify	$1P, 3e_{\mathbb{G}_T}, 2e_{\mathbb{G}_2}, 8e_{\mathbb{G}_1}$	$3P, 3e_{\mathbb{G}_1}$	$2P, 12e_{\mathbb{G}_1}, 11e_{\mathbb{Z}_{n^2}}$	$3P, 2e_{\mathbb{G}_1}$	$2P, 9e_{\mathbb{G}_1}$
Group Signature Length	$3\mathbb{G}_1, 6\mathbb{Z}_p$	$2\mathbb{G}_1, 2\mathbb{Z}_p$	$3\mathbb{G}_1, 6\mathbb{Z}_p, 6\mathbb{Z}_{n^2}^*, 1H$	$3\mathbb{G}_1, 2\mathbb{Z}_p, 1H$	$4\mathbb{G}_1, 5\mathbb{Z}_p, 3H$

TABLE 1. MODEL, ALGEBRAIC COSTS, AND SIGNATURE LENGTH OF THE CONSIDERED GS SCHEMES.

messages for static groups of vehicles. To reduce communication latency, Zhang et al. [38] propose to make RSUs operate as group managers in their geographic domain, rather than having a single centralized group manager. Building on this idea, Lim et al. [27] propose domains with multiple RSUs [28] for more efficient group-key distribution; the drawback is that a malicious or compromised RSU may expose vehicle information [33]. More recently, Zhang et al. [37] combine batch verification with shared group session keys to speed up revocations [33].

Overall, the literature offers several GS-based approaches providing authenticity, unlinkability, and non-repudiation. Limited anonymity, instead, depends on non-trivial considerations on trust management (e.g., who may open a signature and under what circumstances) that need to take into account regulatory frameworks, standards, and architectures. In fact, these GS-based approaches were not designed with ETSI ITS message formats or authority separation in mind. For instance, many assume RSUs or a single group manager as trusted entities, whereas ETSI requires distinct EAs and AAs (recall the description in Section 2). Our solution likewise uses GSs to achieve authenticity, unlinkability, and non-repudiation, but also accounts for limited anonymity by explicitly mapping GS roles onto ETSI’s architecture and lifecycle and by eliminating per-message certificates.

Certificate-based Approaches. Improving upon PKI-based pseudonyms, Moussaoui et al. [29] propose using common pseudonyms among nearby vehicles to confuse trackers, while Singh et al. [34] suggest vehicles to exchange pseudonyms cooperatively to thwart tracking.

These works show creative privacy gains but still incur the drawbacks shown in Section 1 (e.g., heavy certificate management, network overhead, unlinkability trade-off).

Hybrid Approaches. Bussa et al. [8] combine GSs with traditional PKI so that vehicles can use GSs to self-sign their own pseudonym certificates; messages are then signed under these self-issued pseudonym certificates to achieve anonymity with respect to other vehicles. Still, as the authors observe, vehicles must perform long-term authentication with RSUs. In contrast, our solution preserves vehicle anonymity and unlinkability from RSUs as well.

In summary, GS-only approaches are incompatible with ETSI standards, and certificate-based approaches focus only on enhancing privacy. The work in [8] comes closest to our goal, but considers neither unlinkability from RSUs nor compliance with ETSI message formats and EA/AA separation. To the best of our knowledge, ours is the first ETSI-compatible integration of GSs.

4. Scheme Selection

Below, we analyze the suitability of those GS models and schemes described in Section 2 — i.e., BMW (BBS04

[7]), BSZ (PS16 [31] and KLAP20 [25]), CLS (GL19 [22]), and UCL (DL21 [12]) — according to both their security model and performance. In particular, we evaluate the length (in bytes) of the corresponding GSs as well as the run time of those algorithms common to all models and known to potentially impact ITS communications, i.e., *Sign* and *Verify* — the other algorithms (i.e., *Setup*, *Open*, *Issue*, *Judge*, and *Join*), while still relevant, are executed periodically or occasionally only.

Algebraic Cost. We compare costs and signature length of the GS schemes at a theoretical level (that is, in terms of number and type of mathematical operations involved) in Table 1. For simplicity, we represent the cost of a mathematical operation with the symbol of the operation itself: P denotes a pairing operation, $e_{\mathbb{G}}$ an exponentiation in \mathbb{G} ; H is the length of an hash function, while \mathbb{G}_T and \mathbb{G}_1 are element sizes in the respective groups.

Benchmarks. We use IBM’s `libgroupsig` library⁶ — which provides a C implementation of the GS schemes — as a non-biased common baseline for our benchmarks. We test all schemes on the popular BLS12_381 for pairings, and execute the benchmarks on a laptop running Arch Linux x86_64 with kernel version 6.15.7-arch1-1 and an Intel i7-8550 @ 4.00GHz CPU. To reduce measurement errors, we employ the Google Benchmark library⁷ set to 250 iterations per algorithm (i.e., *Sign* and *Verify*) for each GS scheme. The benchmark results are shown in Figure 3, while Figure 4 shows the length of signatures computed over an average-sized DENM message.

Model. Concerning the suitability of GS security models to the requirements elicited in Section 1, we note that ITS services are *dynamic* in nature, and ITS-Ss continuously enter and exit geographical areas that may be tied to different PKI jurisdictions. Hence, a suitable GS model must support dynamic groups. As a result, we exclude BMW, which lacks the *Join* algorithm. Then, although all models inherently provide authenticity and unlinkability, not all provide *limited anonymity*, in the sense that only authorized parties can link (GSs of) messages to ITS-Ss. In detail, CLS and UCL do not offer the *Open* algorithm (or an equivalent opening-like feature) and instead provide user-controlled linkability. As a result, we exclude CLS and UCL. Instead, BSZ supports publicly verifiable openings via the *Judge* algorithm.

Results and Discussion. Based on the previous considerations, BSZ — hence, PS16 and KLAP20 — is the only suitable model for ITS services. PS16 has the shortest signatures (see Figure 4), with KLAP20 having 30% longer signatures. Concerning *Sign* and *Verify*, Table 1 and

6. <https://github.com/IBM/libgroupsig> at commit 7582c60

7. <https://github.com/google/benchmark> (v1.9.4)

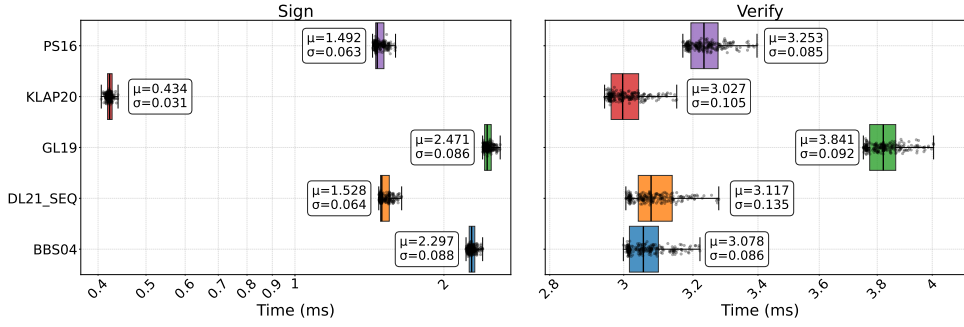


Figure 3. Sign and verify computation time per GS scheme

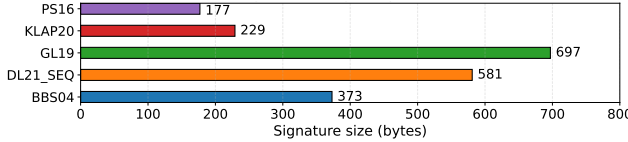


Figure 4. Signature length per scheme using curve BLS12 381.

