POLITECNICO DI TORINO
Repository ISTITUZIONALE

Formal verification of a V2X scheme mixing traditional PKI and group signatures

Original
Formal verification of a V2X scheme mixing traditional PKI and group signatures / Bussa, S., Sisto, R., Valenza, F.. - In: OPEN JOURNAL OF INFORMATION SECURITY AND APPLICATIONS. - ISSN 2374-6262. - 89:(2025). [10.1016/j.jisa.2025.103998]

Availability:
This version is available at: 11583/2997222 since: 2025-02-05T16:28:41Z

Publisher:
Elsevier

Published
DOI:10.1016/j.jisa.2025.103998

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

28 November 2026

Journal of Information Security and Applications 89 (2025) 103998

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Formal verification of a V2X scheme mixing traditional PKI and group signatures

Simone Bussa *, Riccardo Sisto, Fulvio Valenza

Dipartimento di Automatica e Informatica, Politecnico di Torino, Torino, Italy

ARTICLE INFO

Keywords:
V2X communications
Formal verification
Proverif

ABSTRACT

Vehicle-to-Everything (V2X) communications are expected to reshape road mobility in the increasingly near future. This type of communication allows a vehicle to transmit information, such as its position and speed, which can be used for different applications. However, despite the benefits, the increased connectivity and data sent over the network may expose the vehicle to a significant number of cyber attacks. This paper takes one of the schemes proposed in the literature to protect the security and privacy of the vehicles, and analyses it from a security and privacy perspective using Proverif. Specifically, this scheme is unique in combining asymmetric encryption with digital certificates and group signatures used by vehicles to self-certify those certificates. We present a formal model able to capture all the main aspects of the protocol and the context in which it works, and show how security and privacy properties can be expressed for formal verification in Proverif. Our analysis conducted on the model of the protocol revealed some weaknesses for which we tried to provide a solution.

1. Introduction

Intelligent Transportation Systems (ITSs) represent a great revolution in the field of road mobility, aimed at improving the safety, efficiency, and sustainability of the actual transportation network [1]. As a cornerstone technology for ITS is Vehicle-to-Everything (V2X) communication, a novel type of interaction that enables real-time message exchange between moving vehicles, infrastructure and other entities involved (e.g. pedestrians, networks, etc.) [2]. Typically, V2X messages contain vehicle runtime data, such as speed, position, and direction of movement, which are sent in broadcast to all surrounding neighbours. This data can support and complement the information collected by the vehicle from the surrounding environment using sensors and cameras [3]. The benefits and applications that could be derived are many, and they have been described very well in the literature [4,5]. As a direct consequence, the ITS market growth is forecast to accelerate strongly in the coming years, driven by new business opportunities. According to an analysis by Grand View Research, [6], the demand for V2X-equipped vehicles is expected to increase from the current global 3.5 million to 100 million in 2030.

Besides the benefits, however, V2X bring with them a large number of challenges. First among them is cybersecurity. The increase in connectivity results in greater exposure of vehicles to cyber-attacks [7]. Since V2X messages could contain sensitive information and the environment in which they are exchanged is safety-critical, compromising this data could lead to disastrous consequences, with the potential risk of loss of lives. Therefore, mechanisms are needed to ensure message authentication and integrity (with confidentiality as another optional feature). Furthermore, even privacy is important [8]. The information contained in V2X messages must not expose the vehicle's driver to privacy issues. For example, an attacker should not be able to link messages sent by the same vehicle and track its driver's position.

The challenge stems from the fact that V2X networks significantly differ from traditional ones (e.g. networks with a fixed infrastructure and strong security capabilities). Some of their critical aspects are: (i) The V2X environment is highly dynamic, with new nodes joining or leaving communications at every moment. (ii) There are strict communication latency requirements; therefore, security must add as little overhead as possible. (iii) Bandwidth is usually limited, so it is necessary to limit the number of messages exchanged and their size. (iv) Some privacy and security requirements seem to contradict each other. For example, a vehicle must be anonymous, but some sort of accountability is required to allow vehicle revocation in case of misbehaviour.

This gives rise to the need for new solutions. Mixing security with privacy is not trivial. For example, one can think of using digital signatures to ensure authentication and message integrity and a PKI for managing digital certificates. However, any real identifier of the vehicle must be removed from the certificate to avoid privacy issues (i.e., anonymity). The vehicle should be assigned an anonymous

* Corresponding author.
E-mail addresses: simone.bussa@polito.it (S. Bussa), riccardo.sisto@polito.it (R. Sisto), fulvio.valenza@polito.it (F. Valenza).

https://doi.org/10.1016/j.jisa.2025.103998

Available online 14 February 2025
2214-2126/© 2025 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

identity and certificate. Authorities should also maintain a mapping between the real vehicle and its anonymous identity or certificate to achieve accountability, and for vehicle revocation purposes in case of problems with the law. Additionally, a single anonymous identity is not enough for a vehicle. There is the risk that the privacy of the vehicle is simply shifted to the untraceability of its single anonymous identity. In that case, an attacker which successfully links the vehicle real identity and the anonymous one can break all the desired security properties. Multiple messages sent by the same vehicle should, therefore, not be linkable to each other (i.e., unlinkability of messages sent by the same anonymous vehicle). Thus, they must be signed using different certificates. The vehicle must consequently have multiple anonymous identities and change them frequently to avoid linkability. In doing so, it would be linkable only for a short time as long as it uses the same certificate to sign the messages.

The standard way to ensure all these properties is through the use of pseudonyms, i.e., anonymous identities that the vehicle can use to participate in the protocols to satisfy both security and privacy properties. A number of schemes have been proposed over the years to provide vehicles with pseudonyms, supported, when possible, by existing road infrastructure. These schemes can be classified according to the type of architecture and cryptographic mechanisms they use. For example, [9] classifies them into four categories: asymmetric encryption schemes (which use pseudonym digital certificates issued by a CA), identity-based cryptography schemes (in which public keys are derived from a vehicle anonymous ID), group signature schemes (with each vehicle that shares group keys within a group of vehicles), and finally symmetric encryption schemes. Each scheme has its own advantages and disadvantages, most often complementary among the various types of schemes (e.g., the disadvantages of asymmetric encryption schemes could be the strength of the group signature ones).

In this paper, we formally analyse an architecture proposed in the literature, [10], from a security and privacy perspective. The analysed scheme is important because it is the only one existing in the literature that combines asymmetric encryption and group signatures, aiming to take advantage of both. In particular, vehicles use pseudonym digital certificates that are not issued by a CA, but are self-certified using shared group keys. This paper extends an earlier conference work, [11], in which we conducted a preliminary analysis of some properties that the scheme in [10] must satisfy. In this paper, using formal verification and Proverif as the automatic verification tool, we define additional security properties and extend the model and formal analysis to exhaustively verify them as well. For each considered property, we show in a more comprehensive way how it can be represented and tested in a formal abstract model, and discuss the verification results in detail. The analysis performed on the model revealed some critical issues for some of the considered properties, which could affect not only the modelled scheme, but also other V2X architectures proposed in the literature, as well as those defined by Standards Developing Organizations (SDOs) in the existing standards. For those, then, as a further contribution of this paper, we discuss possible solutions that can be adopted to mitigate the problem or challenges that are still open to solutions.

The structure of the paper is as follows. Section 2 presents a background on V2X communications and a brief description of the schemes proposed in the literature to support them. Then, it summarises some related work regarding the analysis of their security and privacy properties. Section 3 describes the V2X scheme we verified in this paper. Section 4 shows how we modelled the scheme in Proverif. Section 5 formalises the security and privacy properties that the scheme must fulfil and shows how they can be expressed in Proverif. Finally, Section 6 discusses the verification results, and Section 7 the conclusions and future work.

2. Background and related work

V2X communications consist of messages transmitted by vehicles to all their surrounding neighbours and to road infrastructure. The structure of the messages and the information they contain have been standardised and are described in CAM [12] and DENM [13] for Europe, and BSM [14] for the USA. This is independent of the technology used to implement the communications, which has not yet been fully determined (IEEE 802.11p, cellular, hybrid approaches), [15]. SDOs, such as IEEE, ETSI and SAE, have also been struggling in recent years to define a security architecture that can assist V2X communications. They all seem to have agreed on a PKI-assisted asymmetric encryption architecture with digital signatures and the distribution of pseudonym certificates to vehicles. ETSI in the EU and IEEE+SAE in the USA have defined the corresponding schemes respectively in [16,17]. For more information and for a summary of active standards, see [18]. However, despite the standardisation effort, V2X schemes based on pure asymmetric encryption have been proven in the literature to still present unresolved issues [9,15], which may severely restrict their adoption and implementation.

From this perspective, in the next subsection, we describe in more detail the generic V2X asymmetric encryption scheme, focusing on its advantages and disadvantages, and the approach based on group signatures, to allow the reader better understand what benefits might derive from a hybrid approach that combines the two, as the one described in [10] and subject of our analysis. Finally, we summarise some related work that has been done concerning the security verification of such schemes.

2.1. Asymmetric encryption schemes

This type of scheme is based on traditional public-key encryption with digital certificates. It uses pairs of mathematically related keys: a private key, kept confidential by a node, and a public key, distributed among all nodes. With the private key, the node can sign messages to authenticate itself. With the public key, other nodes can verify the signature. Typically, the public key is inserted in a digital certificate, which links the key to some identifier of the node.

