An efficient certificateless aggregate signature scheme for vehicular ad-hoc networks

Avleen Kaur Malhi, Shalini Batra

Discrete Mathematics and Theoretical Computer Science DMTCS vol. **17:1**, 2015, 317–338

An Efficient Certificateless Aggregate Signature Scheme for Vehicular Ad-Hoc Networks

Avleen Kaur Malhi* Shalini Batra†

Computer Science & Engineering Dep., Thapar University, Patiala, Punjab, India

received 8th Mar. 2014, revised 21st Feb. 2015, accepted 13th Apr. 2015.

The state-of-the-art telecommunication technologies have widely been adapted for sensing the traffic related information and collection of it. Vehicular Ad-Hoc Networks (VANETs) have emerged as a novel technology for revolutionizing the driving experiences of human. The most effective and widely recognized way for mutual authentication among entities in VANETs is digital signature scheme. The new and attractive paradigm which eliminates the use of certificates in public key cryptography and solves the key escrow problem in identity based cryptography is certificateless cryptography. A new certificateless aggregate signature scheme is proposed in the paper for VANETs with constant pairing computations. Assuming the hardness of computational Diffie-Hellman Problem, the scheme is proved to be existentially unforgeable in the random oracle model against adaptive chosen-message attacks.

Keywords: VANETs, Certificateless Cryptography, Certificateless Aggregate Signature Scheme, Random Oracle Model

1 Introduction

Vehicular Ad-Hoc Networks have attracted comprehensive consideration in last few years for their assurance in enhancing driving safety and revolutionizing the transportation systems. Fundamentally, VANET security design should assure the security primitives of authentication, privacy, non-repudiation, integrity, availability, and in some peculiar application scenarios, confidentiality, to defend the network against intruders. Authentication ensures that a message is trustable by correctly identifying the sender of the message. Privacy is an important factor for the public acceptance and successful deployment of VANETs. It is attained by employing the pseudonymous approach in the communication among entities. The vehicle should not be able to relate the messages with its sender to ensure the private communication but at the same time; there should be mechanism termed as non-repudiation, to track the vehicles by law enforcement authorities, in case the vehicle transmits the wrong information in the network. Integrity requirements demand that the information from the sender to the receiver must not be altered or dropped. The availability ensures that the wireless channel has to be available so that approaching vehicles can still receive the warning messages in safety applications like post-crash warning. Confidentiality requires that

*Email: avleenmalhi@hotmail.com
†Email: sbatra@thapar.edu

the information flowing from sender to receiver should not be eavesdropped. A well recognized resolution is to sign each message with a digital signature.

Al-Riyami and Paterson (2003) invented the concept of Certificateless Public Key Cryptography (CL-PKC) which uses a third party called Key Generation Center (KGC) to generate the partial private key for an entity which is then combined with the secret key chosen by entity for the generation of full private key. Then, the user uses the public parameters of KGC and the secret key to compute the public key. The concept of partial private keys is introduced in CL-PKC as if the full private key is generated by the KGC as in Identity Based Cryptography, Baek et al. (2007), then KGC will have the full access on the private key of users and may abuse the capabilities of the network. The notion of partial private key was given in CL-PKC to ensure high security of the network. CL-PKC is considered to be well suited for VANETs in perspective of limited bandwidth and the dynamic nature of such networks. The Certificateless Signature Scheme presented by Al-Riyami and Paterson (2003) can not be used in VANETs as it employs more computational cost in signature generation and verification processes but the high mobility of vehicles in the networks puts an urgent need to reduce the computational time as much as possible to support the reliable message delivery in the highly dynamic vehicular ad-hoc networks. So, this paper majorly focuses on designing of the CLS Scheme with less computational cost but at the same time, the security of the network is also not compromised. In this paper, the Aggregate Certificateless Signature Scheme is proposed as aggregate signatures have the advantage of verifying all the signatures received from the other vehicles in an aggregate manner and reduces the signature verification time drastically because a lot of time will be consumed if the message signatures are verified independently leading to drop of important messages without being verified. Thus, the aggregate signatures enhance the network efficiency by verifying the more message signatures in a stipulated time leading to reduction in the message drop.