Figure 3 show that KLAP20 has the best performance: our benchmark reports an average of 0.434ms and 3.027ms for sign and verify computations in KLAP20, respectively, and 1.492ms and 3.252ms for PS16. In summary, PS16 has shorter signatures while KLAP20 has better performance: the choice between them ultimately depends on what aspect needs to be prioritized.

5. Design

We now describe a redesign of ETSI 102 941 [18] and ETSI 103 097 [14] authentication and authorization lifecycle and message exchange sequence diagrams to (ideally, incrementally) add support to a BSZ-based GS scheme. To simplify the description of our redesign, we consider a PKI with one EA, one AA, and one MA; multiple instances of such entities across different geographical areas (and PKIs) would simply operate independently.

5.1. Architecture Mapping and Duties Separation

In BSZ, users need certificate-bound public keys for authenticating to the issuer when joining a group; in turn, this requires a PKI. Hence, we simply map BSZ’s PKI to ETSI’s PKI authority chain composed by EAs and RCAs, while BSZ user certificates correspond to ETSI’s ECs. Then, we remark that ETSI TS 102 940 [16] and 102 941 [18] require the separation of duties between the EA and AA. As a corollary, the AA must not access the EC of the ITS-S requesting authorization. Also, subsequent authorization requests made by the same ITS-S to the AA must be unlinkable by the AA. Similarly, the EA must not be able to link an AT — in our case, GS — to the corresponding EC. Given the separation of duties requirement, it may seem sensible to give the EA the opener role and the AA the issuer role in the BSZ model. However, giving opening capabilities to the EA would allow it to open any GS (i.e., AT) and link it to an EC. Also, the AA could access the EC, since the signature associated with the EC would be presented to the AA

during the execution of the *Join* algorithm to prove the ITS-S’s identity, breaking the unlinkability requirement previously discussed. On the other hand, inverting the roles — i.e., have the AA as opener and the EA as issuer — still breaks unlinkability. In fact, the EC would be shared between the EA and the AA because the AA needs read access to the identity registry to open GSs, as required by BSZ. Therefore, to satisfy the separation of duties requirement and provide unlinkability, we encrypt the identities and signatures in the register so that the EA is not able to link them to the GSs without the AA’s authorization (see step (6) *AuthorizationResponse* below).

Finally, we note that the RCA acts as both the trust anchor of the PKI and the authority responsible for running the GS *Setup* algorithm. RCAs are usually assumed to be trusted [5], hence having a single authority for both roles is not (or should not) be a concern. Nonetheless, we remark that it is surely possible to distinguish between the GS trusted authority and the PKI trust anchor.

5.2. Modifying the Lifecycle

To integrate the use of GSs, we modify three state transitions of the lifecycle shown in Figure 2, and the related messages as described in Figures 15—18 in Annex A.

- *Authorize* is mapped to *group join*: from the *enrolled and not authorized* ($enr \wedge \neg auth$) state, an ITS-S sends a join request to the AA by specifying the requested permissions and signing the request with the private key associated with its EC. Then, the AA queries the EA to validate the EC and confirms that the ITS-S can access the requested permissions. Upon a positive validation, the AA and the ITS-S execute the *Join* and *Issue* algorithms to derive the ITS-S’s secret key associated with the group with the requested permissions, and the ITS-S becomes *authorized for service* (*auth*).
- *AT expiration* is mapped to *group expiration*: as prescribed in the C-ITS certificate policy [26] for ATs, we assign a validity for groups — called *epoch* — of (at most) one week, along with an epoch identifier. Within an epoch, an ITS-S can sign messages under its epoch-bound membership; upon epoch rollover, continued authorization requires a refresh that re-validates the EC and rebinds membership to the new epoch. This design aligns with the policy that limits authorization lifetimes to weekly granularity [26].
- *Revoke* is mapped to *deregistration*: on end-of-life or compromise, subsequent attempts of an ITS-

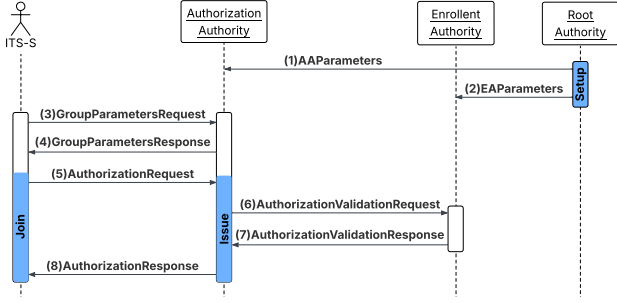


Figure 5. Group join.

S to transition back to the *authorized for service (auth)* state are blocked by the EA, which denies validation for any new join request reaching the AA, as per lifecycle management in ETSI 102 941 [18]. In case of a misbehavior report, where ETSI 102 941 [18] requires linking an AT to an EC, in our solution, the EA opens the reported GS, identifies the misbehaving ITS-S, and produces a publicly verifiable proof linking the reported GS to the ITS-S’s EC.

5.3. New Message Exchange Sequence Diagrams

Below, we discuss the message exchange sequence diagrams for the two state transitions we modify: *group join* and *deregistration (GS opening)* — no group expiration, as it is passive. We represent hybrid public-key encryption with Σ , public-key signature with Π , and symmetric encryption with Γ . We collect symbols used throughout this section in Table 2.

In the following, we use $params_{group}$ to generalize the freshness parameters that are sent by the issuer to an ITS-S that wishes to join a group — e.g., generator $g \xleftarrow{\$} G_1$ and nonce n used by PS16 and discussed in Section 6. Also, we use $params_{join_i}$ to represent the parameters computed by the ITS-S that will be used by the issuer to compute the ITS-S credential $cred_i$ and that will be included in the accountability register reg — e.g., tuple $(\tau, \tilde{\tau}, spk)$ used by PS16 and discussed in Section 6.

Group Join. Figure 5 depicts the join and issuance execution among the ITS-S, the AA, and the EA, highlighting the ITS-S identity validation (A more detailed diagram can be found in Annex A at Figure 11).

(1) **AAParameters** (RA \rightarrow AA). This message contains the AA certificate signed by the RCA, the group public key gpk and the issuer key gsk_{iss} .

(2) **EAParameters** (RA \rightarrow EA). This message contains the EA certificate signed by the RCA, the group public key gpk and the issuer key gsk_{open} .

(3) **GroupParametersRequest** (ITS-S \rightarrow AA). The ITS-S initiates the session by requesting from the AA the group parameters $params_{group}$ of a given group.

(4) **GroupParametersResponse** (AA \rightarrow ITS-S). The AA replies with the group public key gpk and parameters $params_{group}$. The requester can then compute its user parameters $params_{join_i}$ to compute its credentials $cred_i$ with the issuer collaboratively.

(5) **AuthorizationRequest** (ITS-S \rightarrow AA). For the unlinkability of multiple requests, we adopt the same

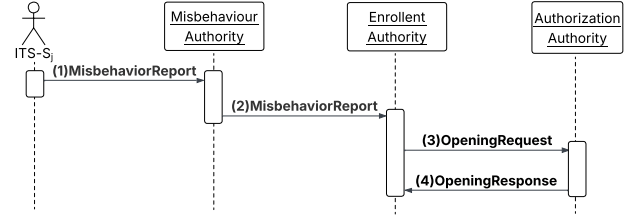


Figure 6. ITS-S signature opening.