In this scheme, an enrolment certificate is issued to the vehicle during manufacturing. This enrolment certificate contains the real identity of the vehicle, so it cannot be used directly for V2X communications. In contrast, it is used by the vehicle, together with the corresponding private key, to authenticate and request pseudonym certificates from the PKI, usually more than one at a time, that are used, in turn, to sign V2X messages. Pseudonym certificates do not contain any vehicle identifying information and are usually valid for a limited period of time, after which the vehicle has to change the currently active certificate and move to the next one. When sending a V2X message to its surrounding neighbours, the vehicle uses the private key associated with the pseudonym certificate currently in use to sign the V2X message, and attaches the pseudonym certificate to the signed message. The vehicle that receives the message verifies the signature using the public key contained in the pseudonym certificate. Pseudonym certificates work like any traditional digital certificate. There is a Public Key Infrastructure (PKI) to assist the process, which usually coincides with Road Side Units (RSUs) placed at the sides of the road. When managing vehicle requests, the RSU stores the association vehicle real identity with the issued pseudonyms, to handle vehicle revocation in case of misbehaviour. Revocation is done by means of Certificate Revocation Lists (CRLs) distributed among vehicles.

Advantages. Asymmetric encryption and PKIs are well-established practices. Digital signatures and related signature verifications are fast operations, falling within the strict latency requirements imposed on V2X communications.

Disadvantages. The main disadvantage is the so-called pseudonym refill problem: the vehicle has to continuously request new pseudonym

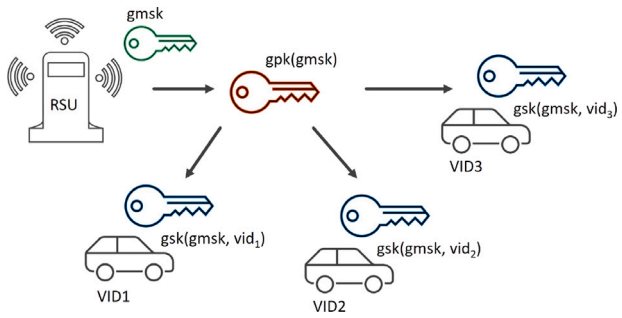

Fig. 1. Group signature general scheme.

certificates from the CA, as it consumes the ones in its possession. This requires high band consumption and constant connectivity with the CA (which is not always possible). Another significant problem is Pseudonym revocation: when a vehicle is revoked, all pseudonym certificates that have been issued to that vehicle must be revoked as well. This can cause scalability issues and high band consumption when distributing the CRL among the vehicles.

2.2. Group signatures scheme

This type of scheme uses group signatures. They are a particular type of cryptography in which there is a single public key associated with multiple private keys. All the signatures done by any of the private keys are verified using the same public key, and it is impossible starting from a signature to discover which private key performed it (which provides anonymity within the group). Moreover, multiple signatures done using the same key cannot be linked together (which provides unlinkability).

A general group signature scheme is the one shown in Fig. 1. Vehicles can form groups based on their geographic location. In this case, there is a Road Side Unit (RSU) that acts as a group manager and has a group master secret key, $gmsk$. The $gmsk$ is used to generate the keys for the group. In particular, it can generate a shared group public key, $gpk(gmsk)$, and multiple private keys to be assigned to each vehicle, combining the $gmsk$ and the vehicle id, $gsk(gmsk, vid)$. A vehicle uses the gsk to sign V2X messages to send to other vehicles, and other vehicles can use the shared gpk to verify the signature. Only the RSU, which knows the $gmsk$, starting from a signature can discover the specific gsk used for it, performing an operation called *open*. This can be used for accountability.

Advantages. The main advantage is that there is no need for the generation, delivery, or storage of pseudonym certificates to be sent together with the signed message, and to be changed frequently. Vehicles need only a public–private group key pair and have to request new group keys only when they change group. Therefore, the pseudonym refill problem no longer exists. Moreover, no certificate must be sent attached to the signed V2X messages, and this consumes less bandwidth. Another main advantage concerns vehicle revocations. In this type of scheme, when a vehicle is revoked, the RSU updates the $gmsk$ and notifies all vehicles to request new group keys. Internally, the RSU maintains a list of revoked vid, so that the revoked vehicle will not be issued updated keys. Therefore, the Revocation list is shared only among RSUs and it is not distributed to vehicles.

Disadvantages. Group signatures are complex operations that require a lot of time to be executed (around 50 ms). Because of this big overhead, it is generally agreed that a scheme that requires a group signature and its verification for each message is not compatible with the latency requirements imposed on V2X. Consequently, group signatures could be used for V2X communications only with schemes that do not require a signature and its verification for each message.

Table 1
V2X requirements taken from [8].

Basic system requirements	Security requirements
Real-time constraints	Authentication
Robustness	Accountability
Scalability	Restricted credential usage
Communications support	Credential revocation
Privacy requirements	
Minimum disclosure	
Distributed resolution	
Anonymity	
Unlinkability	
Perfect forward privacy	

2.3. Related work

In the literature, V2X communications have been subject to extensive security assessment. Many papers have been published to describe their potential security and privacy issues. Some examples, already mentioned in the Introduction, are [4,7]. Both these works seek to identify vulnerabilities and attacks that may affect V2X communications, as well as possible solutions to solve the issues encountered.

Our work relates to these papers since it aims to analyse the security and privacy of V2X communications. However, it achieves this goal by using formal verification. To the best of our knowledge, formal verification has never been applied in the literature to verify an overall architecture supporting V2X. We only found two works that are quite related, [19,20]. However, [19] focuses only on the revocation phase of a V2X scheme. While [20] verifies a protocol for electric vehicle charging. Although it can be considered a sort of V2I (vehicle-to-infrastructure), it operates in a context very different from the one analysed in this paper, in which multiple vehicles exchange messages in a highly dynamic environment. The use of formal verification to analyse the overall V2X interactions, therefore, stands out as a major contribution of this paper, differentiating it from the work that has been published and mentioned above.

This paper extends an earlier conference work, [11], by adding additional properties and enhancing the Proverif model used to verify the V2X scheme described in [10]. In particular, concerning V2X requirements, we referenced the paper [8]. It splits the requirements into three branches, as shown in Table 1. We used the properties listed in the table as a starting point for conducting our formal verification in Proverif. Other similar documents describing security and privacy requirements for V2X communications are [21,22]. Different SDOs took these papers as a reference to standardise the properties for their V2X schemes. For example, ETSI standardised them in ETSI-TS 102 941 [23], while IEEE included both requirements and architecture in the standard 1609.2 [24].

For the formalisation of security properties, instead, we took inspiration from numerous works in the literature that show how security and privacy can be analysed using formal verification. In particular, [25] shows models for the most common network security properties. [26,27], on the other hand, refer specifically to privacy properties.

3. Protocol scheme

This section describes the V2X scheme that we modelled and formally verified using Proverif. The scheme is taken from [10], and it combines asymmetric encryption with group signatures. In particular, it adopts traditional pseudonym certificates based on public-key asymmetric encryption to sign V2X messages, and group signatures used by vehicles to generate and self-certify these certificates. Since the reference scheme lacks detail when describing some key steps of the protocol, we added these details to what is presented in [10] to get a formal specification of the overall architecture that is complete and verifiable.

3.1. The protocol in a nutshell

Depending on their geographical position, vehicles form groups around an RSU, which acts as group manager. From the RSU, each vehicle receives a group public key, which is shared among all group members, and a group private key, which is individual to each vehicle. The group private key is never used by the vehicle to sign V2X messages, but instead, it is used to self-certify pseudonym certificates that the vehicle creates for itself when needed (vehicle anonymity). Basically, the vehicle creates the pseudonym certificate without interacting with any authority, and affixes a signature to the bottom of the certificate done using the group private key, to certify that it is a valid member within the group. A vehicle receiving the pseudonym certificate can verify the group signature using the group public key shared with the other vehicles. If the signature is correct, it means that the vehicle that generated the pseudonym certificate is part of the group and is, therefore, entitled to participate in communications (vehicle authentication). The self-issued pseudonym certificate, on the other hand, is a traditional digital certificate based on asymmetric encryption. It contains a public key, associated with a private key that can be used to sign V2X messages. This pseudonym certificate is sent attached to V2X messages, so that the receiving vehicle can extract the public key and verify the signature on the message (message integrity and authentication).

3.2. The protocol in more detail

This subsection describes the protocol in more detail, taking Fig. 2 as a reference. The main actors involved in the protocol are:

- *CA:* it is the core of the PKI and acts as the root of trust for the entire certification chain. In a preliminary phase, the CA (or a subordinate entity) registers the vehicle and issues it with an enrolment certificate. The CA also connects with RSUs: it issues valid certificates to each RSU, and manages and distributes the Revocation Lists among them. A hierarchical relationship exists between an RSU and the CA: several regional RSUs are under the authority of the same CA. Issued vehicle enrolment certificates and RSUs certificates are considered valid by any entity only if they contain the signature of the CA.
- *RSU:* entity that acts as regional group manager and issues group keys to requesting vehicles entering its range. The RSU can also revoke vehicles that misbehave while travelling in its region of jurisdiction, by inserting their (real) vid in the Revocation Lists shared with the CA and the other regional RSUs. RSU communicates with vehicles through a public channel: it is, therefore, necessary to secure communications with encryption. Every certain time, the RSU broadcasts to nearby vehicles its digital certificate containing the public key to be used to encrypt messages. In contrast, communications with the CA take place via traditional wired networks and, therefore, are protected with classical protection mechanisms, e.g. using TLS.
- *Vehicles:* these are the vehicles travelling on the road. An unbounded number of honest vehicles and malicious ones controlled by the attacker can co-exist simultaneously. Each vehicle can generate a discretionary number of pseudonym certificates and use them to sign messages to be broadcast to other vehicles. Vehicle to vehicle communications are transmitted on a public channel: messages are sent unencrypted with a digital signature affixed, done using the current active pseudonym certificate. Communications between vehicle-RSU, instead, are both encrypted and signed. To this purpose, the enrolment certificate and the RSU certificate issued by the CA are used.