2 Related Work

Many security frameworks have been proposed so far to achieve the desired properties of vehicular ad hoc networks. Huang et al. (2005) proposed a security model for common certificateless signature scheme. An ID based ring signature scheme was adopted by Gamage et al. (2006) by modifying ring signature scheme with enhanced privacy, to achieve signer ambiguity for fulfilling the privacy requirement in VANET applications. However the ring signature scheme is not viable in context of VANET applications, due to unconditional privacy, resulting in requirement of non-repudiation unattainable. The schemes based on group signature are proposed in Lin et al. (2007); Lu et al. (2008); Studer et al. (2009), where the signer privacy is conditional on the group manager as the group manager possess the group master key. All these schemes have the problem of identity escrow, as a group manager can arbitrarily disclose the identity of any group member. Moreover, due to limitation of group formation, these group signature schemes are not feasible in highly mobile networks such as VANETs. An ID-based security framework for VANETs was put forward by Kamat et al. (2006, 2008) by providing authentication, non-repudiation, and privacy but it is restricted by the strong dependence on the infrastructure for short-lived pseudonym generation after each message, which renders the signalling overhead overwhelming. A user privacy preserving scheme was proposed by Sun et al. (2007) by adopting pseudonyms approach where traceability was provided simultaneously.

The first certificateless encryption scheme was given by Shamir (1985) who had given the formal proof for his scheme. Boneh et al. (2003) firstly introduced the concept of aggregate signatures. Many aggregate signature schemes have been proposed, Bagherzandi and Jarecki (2010); Boldyreva et al. (2007);

Lysyanskaya et al. (2004), since Boneh et al.'s scheme is proposed. The concept of aggregate signatures can be much efficiently used with ID-based signatures and certificateless signatures as certificateless cryptography does not pose a constraint of certificate overhead. The security model of CL Signature Schemes was given by Huang et al. (2005), but the ability of adversaries are not fully caught in CL-PKC by this model as the Certificateless scheme which is secure in this model may be insecure in actual practice. The more generic way to construct CL Signature Schemes was put forward by Yum and Lee (2004) which was proved insecure by Hu et al. (2006) and presented a new one. In addition, the security model of CLS schemes was developed by Hu et al. (2006). A new CLS scheme was propounded by Liu et al. (2007) which was proven secure in standard model. Huang et al. (2007) presented the two new constructions of the security models of certificateless signature schemes. Choi et al. (2007) presented two new efficient constructions of Certificateless Signature Schemes but Boneh et al. (2003) proved their security in weak adversary model. Du and Wen (2009) had given a new very efficient short CLS scheme with few mistakes in their security proof. A very efficient CLAS scheme was presented by Zhang and Zhang (2009) which is secure in the random oracle model. Certificateless signcryption scheme was proposed by Miao et al. (2013) but signcryption has high computational overhead in highly dynamic networks such as VANETs. The certificateless threshold signature scheme was given by Xiong et al. (2013) where the signing power was distributed among the multiple signers which is not a viable solution in VANETs. A certificateless two party authenticated key agreement protocol was proposed by He et al. (2012) but the two party communication is not viable in large scale vehicular ad-hoc networks. So far, very little attention has been devoted for the design of such Certificateless Signature Schemes for specific application scenarios such as VANETs.

2.1 Our Contribution

In this paper, a formal security model of CLS Schemes is presented and a new Certificateless Aggregate Signature Scheme for vehicular ad-hoc networks is adduced which provides the security requirements of integrity, authenticity, privacy and non-repudiation. The authenticity and integrity are provided itself by the digital signatures. Privacy is achieved by the issuance of short term identifiers called pseudonyms by the road side infrastructures which prevents the tracking of vehicles but at the same time, provides the non-repudiation as the authorities can link the pseudonyms with the actual identity of the entity. Aggregate signatures have the advantage of verifying all the signatures together leading to reduction in computation time. Assuming the hardness of computational Diffie-Hellman problem over groups in bilinear maps, the proposed CLAS scheme is proven secure in random oracle model, Bellare and Rogaway (1993).

The rest of the paper is organized as follows. Section 3 gives the preliminaries including bilinear maps, modelling CLAS scheme and the security model of CLS scheme. The setup and the new certificateless signature scheme for VANETs are given in Section 4 and the security proof is shown in Section 5. Finally, the proposed certificateless aggregate signature scheme and the security proof is presented in Section 6. The efficiency of the proposed scheme is in Section 7 and finally, Section 8 concludes the paper.

3 Preliminaries

In this section, the overview of bilinear maps is presented along with the CLAS modelling and adversarial model for CLS schemes.

3.1 Bilinear Maps

Let a cyclic additive group $\mathbb{G}_1$ be of prime order q having a generator point P. Let a cyclic multiplicative group $\mathbb{G}_2$ be of the same order q. The map $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ is considered as a bilinear pairing if the following properties are satisfied:

1. Bilinearity: For any $P,X,Y \in \mathbb{G}_1$, $e(P,X+Y) = e(P,X)e(P,Y)$ and for any $a,b \in \mathbb{Z}_q^*$, $e(aP,bP) = e(P,P)^{ab} = e(abP,P) = e(P,abP)$
2. Non-degenerate: $e(P,P) \neq 1\mathbb{G}_2$
3. Computability: There exists $P,R \in \mathbb{G}_1$, an efficient algorithm should exist to compute $e(P,R)$ for all $P,R \in \mathbb{G}_1$.