Symbol	Description
Π	Digital signature scheme
Σ	Hybrid KEM–DEM encryption scheme
Γ	Symmetric encryption scheme
gpk	Group signature scheme public key
gsk_{iss}	Issuer key used to issue enrollment credentials
gsk_{open}	Opening key used
$params_{group}$	Public group parameters for a given group
$params_{join_i}$	Join parameters computed by ITS-S _i
$params_{join_i}^{hmac}$	Keyed HMAC of $params_{join_i}$ under k_{hmac}
k_{hmac}	Secret key used to compute $params_{join_i}^{hmac}$
$cred_i$	Enrollment credential issued to ITS-S _i on join
gsk_i	Signing key of ITS-S _i derived from $cred_i$
$pubk_i$	Public key of ITS-S _i associated with $cred_i$
$pubk_{ea}$	Public key of the EA
$pubk_i^{ea}$	$pubk_i$ encrypted under $pubk_{ea}$ using $\Sigma.Enc$
$pubk_i^{aa}$	$pubk_i^{ea}$ encrypted under $symk_{aa}$ using $\Gamma.Enc$
η_i	Signature on $(params_{join_i}, pubk_i)$ using Π
η_i^{ea}	η_i encrypted under $pubk_{ea}$ using $\Sigma.Enc$
η_i^{aa}	η_i^{ea} encrypted under $symk_{aa}$ using $\Gamma.Enc$
$symk_{aa}$	Symmetric key held by the AA for use with Γ
reg	Accountability register for issued credentials
reg_i	The issued credential in reg for ITS-S _i : $(params_{join_i}, pubk_i^{aa}, \eta_i^{aa})$
r	Misbehavior report tuple (m, σ_i)
r_j	Misbehavior report tuple (r, σ_j) sent by ITS-S _j
m	Message contained in r
m_i	Message whose GS σ_i is being opened
σ_i	GS attached to the reported message m
σ_j	GS by the reporter on the tuple r
η_{ma}	Signature of a misbehavior report r_j by the MA
$privk_{ma}$	Private signing key of the MA
π	Non-interactive proof of correct <i>Open</i> output
\top, \perp	Boolean outputs for accept (\top) or reject (\perp)

TABLE 2. TABLE OF SYMBOLS

method of ETSI TS 102 941: the public key $pubk_i$ associated with the EC and its signature $\eta_i \leftarrow \Pi.Sign(params_{join_i}^{hmac}, pubk_i)$ are encrypted for the EA using the hybrid KEM–DEM scheme Σ as $pubk_i^{ea} \leftarrow \Sigma.Enc(pubk_i, pubk_{ea})$ and $\eta_i^{ea} \leftarrow \Sigma.Enc(\eta_i, pubk_{ea})$ respectively; only the EA can recover the identity and check the validity of the request. The signature η_i is computed over the keyed HMAC $params_{join_i}^{hmac} \leftarrow HMAC(params_{join_i}, k_{hmac})$. This is necessary as this HMAC is used to bind the identity of the requesting ITS-S via its signature — see message (7) of the group join protocol — while avoiding the EA recovering its identity when accessing the BSZ register during an *OpeningRequest* — see message (3) and (4) of the protocol for the opening of GSs. Finally, the ITS-S sends the

tuple $(\eta_i^{ea}, \text{pubk}_i^{ea}, \text{params}_{\text{join}_i}, \text{params}_{\text{join}_i}^{hmac}, k_{hmac})$ to the AA — which corresponds to a group join request.

(6) **AuthorizationValidationRequest (AA → EA)**. The AA checks the validity of $\text{params}_{\text{join}_i}^{hmac}$. If valid, the AA relays $(\eta_i^{ea}, \text{pubk}_i^{ea}, \text{params}_{\text{join}_i}^{hmac})$ to the EA.

(7) **AuthorizationValidationResponse (EA → AA)**. The EA decrypts the payload, verifies the validity of the signature over $\text{params}_{\text{join}_i}^{hmac}$ as $\top \perp \leftarrow \Pi.Ver(\eta_i, \text{params}_{\text{join}_i}^{hmac}, \text{pubk}_i)$, and returns a positive decision if the requester holds a valid EC.

(8) **AuthorizationResponse (AA → ITS-S)**. On acceptance, the AA symmetrically encrypts η_i^{ea} and pubk_i^{ea} as $\eta_i^{aa} \leftarrow \Gamma.Enc(\eta_i^{ea}, \text{symk}_{aa})$ and $\text{pubk}_i^{aa} \leftarrow \Gamma.Enc(\text{pubk}_i^{ea}, \text{symk}_{aa})$ respectively. The AA then stores in the register *reg* a tuple $\text{reg}_i \leftarrow (\text{params}_{\text{join}_i}, \text{pubk}_i^{aa}, \eta_i^{aa})$ for accountability. The AA sends the requesting ITS-S its credential *cred_i* that the ITS-S uses to compute *gsk_i*.

Deregistration (GS opening). As per lifecycle management in ETSI 102 941 [18], the EA rejects transition requests to the *authorized for service (auth)* state from ITS-Ss that have reached their end-of-life. Note that deregistration may occur naturally (e.g., when a vehicle is decommissioned) and does not always entail the opening of a GS. However, if a misbehavior report requires the linking of an AT to an EC, the MA, the EA, and the AA can collaboratively open a GS — preserving the separation of duties and unlinkability — as summarized in the workflow presented in Figure 6 (A more detailed diagram can be found in Annex A at Figure 12).

(1) **MisbehaviorReport (ITS-S_j → MA)**. An ITS-S submits a report of a suspicious payload $r_j \leftarrow (r, \sigma_j)$ and contextual evidence as prescribed by ETSI TS 103 759; in our setting, σ_j is a GS computed by the reporter over the reported tuple $(r \leftarrow (m, \sigma_i))$, while *m* and σ_i are the reported message and its GS, respectively.

(2) **MisbehaviorReport (MA → EA)**. The MA authenticates the reporter and reviews the report as per ETSI TS 103 759 [17]. If an opening is needed, the MA forwards the report to the EA with its signature $\eta_{ma} \leftarrow \Pi.Sign(r_j, \text{privk}_{ma})$.

(3) **OpeningRequest (EA → AA)**. After having validated the MA request, the EA links GS σ_i to the encrypted identity, pubk_i^{aa} , and signature, η_i^{aa} , of the reported ITS-S stored in the register tuple *reg_i*. In particular, the EA computes $(\text{reg}_i, \pi) \leftarrow \text{Open}(m_i, \sigma_i, \text{gpk}, \text{gsk}_{\text{open}}, \text{reg})$ where π is a non-interactive proof of correct *Open* output. To recover (η_i, pubk_i) , the EA sends (reg_i, π) to the AA.

(4) **OpeningResponse (AA → EA)**. The AA checks the MA signature on the report to assert the request's validity, then runs $\text{Judge}(\text{gpk}, m, \sigma_i, \pi) \rightarrow \top \perp$ to validate that the EA is not trying to obtain an arbitrary identity. Upon success, the AA then sends tuple $(\eta_i^{ea}, \text{pubk}_i^{ea}, \text{params}_{\text{join}_i}, k_{hmac})$ to the EA. The EA decrypts both η_i^{ea} and pubk_i^{ea} , and computes $\text{params}_{\text{join}_i}^{hmac}$ using the received $\text{params}_{\text{join}_i}, k_{hmac}$ to be able to verify *eta_i* — i.e., the signature computed over the hashed parameters. If these checks are successful, the EA will reject any subsequent group join requests made by the reported ITS-S as per ETSI TS 102 941 [18].

As a final remark, we highlight that our redesign does not affect how interoperability across geographical areas

and PKIs is carried out, as group public keys — like ATs — are associated with a specific area, and cross-area certification does not change — that is, the establishment of trust relationships and the mechanisms for certificates validation based on the CPOC and the TLM is unchanged.

6. Implementation and Initial Evaluation

Below, we use the protocol in Section 5 to prototype an ETSI TS 102 941-compliant [18] integration of a GS scheme with the ETSI ITS authentication and authorization lifecycle. Also, we conduct a preliminary evaluation to measure the resulting performance and message sizes.