The scheme with the various interactions between the entities is shown in Fig. 2 and can be split into six phases.

(0) Offline registration. In this preliminary phase, the vehicle physically goes to the CA and registers itself. After this, it gets the enrolment certificate, $vcert_x$, from the CA: it contains the vehicle id, vid_x, a vehicle enrolment public key, vpk_x, and it is certified by the CA, $sign_{cask}$. The vehicle can use this certificate, and the corresponding enrolment private key associated with it, vsk_x, to authenticate to the RSU and request the group keys. Notice that, as said above, this certificate cannot be used to sign V2X messages, because, since it contains vid_x, it would break the anonymity requirement.

(1) Group key provisioning. As soon as the vehicle enters an area covered by an RSU, it requests from the RSU the group keys used in that region, both the public and private ones. The request contains the vehicle id, vid_x, and a *nonce*, and it is signed by the vehicle using its enrolment private keys, $sign_{vskx}$, to provide vehicle authentication. The enrolment certificate is also attached to the request, to allow the RSU to verify the signature. Since the request and the attached certificate include sensitive data (such as the vehicle id), they are encrypted using the RSU public key, $rsupk$, to achieve confidentiality. After receiving the request, the RSU first decrypts it using its $rsusk$. Then, it verifies the validity of the enrolment certificate by checking if it contains a valid CA signature. At this point, it extracts from the certificate the vehicle vid_x and vpk_x: it checks that the former is not contained in the Revocation List (otherwise it means that the vehicle has been previously revoked and therefore cannot obtain new group keys), while, with the latter, it verifies the vehicle signature on the request. If all checks are ok, the vehicle is entitled to obtain group keys and, thus, to participate in the communications. The RSU generates the group keys for the vehicle starting from $gmsk$. The public key, gpk, has only $gmsk$ as a parameter, while the private key combines both $gmsk$ and vid_x. The RSU, then, inserts the newly generated group keys and the same vid_x and *nonce* contained in the vehicle request into the response message to the vehicle. After that, it signs the response message using its private key, $rsusk$, and encrypts it using the vehicle public key, vpk_x, contained in the enrolment certificate, to provide respectively RSU authentication and message confidentiality. On the contrary, if the vehicle is not allowed to receive the group keys, the RSU returns an error message, containing the vid_x and *nonce*, encrypted with vpk_x.

(2) Pseudonym certificate creation. The vehicle autonomously creates the pseudonym certificates and self-certifies them using the group keys obtained from the RSU. In particular, the vehicle generates n pseudonym certificates, $vpseudocert_i$, each one containing a public key, $vpseudopk_i$, associated with a private key, $vpseudosk_i$, that the vehicle can use to sign V2X messages. A signature done using the vehicle group private key is affixed to the bottom of the certificate, $sign_{gsk}$. This signature will be used by other vehicles to test the validity of the certificate and, thus, to authenticate the vehicle as a member of the group. Every certain amount of time, the vehicle changes the pseudonym certificate in use to maintain unlinkability.

(3) Message sending. Along with the V2X message m, the vehicle also attaches the currently active pseudonym certificate, $vpseudocert_i$, and a signature on the message done using $vpseudosk_i$. The signature is done to provide message authentication and integrity.

(4) Message verification. When a neighbouring vehicle receives a signed V2X message, it first checks the validity of the attached pseudonym certificate used for the signature. To this purpose, it verifies the signature affixed to the certificate, $sign_{gsk}$, using the group public key, gpk, shared among all vehicles. If the signature is valid, it means that the vehicle that self-generated the certificate is a valid member of the group (i.e., it managed to obtain valid group keys from the RSU after authentication). After checking the validity of the certificate, the vehicle extracts $vpseudopk_i$ and verifies the signature on the message. Notice that it is reasonable for a vehicle to verify the certificate only the first time it receives it, and to skip its verification for subsequent messages containing the same certificate. In this way, group signature verification, which is a slow operation not fulfilling the latency requirements imposed by V2X, is not performed for every message received, making this scheme suitable for V2X communications.

(5) Report and Revocation. When a vehicle detects a malicious message (i.e., a message with misleading data), it can report it to the RSU, which starts the revocation process. To this end, the vehicle creates a report containing the malicious message, m, and the pseudonym

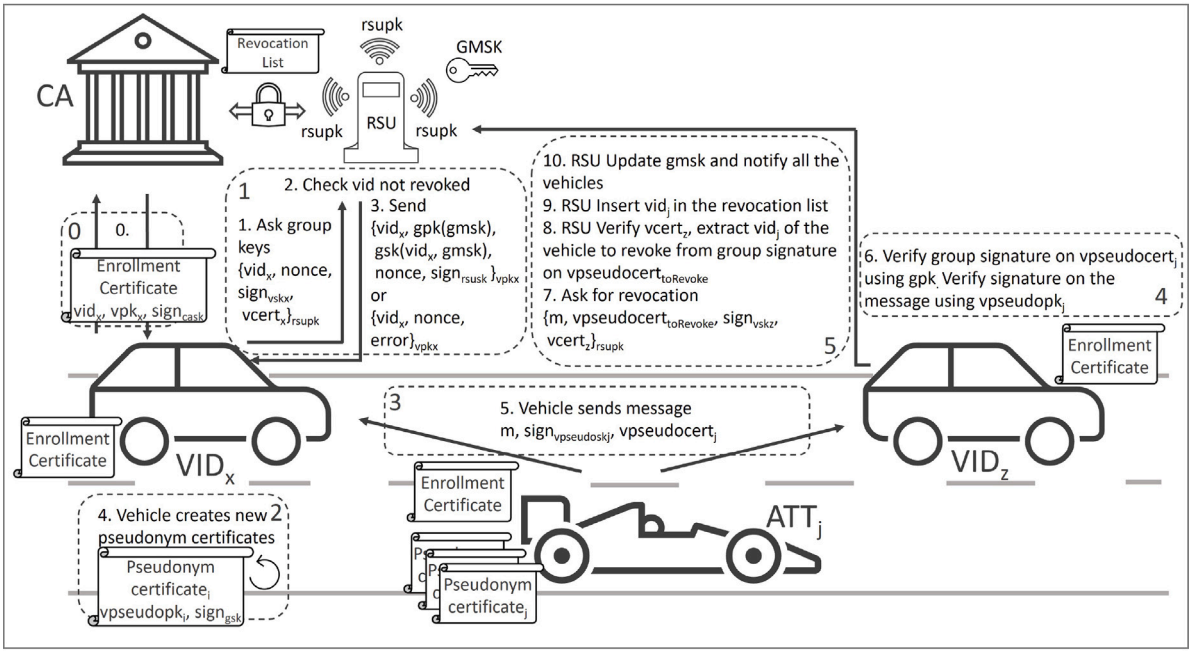

Fig. 2. Pseudonym credential management scheme.

certificate of the vehicle which signed it, $vpseudocert_{toRevoke}$. It, then, signs the report using its enrolment secret key, $sign_{vskz}$, and attaches to the report its enrolment certificate, $vcert_z$, as well, to authenticate itself. The entire request is subsequently encrypted using the RSU public key, $rsupk$, to prevent potential attackers from reading the message and discovering the identity of the vehicle requesting the revocation. Upon receipt of the request, the RSU decrypts it and uses misbehaviour detection algorithms to analyse the reported message. Then, it checks the enrolment certificate of the reporting vehicle: it extracts the vehicle id and checks it against the Revocation List. After this, it extracts the vehicle enrolment public key from the certificate, vpk_x, and verifies the signature on the report, $sign_{vskz}$. If all the checks are ok, the RSU takes $vpseudocert_{toRevoke}$ and, using the $gmsk$, it performs the $open$ operation, extracting from the group signature on the pseudonym certificate the vehicle group private key that performed it. Having this private key, the RSU can retrieve the id of the vehicle to which it was delivered, i.e., vid_j in our case (vehicle accountability). At this point, the RSU inserts vid_j into the Revocation Lists and updates the $gmsk$. It then notifies all the vehicles to request updated group keys. The process is then brought back to phase 1. In this phase, the revoked vehicle will not be able to obtain updated group keys because its vid is inserted in the Revocation List, so it is excluded from communication.

As said, this solution combines both the advantages of asymmetric encryption schemes and group signatures. From asymmetric encryption, it gets the speed of verification of the signature affixed to V2X messages, otherwise impossible to verify within the time restrictions imposed on V2X if the signature was a group signature. Group signatures, on the other hand, solve the pseudonym refill problem, with pseudonym certificates that no longer have to be requested from a CA, but can be generated on the fly by the vehicle itself. Furthermore, they allow a more flexible revocation process, which does not require the distribution of CRLs between vehicles, but a simple key update when a vehicle is revoked.

4. Protocol model

We verified the scheme described in Section 3 using Proverif, an automatic protocol verifier, [28]. Proverif takes as input an abstract model of a protocol and a set of formal security properties that the protocol must satisfy, and automatically verifies whether the properties are satisfied or not, based on internal logic, after converting the protocol and properties into Horn clauses and derivability queries.

This section describes the abstract model used for the formal verification in Proverif. This model captures the behaviour of the scheme and the interactions between the different entities involved. It also describes the capabilities of the attacker and the various assumptions that have been made for the analysis. In Section 5, on the other hand, we will show the modelling of the verified security and privacy properties.

4.1. Component model

The main actors included in the model are the same as those described in Section 3.2.

There is a Root CA, which has an asymmetric key pair, $cask$ and $capk$, trusted by all nodes involved in the protocol. There are RSUs, each one issuing group keys to requesting vehicles. Even the RSU has an asymmetric key pair, $rsusk$ and $rsupk$, and the public key is contained in a certificate signed by the CA, $rsucert$. The RSU also has a group master secret key, $gmsk$, which it uses to generate the vehicle group keys. After a revocation, $gmsk$ is updated to $updatedgmsk$. Then, there are vehicles. Specifically, we modelled: a single trusted vehicle used as a reference to test security and privacy properties, $myvid$, an unbounded number of other honest vehicles participating in the protocol, $ovid_x$, an unbounded number of malicious vehicles belonging to the attacker, $attvid_x$, and a vehicle that already begins as revoked, $revokedvid$. Both honest vehicles and malicious ones have received a valid enrolment certificate from the CA and can legitimately participate in all phases of the protocol. The difference is that, for the latter, the private keys, as well as the vehicle id and the enrolment certificate, are released to the attacker at the beginning of the protocol, and are therefore available information that the attacker can use to try to hack the scheme. The revoked vehicle, on the other hand, despite having a valid enrolment certificate, has its vid inserted in the Revocation List, so it cannot obtain group keys from the RSU. Indeed, although any vehicle can potentially request group keys from the RSU, revoked ones get an error, as specified in the protocol. Once it has obtained the group keys, each vehicle can generate an unlimited number of pseudonym certificates, and use them to sign an unlimited number of V2X messages to be broadcasted to other vehicles.

Assumptions for the Component model. The following assumptions on the components have been made to reduce the complexity of the model. (1) A single Root CA and not an actual entire certification chain. This avoids the need to verify multiple nested signatures, that use the same encryption algorithms, and therefore do not add any significant value to the verification. Assuming that all potential intermediate CAs are trusted, this assumption does not compromise the security analysis conducted on the protocol. (2) A single RSU. We suppose to be in a specific geographical location. Considering that communications between multiple RSUs take place over a secure channel, and are out-of-scope for the protocol described in this paper, we deem this assumption reasonable to simplify the Proverif verification. (3) A single vehicle initially revoked. Although there is only one vehicle in the Revocation List at the beginning, the model nevertheless foresees the possibility for vehicles to request a new revocation, so the number of vids included in that list may increase on the fly.

4.2. Channel model

Our scheme operates on three main channels:

(a) Vehicle ↔ CA (Offline registration phase)
(b) Vehicle ↔ RSU
(c) Vehicle ↔ Vehicle

The first channel is the one used in the preliminary registration phase, when the vehicle receives the enrolment keys and the enrolment certificate from the CA. This channel is the only one that is private among the three. It is, therefore, assumed that the attacker cannot act in any way on the messages exchanged over this channel. The second channel is the one between the vehicle and the RSU, and it is used to request and issue group keys and for revocation purposes. This channel is public, so the attacker can act on the exchanged messages. Since messages travelling over this channel are sensitive and confidential, the protocol requires them to be both encrypted and signed. The third channel, on the other hand, is the one that vehicles use to exchange broadcast V2X messages. Even this channel is public, with the attacker being able to send messages using the malicious vehicles it owns. In particular, messages sent over this channel are in plain text, but signed using pseudonym certificates, to protect at least authentication and integrity.

4.3. Cryptographic model

Proverif works on symbolic models that assume perfect cryptography. In other words, all cryptographic operations are modelled by means of function symbols in an algebra of terms, as black boxes that cannot be hacked. This means that attacks on the protocol that exploit weaknesses related to specific cryptographic operations and algorithms are not detected in our model. The cryptographic operations included in our model are: digital signatures, asymmetric encryption, and group signatures. Specifically, the equations describing the first two are the standard ones taken from the Proverif manual, [28]. Group signatures, instead, are modelled as shown at the bottom of Fig. 3.

The *checksign* operation allows to verify the validity of a traditional digital signature, *sign*, affixed to a message *m* and signed using a secret key, *sk*. The *checksign* takes as input the signature, the message *m*, and the public key associated with the private key used to sign, *pk(sk)*. If all parameters are correct, the signature is valid, *ok*. To encrypt a message using asymmetric encryption, function *aenc* can be used. It takes as input a message *m*, and the public key, *pk(sk)*, associated with the private key of the recipient, *sk*, and returns as output the encrypted message. To decrypt the message, one can use the reverse function *adec*, which takes the encrypted message and the private key *sk*, and returns the message *m* in plain text. As the last operation, to check the validity of a group signature, the function *gchecksign* can be used. It takes a signature *gsign*, on a message *m*, done using a group private key, *gsk*,

```
(* Digital signatures *)
checksign(sign(m, sk), m, pk(sk)) = ok.
(* Asymmetric encryption *)
adec(aenc(m, pk(sk)), sk) = m.
(* Group signatures *)
gchecksign(gsign(m, gsk(vid, gmsk)), m, gpk(gmsk)) = ok.
gopen(gsign(m, gsk(vid, gmsk)), gmsk) = vid.
```

Fig. 3. Cryptographic equations.

and verifies it with the corresponding group public key, *gpk*. As can be seen from Fig. 3, a relationship must exist between the group public and private keys, as both are obtained from the same *gmsk*. The difference is that the private one combines not only the $gmsk$ but also the *vid* of the vehicle. Finally, the *gopen* operation allows to extract the vid of the vehicle which performed a group signature with a specific *gsk* by using the *gmsk*.

4.4. Attacker model

The attacker is modelled as a classic Dolev-Yao attacker, [29]. It has full control over messages exchanged on a public network: it can read, delete, modify, replay, and forward messages. It can also create new messages from its current knowledge and perform cryptographic operations.

Mechanisms are needed to protect messages sent over a public network. In our specific scheme, the attacker can act on all messages exchanged between any two vehicles and between any vehicle and the RSU, but cannot hack messages sent to/from the CA, since communication with the CA takes place on a private channel. In our model, we gave the attacker the possibility of obtaining an unlimited number of valid vehicles (i.e. vehicles that have received a correct enrolment certificate from the CA), with which it can request group keys from the RSU and participate in the protocol as a legitimate entity. In addition, the attacker knows all publicly available information, such as the certificate of the CA, the certificate of the RSU, and all the pseudonym certificates of the other vehicles that are attached to V2X messages.

Assumptions for the Attacker model. The attacker power is limited by the Dolev-Yao model and by the perfect cryptography assumption. This implies, for instance, that the attacker cannot decrypt a message if it does not know the corresponding decryption key, or reverse a hash function, and it cannot perform brute force operations, or side channel attacks.

4.5. Protocol phases

According to the scheme, each time a vehicle is revoked, the RSU inserts its vid into the Revocation List, and updates the group master secret key, overwriting the old value $gmsk$ with a new one, $updatedgmsk$. Because of the complexity of the protocol as a whole, we decided at first to split the verification into two main parts. The first part only analyses the registration phase, the enrolment certificate and group keys provisioning, and the exchange of V2X messages (i.e., phases 0–4 of Fig. 2). This first verification model, therefore, does not include the revocation phase, nor the updating of the $gmsk$. All checks that revoked vehicles should not be able to participate in the protocol are made based on the single initially revoked vehicle, $revokedvid$. The second part of the analysis, in contrast, is specific to the revocation phase. (i.e., phase 5 of Fig. 2). We assumed in this case that vehicles are already provided with group keys and are communicating with each other, and we simulated only the revocation phase, which starts with one vehicle reporting another suspect one to the RSU. Subsequently, since Proverif was able to finish the verification of both models without

any problem, we tried to merge the two parts in a single file, which thus includes both the provisioning and revocation phases.[1] To model the Revocation List, we used a Proverif table, *table revocationlist()*, which contains the *vid* of the vehicles that have been revoked. After receiving a report and verifying the malicious vehicle message, the RSU revokes the vehicle, and performs an insert in the table, *insert revocationlist(torevokevid)*. At this point, it updates the *gmsk*, and all vehicles are expected to request updated group keys to be able to restart communications. The revoked vehicle, which now has its *vid* in the Revocation List, will fail to obtain new keys. The protocol, then, starts again from phase 1 of Fig. 2, with the difference that there is now a second vehicle, in addition to the one already present that was revoked at the beginning, which cannot get group keys. This introduces a "state" problem. Proverif has to base its decisions on the content of the Revocation List, which is not static but dynamic, since it changes as new vehicles are revoked.

A possible solution that allows Proverif to handle stateful protocols is the use of phases. To understand how it works, it must be said that all Proverif processes can be annotated with a phase prefix, using the keyword *phase x*. A process tied to a specific phase is allowed to run only on that particular phase, and if the execution is at an earlier phase, the process waits until its own phase starts. During a phase transition, when a process enters a new phase, all the others that have not yet reached it (i.e., because they are still in previous phases) are discarded. We modelled a phase transition every time a vehicle is revoked and a *gmsk* update occurs. Specifically, at an early stage, vehicles are exchanging messages and participating in the protocol. At some point, a vehicle requests a revocation from the RSU. The RSU inserts the reported *vid* into the Revocation List, and a phase transition is triggered. The RSU updates the *gmsk* and notifies the other vehicles. A new phase transition takes place, with the vehicles requesting the updated keys and resuming the communications.