6.1. Implementation

We choose PS16 as it offers shorter signatures and preserves the interactions between ETSI's authorities. In fact, PS16 presents a single authority acting as both issuer and opener, hence we can maintain the separation of duties between the AA and the EA. We integrate *libgroupsig* PS16 implementation in C2C-Common,⁸ an open-source Java implementation of ETSI 102 941 [18] and ETSI 103 097 [14], focusing on those state transitions we modify, i.e., group join and deregistration (GS opening).

Group Join. Integrating PS16 with C2C-Common using *libgroupsig* presents two main differences from the original definition of PS16. First, as in the BSZ definition, PS16 does not require unlinkability between subsequent join requests; we solve this issue by maintaining the EC encrypted as prescribed by ETSI TS 102 941 [18] so that only the EA can read the long-term identity, preserving unlinkability. Then, the EC is stored by the AA in an encrypted form. The encryption is done by the requesting ITS-S using Elliptic Curve Integrated Encryption Scheme (EICES) with the EA public key, following the same specification as in ETSI TS 102 941 [18]. Below, we describe the message exchange in our modified ITS-S authentication and credential issuance process. The message flow is also depicted in Figure 7 (A more detailed diagram can be found in Annex A at Figure 13). A comparison between the standard and modified *AuthorizationRequest* and *AuthorizationValidationRequest* message can be found in Figures 15 and 17 Annex A:

(1) **GroupParametersRequest (ITS-S → AA)**. The ITS-S requests PS16 group *id_{group}* parameters for the target group identifier to initialize a fresh join.

(2) **GroupParametersResponse (AA → ITS-S)**. The AA returns the group public key $\text{gpk} \leftarrow (\tilde{g}, \tilde{X}, \tilde{Y})$ a generator *g* and a nonce *n*.

(3) **AuthorizationRequest (ITS-S → AA)**. The requester sends a modified *AuthorizationRequest* where field `publicKeys` is replaced by $(\tau, \tilde{\tau}, \text{spk})$ — with $(\tau, \tilde{\tau}) \leftarrow (g^{sk_i}, \tilde{Y}^{sk_i})$ and *spk* a Fiat-Shamir proof — as required by PS16's join — and proving that the requester knows the secret exponent *sk_i* consistent with the parameters sent to the AA. This means that signature η_i is now computed over the HMAC derived from parameters $(\tau, \tilde{\tau}, \text{spk})$. To preserve unlinkability from the AA, the requester encapsulates the ECDSA signature η_i

8. <https://github.com/CGI-SE-Trusted-Services/c2c-common>

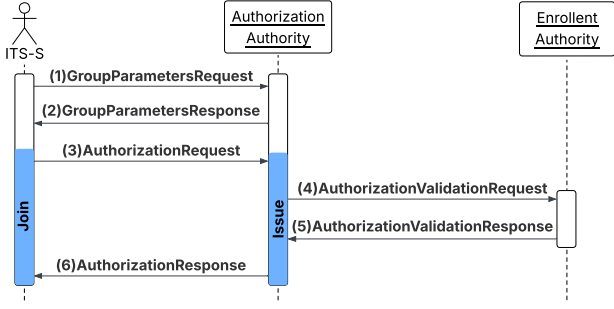


Figure 7. Group join for PS16.

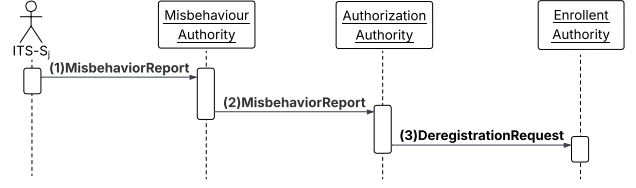


Figure 8. ITS-S deregistration request for PS16.

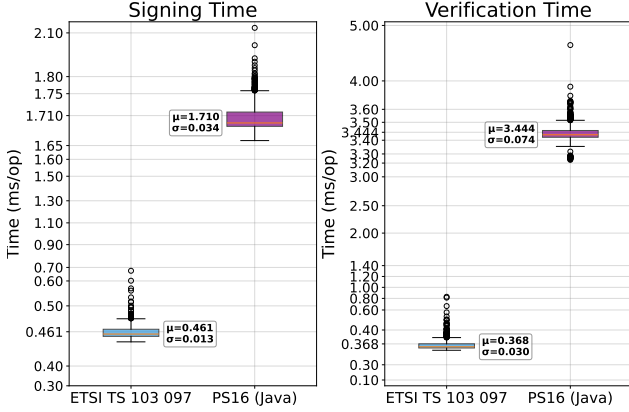


Figure 9. Sign and Verify time — C2C—common with ECDSA vs. PS16.

and the EC public key for the EA using EICES, yielding $\eta_i^{ea} \leftarrow ECIES.Enc(\eta_i, pubk_{ea})$ and $pubk_i^{ea} \leftarrow ECIES.Enc(pubk_i, pubk_{ea})$. Thus, the AA can validate $(\tau, \tilde{\tau}, spk)$ while only the EA can recover $(pubk_i, \eta_i)$.

(4) **AuthorizationValidationRequest (AA → EA)**. The AA validates the join request by checking $e(\tau, \tilde{Y}) = e(g, \tilde{\tau})$ and verifying spk , besides checking the HMAC validity. If both checks succeed, the AA relays to the EA the fields `ecSignature` — containing η_i and $pubk_i$ — and `sharedATRequest` — containing the HMAC of parameters $(\tau, \tilde{\tau}, spk)$ — in an *AuthorizationValidationRequest* message, unchanged in structure with respect to the definition in 102 941 [18].

(5) **AuthorizationValidationResponse (EA → AA)**. The EA decrypts the content of the `ecSignature` payload and verifies the requester’s signature η_i . The EA replies with an *AuthorizationValidationResponse* message; the content of this response follows the standard and confirms or denies the requester’s validity.

(6) **AuthorizationResponse (AA → ITS-S)**. If the EA response confirms validity, the AA stores the tuple $(\eta_i^{ea}, \tau, \tilde{\tau}, pubk_i^{ea})$ for accountability and opening support, and sends an *AuthorizationResponse*. Finally, the field certificate is replaced by the PS16 membership credential $\tilde{\sigma} \leftarrow (\tilde{\sigma}_1, \tilde{\sigma}_2) \leftarrow (g^u, (g^x \cdot (\tau)^y)^u)$, where (x, y) are the AA secret key components. The ITS-S then finalizes its group secret key $gsk_i \leftarrow (sk_i, \tilde{\sigma}, e(\tilde{\sigma}_1, \tilde{Y}))$.

Deregistration (GS opening). We describe the message exchange among the MA, the EA, and the AA following a misbehavior report and the consequence opening of a GS.

The message flow is also depicted in Figure 8 (A more detailed diagram can be found in Annex A at Figure 14).

(1) **MisbehaviorReport (ITS-S_j → MA)**. ITS-S_j submits a report containing payload (m_i, σ_i) , the required evidence, and the attestation (σ_i is a GS on m_i produced under the active epoch). The report is authenticated so that authorities can validate the source prior opening.

(2) **MisbehaviorReport (MA → AA)**. The MA authenticates the reporter and evaluates the evidence. Upon deciding that an opening is needed, MA forwards the report to AA, including its signature on the report — i.e., $\eta_{ma} \leftarrow ECDSA.Sign(r_j, privk_{ma})$ — so that the AA can validate the origin and integrity of the request. The AA verifies the MA’s signature and the reporter’s attestations, then executes the *Open* algorithm iterating over stored tuples $(\eta_i^{ea}, \tau, \tilde{\tau}, spk, pubk_i^{ea})$ and tests $e(\sigma_2, \tilde{g}) \cdot e(\sigma_1, \tilde{X})^{-1} \stackrel{?}{=} e(\sigma_1, \tilde{\tau})$. In a match, it produces (reg_i, π) , where π is a signature proof of knowledge that attests knowledge of $\tilde{\tau}$ consistent with the check.