Our Proverif model using phases is described in Fig. 4. A process not annotated with any phase indication is by default in phase 0. At the beginning, in the main *process*, two processes are created for each vehicle and two processes for the RSU. There is a process, *Vehicle*, which starts immediately in phase 0, and another identical one waiting in phase 2. These processes receive as parameters the *vid*, the enrolment private key, the enrolment certificate, and the public key of the CA to be considered trusted. To model the RSU behaviour, instead, a process *RSUReleaseGroupKey* is created, which vehicles can contact to request group keys, and a process *RSURevoke*, which can be contacted to ask for revocation, both starting in phase 0. Phase 1 is used for synchronisation purposes only. This phase is executed after the RSU has received a revocation request, and after it has inserted the *vid* of the vehicle to be revoked, *torevokevid*, into the *revocationlist*. The only operation performed in this phase is the update of the *gmsk* in *updatedgmsk*. All vehicle processes that were operating in phase 0 are deleted, and the same happens to process *RSUReleaseGroupKey*. After updating the *gmsk*, *RSURevoke* enters phase 2. All *Vehicle* processes that were waiting in phase 2 are now started, and a new process *RSUReleaseGroupKey* is created, which takes the updated key, *updatedgmsk*, as a parameter to create the new group keys.*Vehicles* can then request new group keys from this process.

Assumptions on Protocol phases. The mechanism using phases described above allows to handle state in Proverif, but introduces some limitations. The first great assumption of this model is that only one vehicle can be revoked. If desired, one can increase the number of revoked vehicles by restarting the *RSU Revoke* process in phase 2,

```
process ...
  !( ... create new vehicle ...
    ( Vehicle(vid, vsk, vcert, capk) |
      phase 2;
      Vehicle(vid, vsk, vcert, capk)
    )
  )
  | !RSUReleaseGroupKey(gmsk, ...)
  | !RSURevoke(gmsk, ...)

let RSUReleaseGroupKey(gmsk:gmskey, ...) = ...

let RSURevoke(gmsk, ...) = ...
  insert revocationlist(torevokevid);
  phase 1;
  new updatedgmsk:gmskey;
  phase 2;
  !RSUReleaseGroupKey(updatedgmsk, ...)
```

Fig. 4. Protocol phases.

and repeating the phase-by-phase mechanism described above (e.g., by introducing phases 4, 5, etc.). However, the procedure can only be repeated a limited number of times, and it is not possible to test the case where there is revocation of a potentially unlimited number of vehicles. Despite this, we believe that the verification is not particularly affected by this limitation, since it still analyses a complete protocol cycle (starting from phase 0, to phase 5, and then to phase 1 again). The second assumption concerns the way Proverif handles phases. In particular, when the RSU enters phase 1, all vehicles sending messages in phase 0 are discarded and they are no longer considered in the verification. Despite this, the attacker can continue to operate in all phases, and preserve the knowledge it has learnt in a specific phase, also for subsequent phases. From the above considerations, even if the model does not represent reality exhaustively, its verification still gives high assurance of the validity of the verified properties.

5. Security and privacy properties

In this section, we describe the security and privacy properties verified on the protocol model described in the previous section. For each property, we give a definition of it and explain how it was modelled in Proverif. Specifically, we checked the following properties: *Sanity checks*, *Authentication*, *Secrecy*, *Anonymity*, *Unlinkability*, *Distributed resolution*, and *Perfect forward secrecy*. Apart from Sanity checks, which are used to check the Proverif protocol model behaves as expected, the other properties are those described in the related work [8], and shown in Table 1.

Sanity checks: We modelled two types of sanity checks: the first one using events in specific points of each process simply verifying that the process can reach these events, and the second one using Correspondence Assertions and Temporal Correspondence Assertions to match more complex expected behavioural properties. The complete list of verified sanity checks can be found in Table 3 in Appendix.

Authentication: This property tests the authentication of the actors involved in the protocol in different phases of the scheme. It consists of verifying that a node authenticating to another one is indeed whom it claims to be. In Proverif, authentication can be verified using Correspondence Assertions and Injective Correspondence Assertions. The list of the verified Correspondence Assertion queries can be found in Table 3 in Appendix.

Secrecy: We tested the secrecy of the real identities of the vehicles participating in the protocol and their private keys. In particular, an attacker intercepting all communications must not be able to discover

[1] All Proverif source files can be found at the following open source repository, https://github.com/netgroup-polito/verification-v2x/. The division into folders is as follows: /provisioning contains the file for the first model. /revocation for the model with the revocation part only. /complete-protocol for the model that includes both the previous two.

Anonymity when using the group private key
$(P;\ let\ vpseudocert = pseudocert(vpseudopk, vgsk);\ldots) \approx$
$(P;\ let\ vpseudocert = pseudocert(vpseudopk, anonymgsk);$

Anonymity when signing a V2X message
$(P;\ let\ vpseudocert = pseudocert(vpseudopk, vgsk);$
$let\ signm = sign(m, vpseudosk);$
$out(m, signm, vpseudocert);\ldots) \approx$
$(P;\ let\ anonymcert = pseudocert(anonympk, vgsk);$
$let\ signm = sign(m, anonymsk);$
$out(m, signm, anonymcert);\ldots)$

Fig. 5. Analysing anonymity in Proverif.

Unlinkability in group key requests
$!(P; !VehicleRequestGroupKeys(vid, vsk, vcert)) \approx$
$!(P; VehicleRequestGroupKeys(vid, vsk, vcert))$

Unlinkability in the creation of a pseudocert
$!(P; !VehicleCreatePseudocert(vid, gmsk)) \approx$
$!(P; VehicleCreatePseudocert(vid, gmsk))$

Unlinkability of V2X messages
$!(P; !VehicleSendMessage(vpseudosk, vpseudocert)) \approx$
$!(P; VehicleSendMessage(vpseudosk, vpseudocert))$

Unlinkability in revocation requests
$!(P; !VehicleReport(vid, vsk, vcert, vpseudocert_{toRevoke})) \approx$
$!(P; VehicleReport(vid, vsk, vcert, vpseudocert_{toRevoke}))$

Fig. 6. Analysing unlinkability in Proverif.

the real vid_x of a vehicle, nor its private enrolment key and group key. Secrecy can be modelled in Proverif in different ways. Simple secrecy can be verified as a Reachability property: it tests that an attacker cannot obtain terms that are expected to be secret. Weak secrecy, on the other hand, can be tested using the keyword *weaksecret*. This type of secrecy is the one used to identify guessing attacks, such as dictionary attacks or password guessing. This property assumes that an attacker, which participates in the protocol and obtains the encrypted secret, subsequently fails to distinguish a correct guess for that term from a wrong one without any further interaction with the protocol (i.e., acting offline). Proverif can model weak secrecy as an Observational equivalence. Another type of secrecy is Strong secrecy. It differs from the others in that it not only verifies that the attacker is unable to obtain a secret, but also to distinguish when the secret changes. This property can be verified in Proverif using the keyword *noninterf*, and it is modelled internally as an Observational equivalence. All the secrecy properties tested on the protocol are described in Table 3 in Appendix.

Anonymity: The main claim of the scheme described in this paper is that it allows vehicles to participate in V2X communications without revealing their real identity. So, anonymity is defined to check exactly this. A vehicle sending broadcast V2X messages should remain anonymous at least to other vehicles receiving the messages. V2X schemes typically refer to anonymity not in its strict term, but usually intended as Conditional Anonymity: the vehicle must remain anonymous during communications, but at the same time there must be a mechanism to allow authorities to discover its identity in case of violations of the law. Specifically, in our scheme, we verified the anonymity of the vehicle when it uses its group secret key to self-certify pseudonym certificates, and when it uses the secret key associated with these certificates to sign V2X messages. Anonymity in Proverif can be modelled as an Observational equivalence using the construct *choice*.

Following the code described in Fig. 5, to check the anonymity of a group signature, we created a fake, but valid, anonymous group secret key, *anonymgsk*, different from the vehicle real one, *vgsk*, and we checked if the attacker was able to distinguish a signature done using the real vehicle key from one done using the anonymous one. If a signature attached to a *pseudocert* using *vgsk* appears to the attacker to be indistinguishable from the same signature made using *anonymgsk*, then we have vehicle anonymity. The same applies to the second property, anonymity when signing a V2X message: the attacker should not able to distinguish the case in which a V2X message is signed using a *vpseudosk* from one where an *anonymsk* is used. To verify this second property successfully, it is important to attach the correct certificate to the signed V2X message. That is, if the message is signed with *vpseudosk*, then, *vpseudocert* must be attached. Whereas if the anonymous key is used, *anonymcert* must be sent together with the message. Indeed, if this distinction is not made and the same certificate is used in both cases, there is one case in which the signature verification is correct and one in which the signature verification is incorrect (because the attached certificate contains a public key not associated to the private key used for signing). The attacker would then be able to distinguish the two processes, and Observational equivalence would fail.

Unlinkability: By definition, unlinkability exists when a vehicle can participate multiple times in the protocol, without being tracked. Multiple executions of the protocol performed by the same vehicle must not be linkable to each other. We tested unlinkability in group key requests, in the creation of pseudocerts using the same group private key, in sending V2X messages, and finally in vehicle revocation requests. In Proverif, unlinkability can again be modelled as an Observational Equivalence, using the *equivalence* construct. In particular, to have unlinkability, a process P in which the vehicle participates several times in the protocol must be equivalent to a version of P in which the vehicle participates only once. If this equivalence is verified, it means that the various protocol executions cannot be linked together. The code in Fig. 6 describes the unlinkability properties that have been verified.

As an example, we can take the first tested property. In this case, the equivalence states that a version of the protocol in which the vehicle requests an unlimited number of group keys from the RSU is equivalent to a version of it in which the vehicle requests only one group key. Therefore, if this property holds, an attacker which collects different group key requests cannot tell whether they belong to the same vehicle or to different ones. The same applies to the other unlinkability properties described above.

Distributed resolution: There is distributed resolution when the process to obtain the secret identity of a vehicle is distributed among several authorities, that must cooperate with each other to obtain the secret. No single authority, working alone, should be able to determine the vehicle real identity starting from its anonymous credentials. This property is desired to protect the identity of the vehicle in the event that a CA becomes compromised. In our scheme, the RSU alone, or the CA alone, shall not be able to determine the identity of the vehicle independently, without cooperating with each other. To test this property in Proverif, we first provided the attacker with the secret key of the RSU, and then with the secret key of the CA, simulating them being compromised, and we verified that the attacker, with neither key taken separately, was able to obtain the secret identity of the vehicle.

Perfect forward secrecy/privacy: Finally, perfect forward secrecy exists when the compromise of a credential in a given session only affects secrets exchanged in that particular session, but not those of other sessions. In particular, for a V2X scheme, perfect forward secrecy coincides with perfect forward privacy when the secret to be protected is the real identity of the vehicle. We tested what happens to the protocol in the event of a compromise of *cask*, *rsusk*, and *gmsk*. In Proverif, forward secrecy can be tested using phases. Specifically, assuming that the protocol takes place in phases 0 to $n-1$, a compromise of the key that occurs in phase n should not affect the secrets exchanged in the previous n phases.

6. Verification results

In this section, we describe the results of the verification conducted on the properties described in Section 5, using Proverif. Some of the results that emerge from the verification have already been described in our previous work, [11]. In this paper, we add further considerations to the discussion as a result of adding new properties to the model, and whenever possible, we try to propose solutions that can be adopted to solve the weaknesses found. All the attack traces returned by Proverif can be found at.[2]

6.1. Results from sanity checks

Although sanity checks are most often used to verify the compliance of the Proverif model to the analysed scheme, in this work they have been useful to detect some weaknesses of the modelled protocol. In particular, the queries that led to interesting results are Correspondence Assertions and Temporal Correspondence Assertions. The weaknesses found are the following.

6.1.1. Vehicle continues to be considered valid even after being removed

This weakness is derived from the analysis of properties 8. and 13. of Table 3. Proverif found that there are two specific moments in the protocol in which a vehicle that has been revoked can continue to send messages and be considered trusted by other vehicles. Following the division into phases presented in Section 4.5, this occurs first in the period of transition from phase 0 to phase 1, i.e. immediately after the RSU has inserted the vid of the revoked vehicle in the Revocation List, but before it has updated the $gmsk$ and notified the other vehicles to ask for new group keys. Secondly, it occurs in phase 2, when the RSU may have notified vehicles to request new group keys, but the vehicles may have not yet done it and still continue to use old group keys, thus accepting messages from an attacker using revoked keys. This is a classic problem of Windows of Exposure, which is well known in the literature, and which also exists for other protocols using Certificate Revocation Lists and digital certificates. In our specific scheme, the problem is amplified if the attacker succeeds in blocking communication between the RSU and the vehicle, thus ensuring that the vehicle never receives the notification from the RSU to request updated group keys.

A solution proposed in the literature, [30], which can be adopted even for the scheme described in this paper, is to release group keys with a short life. With this solution, every certain short amount of time, the RSU must update the $gmsk$, albeit there has been no revocation, and all vehicles must request new group keys to continue participating in the protocol. This solution is useful to reduce the Window of Exposure, but at the cost of requiring more interactions between the vehicles and the RSU. A trade-off between security and usability must be studied, to find the optimal life time for group keys.

6.1.2. Pseudonym certificate continues to be used even after the vehicle has been removed

This weakness was detected analysing property 14. of Table 3. It occurs when the attacker owns several vehicles participating in the protocol. Let us imagine that an attacker vehicle is revoked because it has signed a message considered malicious. This vehicle can no longer obtain valid group keys from the RSU. However, it may happen that a second attacker vehicle, which has not been revoked and can request updated group keys, takes the $pseudocert$ of the revoked vehicle and recertifies it using the new group key. By doing so, the message that caused the revocation of the first attacker can continue to be used combined with the new $pseudocert$.

A similar problem may exist in the case of an attacker sharing his private group key with other attacker vehicles. In the modelled scheme, a private group key is unique for each vehicle. However, the attacker could distribute its private key to other attacker vehicles, that could use it to self-generate pseudonym certificates used to sign V2X messages. Honest vehicles cannot understand that different pseudonym certificates, sent by different vehicles, are actually all signed using the same group key. This problem is embedded in the concept of anonymity, and is shared with all other V2X schemes.

Both problems are solved after all attacker vehicles are revoked, one by one. When the attacker has all its vehicles revoked, it can no longer request updated group keys, and thus participate in the protocol. One of the main issues of the scheme described in this paper is that, since pseudonym certificates are self-generated and certified by each vehicle, vehicles have great freedom over them. For instance, an attacker that has a valid enrolment certificate can take old pseudocerts and recertify them, generate pseudocerts that contain a non globally unique key, or even certify for itself a public key that already belongs to another vehicle. To solve this problem, it is necessary to equip each vehicle with an Hardware Security Module or Trusted Platform Module, which directly manages the certificates by communicating with the RSU on a secure channel. The private keys associated with these certificates are never exported externally to the vehicle, which therefore cannot share them with other malicious vehicles.

6.2. Results from secrecy and authentication

All the verified properties concerning the secrecy of the vehicle id are fulfilled by the scheme. The attacker, in fact, has no way to discover the identity of the vehicle, nor to learn if it changes (i.e., strong secrecy is also respected). The same applies to authentication properties.

6.3. Results from anonymity

The analysis conducted on the protocol for what concerns anonymity did not point out any weaknesses. Vehicles participating in the protocol remain anonymous to other vehicles and have their vid always kept secret. In our previous work, [11], we showed a possible anonymity issue for a revoked vehicle. In a nutshell, the attacker, by replaying a group key request sent by another vehicle to the RSU, was able to understand whether the vehicle linked to the request had been revoked or not, based on the RSU response (more specifically, based on whether the RSU responded to the request or ignored the message). The solution proposed in [11] was to introduce an error message in the RSU response to group key requests coming from revoked vehicles, encrypted with the public key of the recipient vehicle. Proverif proved here that the solution actually solves the problem.

6.4. Results from unlinkability

Regarding unlinkability in the request for group keys, Proverif showed that, thanks to the $nonce$ in the message, the property is met. Our previous work, [11], had discussed the implications in case this $nonce$ was not present. Regarding unlinkability of V2X messages, instead, as long as these messages are signed with the same pseudonym certificate, they are linkable to each other. The vehicle, however, is expected to change the pseudonym certificate often, e.g. every certain amount of time or every certain number of signed messages. When this happens, Proverif has shown that the messages signed with the new certificate appear unlinked to the old ones. Finally, no unlinkability issues were found in the creation of pseudonym certificates and in the requests for revocation of a vehicle.

6.5. Results from distributed resolution and perfect forward secrecy

Since properties related to Distributed resolution and Perfect forward secrecy both require the disclosure of some kind of secret key

[2] https://github.com/netgroup-polito/verification-v2x/attack-traces

belonging to any of the authorities to be verified, we grouped the results and weaknesses found by analysing these properties in a single subsection. For each revealed secret, we describe the consequences arising and which properties continue to hold despite the compromise, should there be any.

6.5.1. Leakage of the root CA secret key

With the root CA secret key in its possession, the attacker would be able to do almost anything. It could create valid enrolment certificates and issue them to malicious vehicles, or it could certify bogus RSUs by providing them valid signed certificates. Honest vehicles could, then, be deceived to contact a bogus RSU, and request group keys from it, revealing their real identity. The attacker impersonating the bogus RSU can, in fact, create a fresh $attgmsk$ and issue valid group keys to vehicles starting from it.

However, the attacker would not be able, with the $cask$ alone, to discover the group keys issued to vehicles in the past by honest RSUs. This is because all requests for group keys sent in the past were encrypted with the key of the honest RSU, which is unknown to the attacker. There is, therefore, forward secrecy for group keys issued in the past, and anonymity of the vehicle in the past is still fulfilled.

6.5.2. Leakage of the RSU secret key

With the secret key of the RSU, the attacker would be able to discover the real identity of the vehicle. Indeed, the group key requests sent by the vehicle to the RSU contain the enrolment certificate of the vehicle. Using $rsusk$, the attacker can decrypt the request and extract the actual vid of the vehicle from the certificate. The same happens for a vehicle revocation request. Since we want to be sure of the identity of the requesting vehicle, in fact, even the revocation request contains the vehicle enrolment certificate, from which the actual vid can be extracted. The analysed scheme is, therefore, subject to the so-called identity escrow problem: the RSU manages to obtain the vid of the vehicle even without collaborating with other authorities.

To solve the identity escrow problem, a mechanism similar to the one described in [16] and [17] can be implemented. In these schemes, the responsibility for the secrecy of the vid is shared between two different authorities, which must necessarily cooperate if they want to obtain all data associated with a vehicle. By adapting the solution to our scheme, one could think of the Root CA knowing the real vid of the vehicle and issuing it an anonymous enrolment certificate, which the vehicle can use to authenticate to the RSU anonymously. The RSU, for its part, authenticates the vehicle using its anonymous certificate and issues it with group keys. The RSU keeps the association (group key - anonymous identity), but it does not know the vehicle real vid. The Root CA, on the contrary, has the association (real vid - anonymous identity), but it does not know which group keys were issued to the vehicle by the RSU. To obtain all the information, e.g. retrieve the identity of a vehicle that signed a malicious message, the RSU must first extract the group key used to certify the pseudonym certificate associated with the message. Then, it has to find the corresponding anonymous identity to which that group key was issued, and return it to the Root CA. The Root CA can finally retrieve the actual vid of the vehicle associated with the anonymous identity from its database.