(3) **OpeningRequest (AA → EA)**. The AA returns (reg_i, π) and the report context to the EA. The EA first validates the MA signature that authorized the opening request and then runs the verification of the opening proof to confirm that π binds σ_i to τ . Then, EA decrypts η_i^{ea} contained in *OpeningRequest* and verifies it to confirm that the AA is requesting the opening of the credentials associated with the reported signature and proof π . If all checks are successful, the EA will deny subsequent group joining and EC refresh requests from the holder of the EC used for producing the reported signature η_i .

Our modification of ETSI TS 102 941 [18] and 103 097 [14] satisfies all requirements listed in Section 1 using BSZ GSs with no pseudonyms. Also, we ensure that group join and deregistration (GS opening) can be executed only by both the AA and EA collaboratively, as required in ETSI TS 102 940 [16].

6.2. Initial Evaluation

We implement the PS16 protocols for group join and deregistration (GS opening) using `libgroupsig` and `C2C-Common`. We then perform preliminary benchmarks to measure the time required to sign and verify a DENM in ETSI TS 103 097 [14] and our PS16 protocols. We reuse the same experimental setup as in Section 4.

As shown in Figure 9, signing and verifying DENMs with PS16 is, respectively, around 3.7× and 10× slower than ETSI TS 103 097 implementation [14] (which

uses ECDSA over `secp256r1`); DENM sizes are instead comparable (282B with PS16 vs. 284B with ECDSA).

As a final remark, the opening procedure in BSZ GS schemes lets the opener iterate over the register to find the unique join tuple that satisfies the scheme’s matching predicate with a given signature. Although this test requires the opener key and does reveal long-term identities, an honest-but-curious opener could iterate over the register and deterministically match any observed signature to the per-member tuple created at join time. As such a tuple is fixed for a member under a given group public key, repeated openings resolve to the same register entry, allowing the opener to link all messages produced by the same ITS-S. In other words, our integration with PS16 ensures unlinkability between GSs and authorization requests, but an honest-but-curious opener could still link all the signatures of the same ITS-S.

7. Further Improvements

As said in Section 2, ATs carry permission codes — i.e., ITS-AID and SSP — authorizing the holder ITS-S to send a given message type. However, switching from ATs to GS eliminates such permission codes. Hence, we need to modify the group-join protocol of Section 5 to reintroduce authorization. A naive solution could be to map each (ITS-AID, SSP) pair to a distinct group. However, ETSI TS 102 965 [19] alone defines 20 ITS-AID — plus those reserved for authorities — and each ITS service further specifies its SSP set for future expansions. Intuitively, the naive solution is not scalable and would incur heavy group management. To aggregate permission codes more effectively, we rely on the C-Roads harmonization of interoperable European ITS services within ETSI standards. In detail, the C-Roads use-case definition [26] specifies, for each use case, involved entities and associated (ITS-AID, SSP) pairs. For example, in the public-transport-vehicle-crossing (HLN-PTVC) use case, the public transport operator may transmit CAMs with vehicle role set to 1 and DENMs with SSP set to 97. Hence, to reduce the number of groups, we can instantiate them by use case and role. In this example, rather than creating separate groups for DENMs and CAMs with the required SSP codes, we can instantiate a single group governed by the policy $HLN_PTVC \wedge PUBLIC_TRANSPORT$. Generalizing the example, we encode policies as attributes with ABE — more precisely, Ciphertext-Policy ABE (CP-ABE) — and build an authorization mechanism by modifying the protocol shown in Section 5 to include the CP-ABE-based challenge standardized by ETSI in 103 964 [20]. Essentially, together with the EC, the EA sends to ITS-Ss a CP-ABE key sk_{attr} , where the attributes $attr$ represent the ITS-S role and use cases for which it can send messages. In fact, since ETSI TS 102 941 [18] assigns to the EA the role of accepting or refusing authorization requests, we map the EA to the trusted authority in ABE, while the AA enforces authorization via delegated attributes. Concretely, we modify the following messages from the protocol in Section 5 (see Figure 10):

(4) **GroupParametersResponse** (AA \rightarrow ITS-S). When the ITS-S requests to access a group id_{group} , the AA responds with a challenge $c_{policy} \leftarrow CPABE.Enc(n, pubk_{abe}, policy_{group})$ —

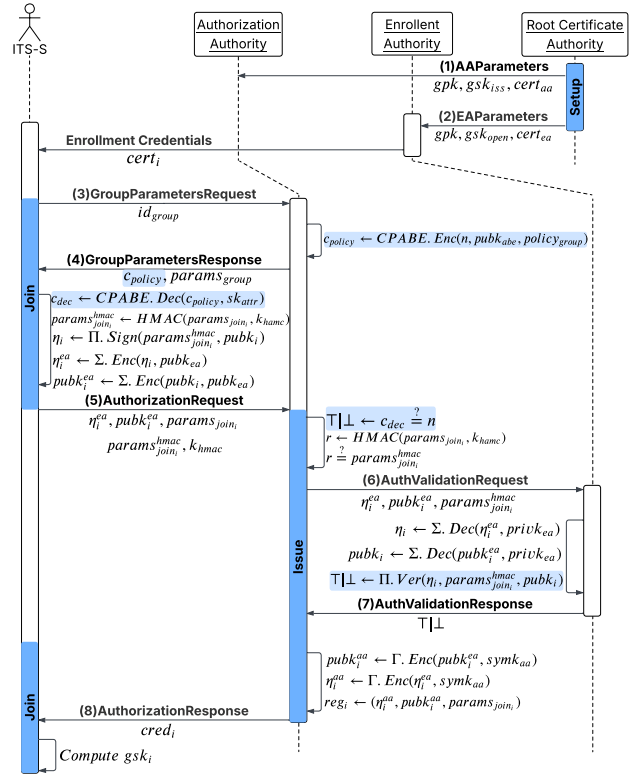


Figure 10. Group join with CP-ABE challenge.

n is a nonce, $policy_{group}$ is the access policy of the group by sk_{attr} , and $pubk_{abe}$ is the CP-ABE public key.

(5) **AuthorizationRequest** (ITS-S \rightarrow AA). The ITS-S includes the decrypted challenge $c_{dec} \leftarrow CPABE.Dec(c_{policy}, sk_{attr})$. If $c_{dec} = n$, the authentication process continues as described in Section 5. Otherwise, the ITS-S is denied access to the requested group.

8. Conclusion and Future Work

We proposed a standard-compatible redesign of the ETSI lifecycle to replace pseudonymous certificates with certificateless authorization using BSZ GSs, satisfying all requirements as well as separation of duties between EAs and AAs. Also, we presented a proof-of-concept based on C2C-Common and IBM `libgroupsig` and conducted a preliminary evaluation against an implementation compliant with ETSI TS 103 097:⁹ signing and verification overhead is notable (3.7× and 10×) but can be reduced by using more performant BSZ GS schemes (e.g., K LAP20) and, especially, introducing batch verification of GSs. Finally, we motivated a group-based authorization model with ABE grounded in C-Roads use cases and roles.

Future Work. We plan to formally verify the security of the protocols in Section 5, conduct further benchmarks using OBU-class processors, mitigate the linkability concern due to honest-but-curious openers by using secure multi-party computation in the *Open* algorithm, and add a batch verifier for GSs, mapping it with an ETSI entity like RSUs or a new one such as a Cloud broker as in [6].

9. The code of the proof-of-concept and all experimental results are available at <https://github.com/anonAccount99/ps16-etsi-pki>.