Concerning the other properties, there is no forward secrecy as a result of $rsusk$ leakage. Being the $rsusk$ a long-term key, the attacker in possession of it can decrypt past and present group key/revocation requests. As a consequence of $rsusk$ leakage, other problems may arise. Specifically, the attacker could impersonate the RSU, generate an $attgmsk$, request vehicles to update their group keys, and issue them group keys based on the newly created $attgmsk$. Knowing the group private key issued to each vehicle, the attacker would be able to find all the pseudonym certificates signed using that private key, thus to associate each message with the identity of the vehicle that generated it. Moreover, knowing the group private key issued to each vehicle, the attacker could also generate bogus pseudonym certificates by signing them with the vehicle key, and impersonate the vehicle.

6.5.3. Leakage of the RSU group master secret key

The leakage of the group master secret key can also create several problems to the scheme. First, with the $gmsk$, the attacker can perform the $gopen$ operation and obtain the true vid of the vehicle by extracting it from the group signature attached to a self-certified $vpseudocert$. The attacker can later collect all the V2X messages associated with that $vpseudocert$, and link those messages to the identity of the vehicle. The protocol would thus lose both anonymity and unlinkability. Another significant issue is that, using the $gmsk$, the attacker can create bogus group keys, both the private and the public part, and distribute them to other malicious vehicles, that do not have an enrolment certificate.

Perfect forward secrecy, on the other hand, still holds, but only if defined at a "phase" level and not to all individual messages. That is, if the attacker obtains the group master secret key, vehicles lose their anonymity for all messages exchanged in the current phase that uses that $gmsk$ to generate group keys. But, as soon as the $gmsk$ is updated by the RSU, either because its lifetime has expired or because a vehicle has been revoked, the attacker loses its power. Secrecy of the vid for other phases becomes valid again, because the attacker does not know the $updatedgmsk$.

6.6. Discussion and other considerations

Table 2 shows a summary from the analysis of the various properties in Proverif. Results confirm what was expected from the scheme, i.e. in the absence of any key leakage, most of the security and privacy properties are fulfilled by the protocol model. The only exceptions concern *Distributed resolution* and *Perfect forward secrecy*. Indeed, as we discussed in the previous subsections, the scheme is sensitive to an authority behaving maliciously, either the RSU or the CA, and/or a compromise of any of their keys. Other considerations that are necessary from this point of view are as follows.

In the group signature-based scheme modelled in this paper, privacy is conditional on both the RSU and the CA. Consequently, the scheme suffers from *identity escrow*: the RSU and the CA both have a way to discover the identity of a vehicle participating in the protocol even without collaborating with each other. This is one of the main problems of the protocol, because it allows a compromised entity to break most of the required security and privacy properties.

Another important consideration to be made is that, with group-signature based schemes such as the one described in this paper, vehicles have full control over the pseudonym certificates they use to sign V2X messages. These certificates are, in fact, self-generated and self-certified by each vehicle. Therefore, the vehicle can insert any timestamp and validity period it wishes into these certificates. Several certificates can be generated and used simultaneously, all having the same validity period. There is therefore the risk that a single malicious vehicle may generate a large number of certificates and distribute them to other attackers. In other schemes, such as the one using only asymmetric encryption, the control over the certificates is in the hands of a CA. The vehicle can obtain several certificates from the CA at the same time, but each with a specific validity period, typically sequential to the previous one. It is therefore impossible to use several certificates at the same time and distribute them to other vehicles.

7. Conclusions

In this paper, we have modelled and verified a V2X scheme proposed in the literature, [10], that combines asymmetric encryption and group signatures. This type of scheme provides a viable alternative to the purely asymmetric encryption based models that have been chosen by SDOs to support V2X communications, which still present significant issues limiting their adoption and implementation. We have recalled the advantages of using such a combined scheme and, then, formally analysed its main security and privacy properties.

Table 2
Verification results on properties.

Property	Description	Result
Authentication	Authentication of the vehicle and RSU in the different protocol phases	✓
Secrecy	Secrecy of the vehicle identity and private keys	✓
Anonymity	Anonymity of the vehicle	✓
Unlinkability	Unlinkability of vehicle messages	✓
Distributed resolution	Distributed resolution between the RSU and the CA	✗
Perfect forward secrecy	Secrecy in case of compromise of the RSU and/or CA keys	✗

The analysis has been carried out using Proverif. We have proposed a formal model that is able to formalise both the details related to the specific protocol under analysis and the security and privacy properties it must fulfil. The verification results have shown that the analysed V2X scheme satisfies most of the expected security and privacy properties, with the exception of some issues discussed in this paper. For each weakness found, we have proposed a solution, combining the scheme with some countermeasures that have been proposed in other papers in the literature.

As a future work, we think that starting from the model a possible implementation of the protocol can be built, to test its feasibility in real case scenarios. An implementation and then large-scale experiments are in fact necessary to bring V2X to a more advanced stage, and this is what most proposed schemes usually still lack. Furthermore, the model described in this paper can be easily adapted to analyse other V2X schemes existing in the literature. We plan to extend our analysis to those as well as future work.

CRediT authorship contribution statement

Simone Bussa: Writing – original draft, Formal analysis. **Riccardo Sisto:** Writing – original draft, Formal analysis. **Fulvio Valenza:** Writing – original draft, Formal analysis.

Declaration of competing interest

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

Appendix

See Table 3.

Data availability

No data was used for the research described in the article.

Table 3
Security and privacy properties: Proverif queries.

Sanity checks		
1.	Vehicle created	query event(VehicleCreated({myvid | ovid$_x$ | attvid$_x$}))
2.	Vehicle can get a valid group key	query event(ValidGroupPrivateKeyReceived({myvid | ovid$_x$ | attvid$_x$}))
3.	Revoked vehicle cannot get a valid group key	query not event(ValidGroupPrivateKeyReceived(revokedvid$_x$)), query event(RevokedCannotGetGroupKey(revokedvid$_x$))
4.	Revoked vehicle receives an error message from the RSU	query event(ErrorMessageReceived(revokedvid$_x$))
5.	Vehicle can send a message	query event(ValidMessageSent({myvid | ovid$_x$ | attvid$_x$}, vpseudocert$_j$, m))
6.	Vehicle can send two messages with the same pseudonym	query event(ValidMessageSent({myvid | ... }, vpseudocert$_j$, m1)) && event(ValidMessageSent({myvid | ... }, vpseudocert$_j$, m2))
7.	Vehicle can send two messages with different pseudonyms	query event(ValidMessageSent({myvid | ... }, vpseudocert$_j$, m1)) && event(ValidMessageSent({myvid | ... }, vpseudocert$_z$, m2))
8.	Revoked vehicle cannot send a message	query not event(ValidMessageSent(revokedvid$_x$, vpseudocert$_j$, m))
9.	Vehicle can ask revocation of another vehicle	query event RevocationAsked({myvid | ovid$_x$ | attvid$_x$}, vcert$_x$, torevokecert$_j$)
10.	RSU can revoke vehicles	query event RevokedVid(torevokevid$_x$)
11.	If a vehicle cannot get keys, then it has been revoked	query event(RevokedCannotGetGroupKey(vid)) ==> event(RevokedVid(vid))
12.	If a vehicle cannot get keys, then it has been revoked	(Temporal) query event(RevokedCannotGetGroupKey(vid))@t2 && event(RevokedVid(vid))@t1 ==> t1 < t2.
13.	Cannot receive a valid message from a revoked vehicle	(Temporal) query event(ValidMessageSent(sendervid, vpseudocert, m))@t1 && event(ValidMessageReceived(receivervid, m))@t2 && event(RevokedVid(sendervid))@t3 ==> t1 < t2 && t2 < t3
14.	Cannot receive a valid message signed using a revoked cert	(Temporal) query event(ValidCertReceived(receivervid, vpseudocert))@t1 && event(RevokedCert(vpseudocert))@t2 ==> t1 < t2
Authentication		
15.	Authentication of the vehicle to the RSU in phase 1	query event(ValidGroupKeyRequestReceived(rsusk, vid$_x$, nonce)) ==> event(ValidGroupKeyRequestSent(vid$_x$, nonce)) | | event(AttackerGetsEnrollmentCertificate(vid$_x$))
16.	Authentication of the RSU to the vehicle in phase 1	query event(ValidGroupPrivateKeyReceived(vid$_x$, vgsk)) ==> event(ValidGroupPrivateKeySent(rsusk, vid$_x$, vgsk))
17.	Authentication of the vehicle to the RSU in phase 5	query event(ValidRevocationRequestReceived(rsusk, vid$_x$, vcert$_x$, torevokecert$_y$)) ==> event(ValidRevocationRequestSent(vid$_x$, vcert$_x$, torevokecert$_y$)) | | event(AttackerGetsEnrollmentCertificate(vid$_x$))
18.	Authentication of the vehicle when sending a message	query event(ValidMessageReceived(receivervid$_x$, vpseudocert$_y$, m)) ==> event(ValidMessageSent(sendervid$_z$, vpseudocert$_y$, m)) | | event(AttackerGetsEnrollmentCertificate(sendervid$_z$)).
Secrecy		
19.	Simple secrecy of the vehicle identity	query attacker(vid$_x$)
20.	Weak secrecy of the vehicle identity	weaksecret vid$_x$
21.	Strong secrecy of the vehicle identity	noninterf vid$_x$
22.	Simple secrecy of the vehicle group private key	query attacker(gsk(vid$_x$, gmsk))

(*continued on next page*)

Table 3 (*continued*).