References

- [1] Maria Azees, Pandi Vijayakumar, and Lazarus Jegatha Deboarh. EAAP: Efficient anonymous authentication with conditional privacy-preserving scheme for vehicular ad hoc networks. *IEEE Transactions on Intelligent Transportation Systems*, 18(9):2467–2476, February 2017.
- [2] Lina Bariah, Dina Shehada, Ehab Salahat, and Chan Yeob Yeun. Recent advances in vanet security: A survey. In *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)*, pages 1–7, 2015.
- [3] Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *Advances in Cryptology — EUROCRYPT 2003*, pages 614–629, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- [4] Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In Alfred Menezes, editor, *Topics in Cryptology – CT-RSA 2005*, pages 136–153, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
- [5] Stefano Berlato, Marco Centenaro, and Silvio Ranise. Smart card-based identity management protocols for v2v and v2i communications in CCAM: A systematic literature review. *IEEE Transactions on Intelligent Transportation Systems*, 23(8):10086–10103, 2022.
- [6] Stefano Berlato, Silvio Cretti, Domenico Siracusa, and Silvio Ranise. Multi-objective microservice orchestration: Balancing security and performance in ccam. In *2024 27th Conference on Innovation in Clouds, Internet and Networks (ICIN)*, pages 88–90, 2024.
- [7] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matt Franklin, editor, *Advances in Cryptology – CRYPTO 2004*, pages 41–55, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.
- [8] Simone Bussa, Riccardo Sisto, and Fulvio Valenza. Formal verification of a v2x scheme mixing traditional pki and group signatures. *Journal of Information Security and Applications*, 89:103998, 2025.
- [9] C-Roads Platform. The C-Roads platform: An overview of harmonised C-ITS deployment in europe. Technical report, Austriatech, 2021.
- [10] Marco Centenaro, Stefano Berlato, Roberto Carbone, Gianfranco Burzio, Giuseppe Faranda Cordella, Roberto Riggio, and Silvio Ranise. Safety-related cooperative, connected, and automated mobility services: Interplay between functional and security requirements. *IEEE Vehicular Technology Magazine*, 16(4):78–88, 2021.
- [11] David Chaum and Eugène van Heyst. Group signatures. In *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer, 1991.
- [12] Jesus Diaz and Anja Lehmann. Group signatures with user-controlled and sequential linkability. *Cryptology ePrint Archive*, Paper 2021/181, 2021.
- [13] ISO/TC 22/SC 32 Electrical, electronic components, and general system aspects Committee. Road vehicles — cybersecurity engineering. Standard, SAE International - Vehicle Cybersecurity Systems Engineering Committee, February 2021.
- [14] ETSI. (ITS); security; security header and certificate formats; release 2. Technical Report TS 103 097, ETSI, 2021. V2.1.1.
- [15] ETSI. (ITS); basic set of applications; decentralized environmental notification service; release 2. Technical Report TS 103 831, ETSI, 2022. V2.2.1.
- [16] ETSI. (ITS); its communications security architecture and security management; release 2. Technical Report TS 102 940, ETSI, 2022. V2.2.1.
- [17] ETSI. (ITS); security; misbehaviour reporting service; release 2. Technical Report TS 103 759, ETSI, 2022. V2.2.1.
- [18] ETSI. (ITS); security; trust and privacy management; release 2. Technical Report TS 102 941, ETSI, 2022. V2.2.1.
- [19] ETSI. (ITS); application object identifier (its-aid); registration; release 2. Technical Report TS 102 965, ETSI, 2024. V2.2.1.
- [20] ETSI. Cyber security (cyber); a verifiable credentials extension using attribute-based encryption. Technical Report TS 103 964, ETSI, 2025. V1.1.1.
- [21] ETSI. (ITS); facilities layer; cooperative awareness service; release 2. Technical Report TS 103 900, ETSI, 2025. V2.2.1.
- [22] Lydia Garms and Anja Lehmann. Group signatures with selective linkability. *Cryptology ePrint Archive*, Paper 2019/027, 2019.
- [23] Jinhua Guo, John P. Baugh, and Shengquan Wang. A group signature based secure and privacy-preserving vehicular communication framework. In *2007 Mobile Networking for Vehicular Environments*, pages 103–108, 2007.
- [24] IEEE. Ieee standard for wireless access in vehicular environments—security services for application and management messages. *IEEE Std 1609.2-2022 (Revision of IEEE Std 1609.2-2016)*, pages 1–349, 2023.
- [25] Hyoseung Kim, Youngkyung Lee, Michel Abdalla, and Jong Hwan Park. Practical dynamic group signature with efficient concurrent joins and batch verifications. *Cryptology ePrint Archive*, Paper 2020/921, 2020.
- [26] Tink Cryptographic Library. Certificate policy for deployment and operation of european cooperative intelligent transport systems(c-its) release 3.0, 2024. Accessed: 2025-08-14.
- [27] Kiho Lim, Kastuv M. Tuladhar, Xiwei Wang, and Weihua Liu. A scalable and secure key distribution scheme for group signature based authentication in vanet. In *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*, pages 478–483, 2017.
- [28] Sheraz Mazhar, Abdur Rakib, Lei Pan, Frank Jiang, Adnan Anwar, Robin Doss, and Jeremy Bryans. State-of-the-art authentication and verification schemes in vanets: A survey. *Vehicular Communications*, 49:100804, 2024.
- [29] Boubakeur Moussaoui, Nouredine Chikouche, and Hacène Fouchal. An efficient privacy scheme for c-ITS stations. *Computers and Electrical Engineering*, 107:108613, 2023.
- [30] NIST. Guideline for Using Cryptographic Standards in the Federal Government: Cryptographic Mechanisms. Technical Report 800-175B, NIST, March 2020. Last accessed March 2025.
- [31] David Pointcheval and Olivier Sanders. Short randomizable signatures. *Cryptology ePrint Archive*, Paper 2015/525, 2015.
- [32] Yusuke Sakai, Jacob C. N. Schuldt, Keita Emura, Goichiro Hanaoka, and Kazuo Ohta. On the security of dynamic group signatures: Preventing signature hijacking. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *Public Key Cryptography – PKC 2012*, pages 715–732, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
- [33] Mahmoud A. Shawky, Syed Tariq Shah, Mohammed Abdrabou, Muhammad Usman, Qammer H. Abbasi, David Flynn, Muhammad Ali Imran, Shuja Ansari, and Ahmad Taha. How secure are our roads? an in-depth review of authentication in vehicular communications. *Vehicular Communications*, 47:100784, 2024.
- [34] Pranav Kumar Singh, Shivram N Gowtham, Tamilselvan S, and Sukumar Nandi. CPESP: Cooperative pseudonym exchange and scheme permutation to preserve location privacy in VANETS. *Vehicular Communications*, 20:100183, 2019.
- [35] A. Wasef and X. Shen. Efficient group signature scheme supporting batch verification for securing vehicular networks. In *2010 IEEE International Conference on Communications*, pages 1–5, 2010.
- [36] Lingbo Wei, Jianwei Liu, and Tingge Zhu. On a group signature scheme supporting batch verification for vehicular networks. In *2011 Third International Conference on Multimedia Information Networking and Security*, pages 436–440, 2011.
- [37] Chunhua Zhang, Xiaoping Xue, Lijuan Feng, Xin Zeng, and Jingxiao Ma. Group-signature and group session key combined safety message authentication protocol for vanets. *IEEE Access*, 7:178310–178320, 2019.
- [38] Lei Zhang, Qianhong Wu, Agusti Solanas, and Josep Domingo-Ferrer. A scalable robust authentication protocol for secure vehicular communications. *IEEE Transactions on Vehicular Technology*, 59(4):1606–1617, 2010.