Anonymity		
23.	Anonymity of a group signature affixed to a pseudocert	let vpseudocert = pseudocert(vpseudopk, choice[vgsk, anonymousgsk])
24.	Anonymity of a signature affixed to a V2X message	let VehicleSendMessages(vid, vgsk, choice[vpseudosk, anonymsk])
Unlinkability		
25.	Unlinkability in group key requests	See Proverif source files at github.com/netgroup-polito/verification-v2x/
26.	Unlinkability in the creation of a pseudocert	See Proverif source files at github.com/netgroup-polito/verification-v2x/
27.	Unlinkability of V2X messages	See Proverif source files at github.com/netgroup-polito/verification-v2x/
28.	Unlinkability in revocation requests	See Proverif source files at github.com/netgroup-polito/verification-v2x/
Distributed resolution		
29.	RSU alone should not be able to obtain the vehicle identity	out(rsusk); query attacker(vid_x)
30.	CA alone should not be able to obtain the vehicle identity	out(cask); query attacker(vid_x)
Perfect forward secrecy		
31.	Compromise of the CA secret key	phase n; out(cask); query attacker(vid)
32.	Compromise of the RSU secret key	phase n; out(rsusk); query attacker(vid)
33.	Compromise of the group master secret key	phase n; out(gmsk); query attacker(vid)

References

[1] Qureshi Kashif Naseer, Abdullah Abdul Hanan. A survey on intelligent transportation system. Middle- East J Sci Res 2013;15(5):629–42.

[2] MacHardy Zachary, Khan Ashiq, Obana Kazuaki, Iwashina Shigeru. V2X access technologies: Regulation, research, and remaining challenges. IEEE Commun Surv Tutor 2018;20(3):1858–77.

[3] Liu Liangkai, Lu Sidi, Zhong Ren, Wu Baofu, Yao Yongtao, Zhang Qingyang, Shi Weisong. Computing systems for autonomous driving: State of the art and challenges. IEEE Internet Things J 2020;8(8):6469–86.

[4] Parkinson Simon, Ward Paul, Wilson Kyle, Miller Jonathan. Cyber threats facing autonomous and connected vehicles: Future challenges. IEEE Trans Intell Transp Syst 2017;18(11):2898–915.

[5] Alsabaan Maazen, Alasmary Waleed, Albasir Abdurhman, Naik Kshirasagar. Vehicular networks for a greener environment: A survey. IEEE Commun Surv Tutor 2012;15(3):1372–88.

[6] Automotive Vehicle-to-Everything Market Size, Share and Trends Analysis Report - Grand View Research, https://www.grandviewresearch.com/industry-analysis/automotive-vehicle-to-everything-v2x-market.

[7] Taslimasa Hamideh, Dadkhah Sajjad, Neto Euclides Carlos Pinto, Xiong Pulei, Ray Suprio, Ghorbani Ali A. Security issues in internet of vehicles (IoV): A comprehensive survey. Internet Things 2023;22:100809.

[8] Schaub Florian, Ma Zhendong, Kargl Frank. Privacy requirements in vehicular communication systems. In: 2009 International conference on computational science and engineering, vol. 3. IEEE; 2009, p. 139–45.

[9] Petit Jonathan, Schaub Florian, Feiri Michael, Kargl Frank. Pseudonym schemes in vehicular networks: A survey. IEEE Commun Surv Tutor 2014;17(1):228–55.

[10] Calandriello Giorgio, Papadimitratos Panos, Hubaux Jean-Pierre, Lioy Antonio. Efficient and robust pseudonymous authentication in VANET. In: Proceedings of the fourth ACM international workshop on vehicular ad hoc networks. 2007, p. 19–28.

[11] Bussa Simone, Sisto Riccardo, Valenza Fulvio. Formal verification of a V2X privacy preserving scheme using proverif. In: 2023 IEEE international conference on cyber security and resilience. 2023, p. 341–6. http://dx.doi.org/10.1109/CSR57506.2023.10224908.

[12] ETSI. 2019. Intelligent transport systems (ITS) vehicular communications; basic set of applications; part 2: Specification of cooperative awareness basic service. 2019, https://www.etsi.org/deliver/etsi_en/302600_302699/30263702/01.04.01_60/en_30263702v010401p.pdf.

[13] ETSI. 2019. Intelligent transport systems (ITS). Vehicular communications; basic set of applications; part 3: Specifi- cation of decentralized environmental notification basic service.. 2019, https://www.etsi.org/deliver/etsi_en/302600_302699/30263703/01.03.01_60/en_30263703v010301p.pdf.

[14] V2X Core Technical Committee. V2X communications message set dictionary. 2022, http://dx.doi.org/10.4271/J2735_202211.

[15] Yoshizawa Takahito, Singelée Dave, Muehlberg Jan Tobias, Delbruel Stephane, Taherkordi Amir, Hughes Danny, et al. A survey of security and privacy issues in V2X communication systems. ACM Comput Surv 2023;55(9):1–36.

[16] ETSI-TS 102 940 v2.1.1 - ITS communications security architecture and security management. 2021, https://www.etsi.org/deliver/etsi_ts/102900_102999/102940/02.01.01_60/ts_102940v020101p.pdf.

[17] Brecht Benedikt, Hehn Thorsten. A security credential management system for V2X communications. Connect Vehicles Intell Transp Syst 2019;83–115.

[18] Yoshizawa Takahito, Preneel Bart. Survey of security aspect of V2X standards and related issues. In: 2019 IEEE conference on standards for communications and networking. IEEE; 2019, p. 1–5.

[19] Whitefield Jorden, Chen Liqun, Kargl Frank, Paverd Andrew, Schneider Steve, Treharne Helen, Wesemeyer Stephan. Formal analysis of V2X revocation protocols. In: Security and trust management: 13th international workshop, STM 2017, oslo, Norway, September 14–15, 2017, proceedings 13. Springer; 2017, p. 147–63.

[20] Fazouane Marouane, Kopp Henning, van der Heijden Rens W, Le Métayer Daniel, Kargl Frank. Formal verification of privacy properties in electric vehicle charging. In: Engineering secure software and systems: 7th international symposium, eSSoS 2015, milan, Italy, March 4-6, 2015. proceedings 7. Springer; 2015, p. 17–33.

[21] Bißmeyer Norbert, Stübing Hagen, Schoch Elmar, Götz Stefan, Stotz Jan Peter, Lonc Brigitte. A generic public key infrastructure for securing car-to-x communication. In: 18th ITS world congress, orlando, USA, vol. 14. 2011.

[22] Papadimitratos Panagiotis, Buttyan Levente, Holczer Tamás, Schoch Elmar, Freudiger Julien, Raya Maxim, et al. Secure vehicular communication systems: design and architecture. IEEE Commun Mag 2008;46(11):100–9.

[23] ETSI-TS 102 941 v2.2.1 - security; trust and privacy management; release 2. 2022, https://www.etsi.org/deliver/etsi_ts/102900_102999/102941/02.02.01_60/ts_102941v020201p.pdf.

[24] IEEE standard for wireless access in vehicular environments–security services for application and management messages. 2022, https://standards.ieee.org/ieee/1609.2/10258/.

[25] Ryan Mark D, Smyth Ben. Applied pi calculus. In: Formal models and techniques for analyzing security protocols. Ios Press; 2011, p. 112–42.

[26] Brusó Mayla, Chatzikokolakis Konstantinos, Etalle Sandro, Den Hartog Jerry. Linking unlinkability. In: International symposium on trustworthy global computing. Springer; 2012, p. 129–44.

[27] Arapinis Myrto, Chothia Tom, Ritter Eike, Ryan Mark. Analysing unlinkability and anonymity using the applied pi calculus. In: 2010 23rd IEEE computer security foundations symposium. IEEE; 2010, p. 107–21.

[28] Blanchet Bruno, Smyth Ben. ProVerif 1.85: Automatic cryptographic protocol verifier, user manual and tutorial. 2011.

[29] Dolev Danny, Yao Andrew. On the security of public key protocols. IEEE Trans Inform Theory 1983;29(2):198–208.

[30] Lin Xiaodong, Sun Xiaoting, Ho Pin-Han, Shen Xuemin. GSIS: A secure and privacy-preserving protocol for vehicular communications. IEEE Trans Veh Technol 2007;56(6):3442–56.



Simone Bussa received the M.Sc. degree (summa cum laude) in computer engineering from the Politecnico di Torino, Turin, Italy, in 2021, where he is currently working towards the Ph.D. degree in control and computer engineering. His research interests include distributed systems security and formal verification applied in the field of cyber–physical systems.



Riccardo Sisto received the Ph.D. degree in Computer Engineering in 1992, from Politecnico di Torino, Italy. Since 2004, he is Full Professor of Computer Engineering at Politecnico di Torino. His main research interests are in the area of formal methods, applied to distributed software and communication protocol engineering, distributed systems, and computer security. He has authored and co-authored more than 100 scientific papers. He is a Senior Member of the ACM.



Fulvio Valenza received the M.Sc. degree (summa cum laude) and the Ph.D. degree (summa cum laude) in computer engineering from the Politecnico di Torino, Torino, Italy, in 2013 and 2017, respectively, where he is currently a Tenure-Track Assistant Professor. His research activity focuses on network security policies, orchestration and management of network security functions in SDN/NFV-based networks, and threat modeling.