Appendix A. Detailed Message Exchange Sequence Diagrams and Message Structures

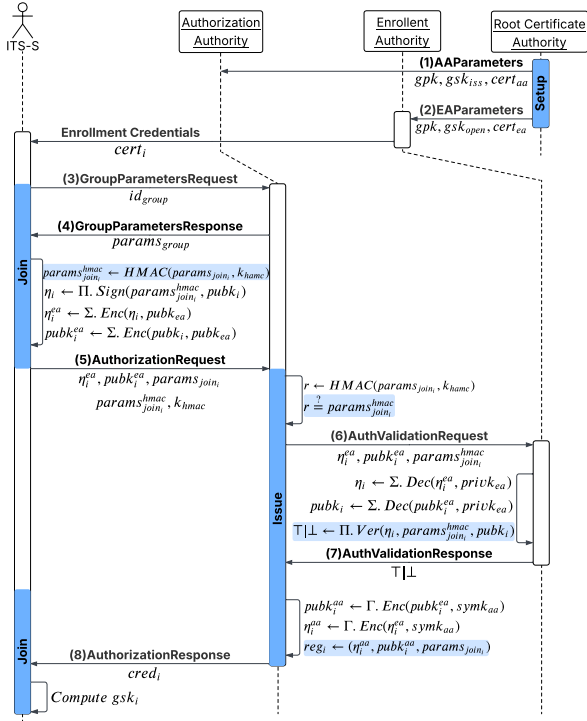


Figure 11. Group join message exchange sequence diagram. With Σ we represent a Hybrid Public-Key Encryption scheme, Π is a public-key signature scheme, while Γ represents a symmetric encryption scheme.

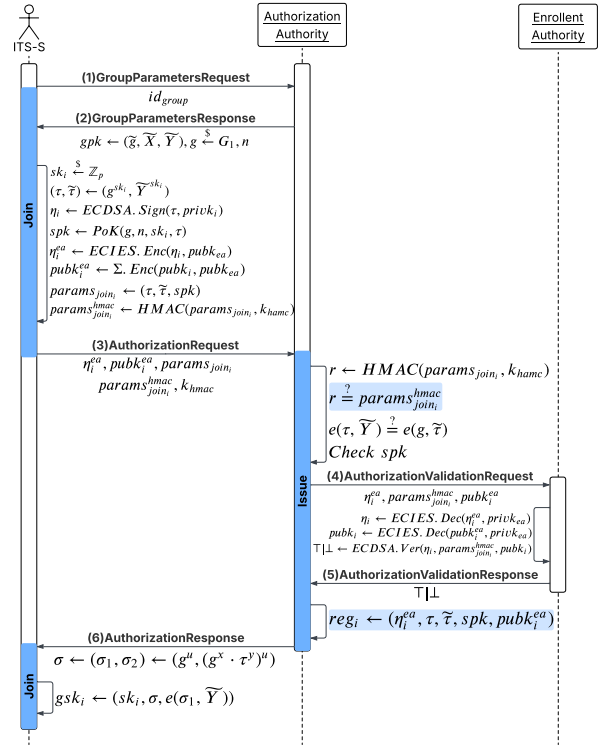


Figure 13. Group join message exchange sequence diagram for PS16 integration.

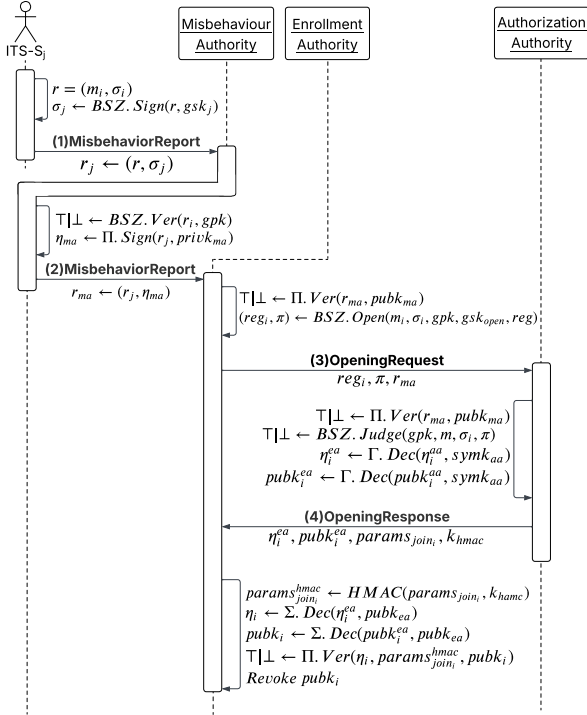


Figure 12. GS opening sequence message exchange diagram. With Σ we represent a Hybrid Public-Key Encryption scheme, Π is a public-key signature scheme, while Γ represents a symmetric encryption scheme.

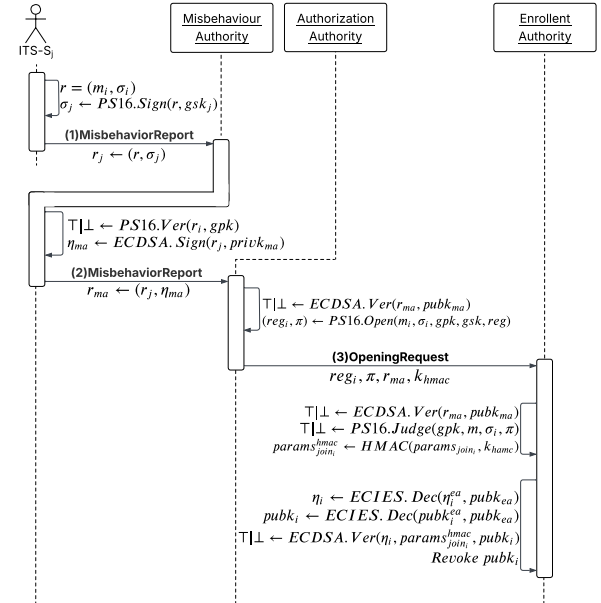


Figure 14. ITS-S GS opening request message exchange sequence diagram for PS16 integration.

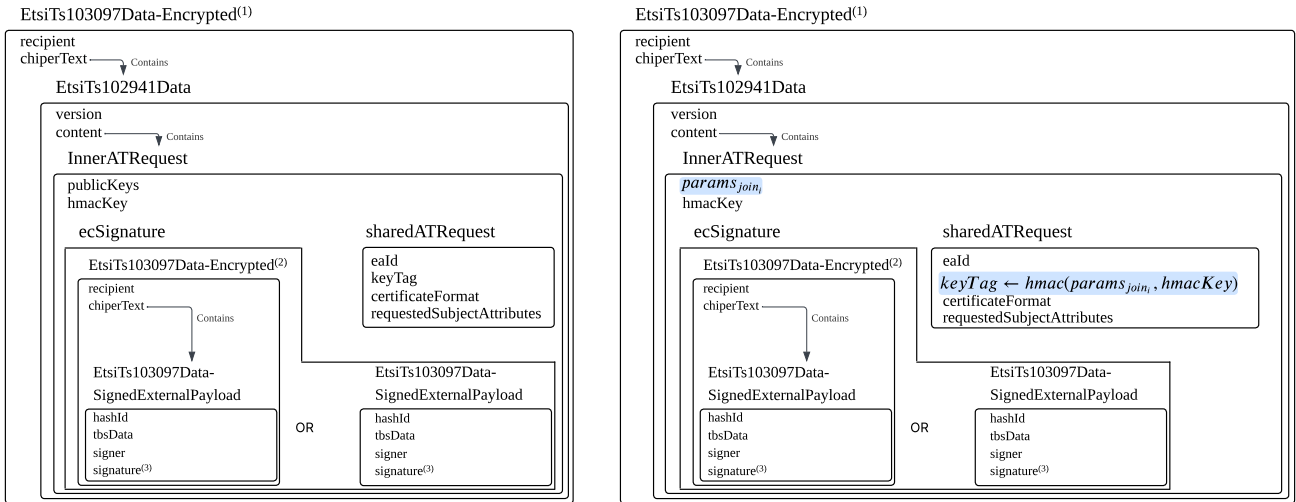


Figure 15. Comparison between the AuthorizationRequest message in ETSI TS 102 941 (left), and in our modification (right). (1) The message is encrypted with ECIES using the AA public key. (2) Payload of ecSignature can be encrypted with ECIES using the EA public key to ensure the unlinkability of multiple requests and to not expose the identity of the requestor to the AA. (3) The signature is computed over the first 128 bit of the hash of payload sharedATRequest.

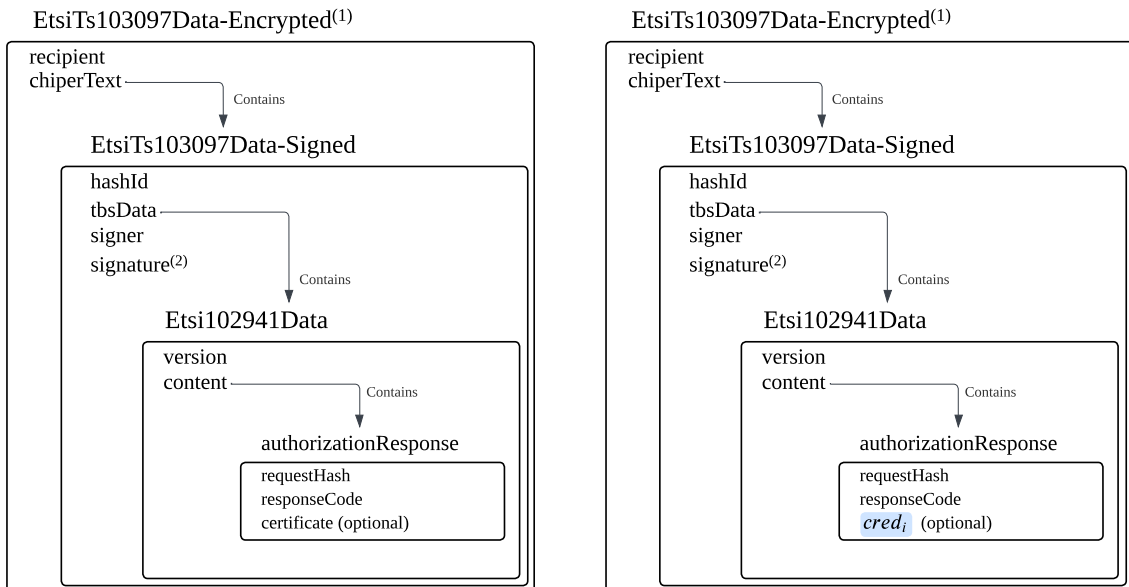


Figure 16. Comparison between the AuthorizationResponse message in ETSI TS 102 941 (left), and in our modification (right). (1) The message is encrypted with AES using the previously negotiated using ECIES. (2) The message is signed using the private gsk_i key associate with the AA certificate. The modification consists in sending the user credential $cred_i$ used to compute the user group secret key gsk_i .

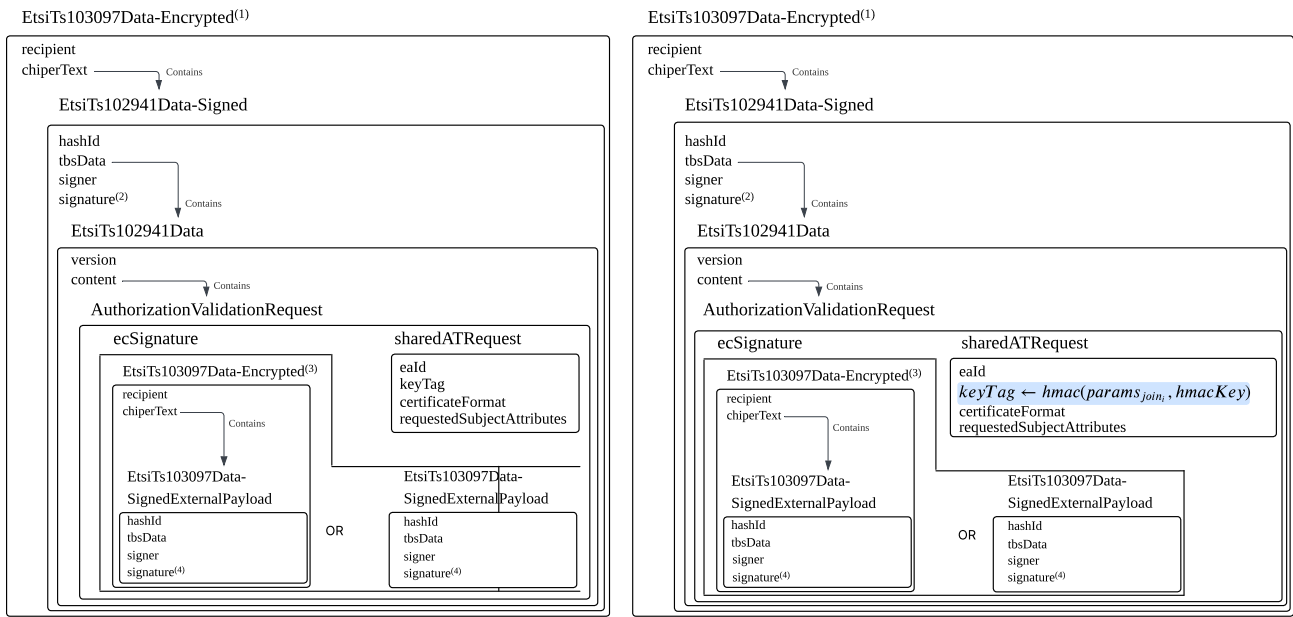


Figure 17. Comparison between the AuthorizationValidationRequest message in ETSI TS 102 941 (left), and in our modification (right). (1) The message is encrypted with ECIES using the EA public key. (2) The message is signed using the private key associate with the AA certificate. (3) Payload of ecSignature can be encrypted with ECIES using the EA public key. (4) The signature is computed over the first 128 bit of the hash of payload sharedATRequest. This message purpose is to relay ecSignature and sharedATRequest payloads to the EA.

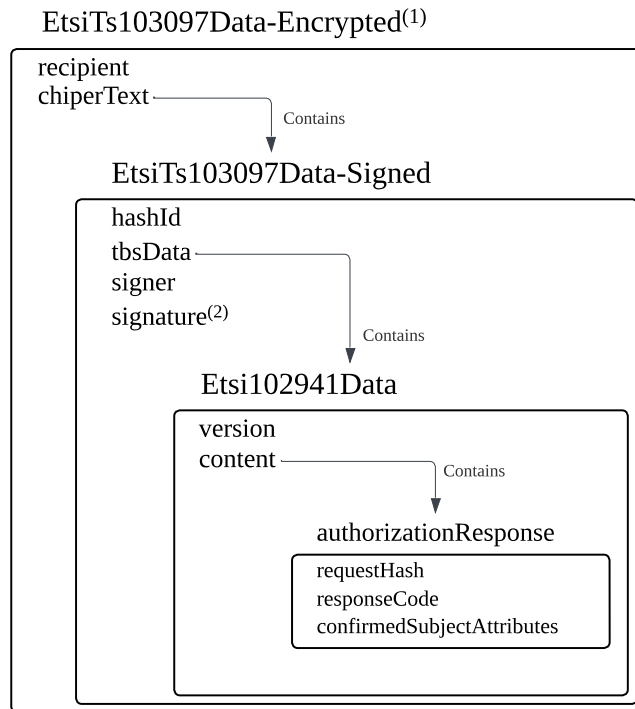


Figure 18. Comparison between the AuthorizationValidationResponse message in ETSI TS 102 941. This message is not modified.