The proposed certificateless signature scheme's security depends on the hardness of the Computational Diffie-Hellman (CDH) problem in the group.

Computational Diffie-Hellman (CDH) problem: Let cyclic group $\mathbb{G}_1$ with order q has a generator P, and given two points, aP and bP for unknown $a,b \in \mathbb{Z}_q^*$, then to compute abP is computationally infeasible.

3.2 Modelling Certificateless Aggregate Signature Scheme

- **Setup:** It takes security parameter 1^ℓ as input, then it generates and publishes a list of system parameters as *params*.
- **PartialKeyGen:** This algorithm is performed by KGC once for each vehicle as it enters the region of new RTA (Regional Transportation Authority). It takes the inputs *params*, masterkey and identity ID_i to generate the partial private key pp_i.
- **UserKeyGen:** The algorithm is run by user that takes the user identity ID_i as input and selects a random $x_i \in \mathbb{Z}_q^*$, to output the secret key as x_i and public key as P_i.
- **Sign:** The algorithm is run by the user with inputs as *params*, Identity ID_i, secret key x_i, partial private key pp_i, public key P_i and message m_k to produce the output as signature σ_{ik} on message m_k generated by user with identity ID_i.
- **Aggregate:** The aggregate signature generator runs this algorithm by aggregating the signatures of n users $\{\mathscr{U}_1.....\mathscr{U}_n\}$ with public keys $\{\mathscr{P}_1.....\mathscr{P}_n\}$, messages $\{\mathscr{M}_1.....\mathscr{M}_n\}$ and signatures $\{\sigma_1.....\sigma_n\}$ to generate the aggregate signature σ as output.
- **AggregateVerify:** This algorithm takes as input an aggregate signature σ which is signed by n users $\{\mathscr{U}_1.....\mathscr{U}_n\}$ with public keys $\{\mathscr{P}_1.....\mathscr{P}_n\}$ on messages $\{\mathscr{M}_1.....\mathscr{M}_n\}$. It outputs true if the aggregate signature is valid or false otherwise.

3.3 Adversarial Model of Certificateless Signature Schemes

This section reviews the adversarial model, Hu et al. (2007) of Certificateless Signature Scheme. Two types of adversaries are considered in Certificateless Cryptography namely, Type I and Type II adversary. Let $\mathscr{A}_1$ denote a Type I attacker and $\mathscr{A}_2$ denote a Type II attacker. Two games are considered, "Game I" where challenger $\mathscr{C}$ interacts with adversary $\mathscr{A}_1$ and "Game II" where $\mathscr{C}$ interacts with adversary $\mathscr{A}_2$. The master key of KGC cannot be accessed by adversary $\mathscr{A}_1$, but the public key of any entity can be replaced by $\mathscr{A}_1$ with the value chosen by it whereas the master key of KGC can be accessed by the adversary $\mathscr{A}_2$ but $\mathscr{A}_2$ cannot perform public key replacement. The Certificateless Signature Scheme is existentially unforgeable against the adaptive chosen message attack, if the both adversaries $\mathscr{A}_1$ and $\mathscr{A}_2$ have negligible success probability. The following six oracles can be accessed by the adversaries.

- **RevealPartialKey:** The adversary requests the partial private key of any vehicle with identity ID_i. The challenger $\mathscr{C}$ searches the list L and returns the partial private key pp_i corresponding to the ID_i. If not found, $\perp$ is returned.

- **RevealSecretKey:** The adversary requests the secret key of a vehicle with identity ID_i. The challenger responds with the corresponding secret key x_i if the entry exists in the list L. Otherwise, it outputs $\perp$.

- **RevealPublicKey:** The adversary can request the public key of the vehicle with identity ID_i. The challenger responds with the public key P_i. If the entry corresponding to ID_i does not exist, $\perp$ is returned.

- **RevealPseudonym:** The adversary makes a request for the pseudonym of the vehicle with ID_i. The challenger searches the list L and responds with PS_j if the entry exists, otherwise it returns $\perp$.

- **ReplacePublicKey:** On input an identity ID_i, the adversary may replace the public key P_i of the vehicle with the public key of its own choice, say $P_i^{'}$. If the list L does not contain the entry corresponding to ID_i, then nothing is done.

- **Sign:** On input a message $\mathscr{M}_i \in \{0,1\}^*$, the adversary can request the signature corresponding to the identity ID_i. A valid signature σ_i is returned if the entry corresponding to the ID_i exists in the list L. If the public key corresponding to ID_i has been replaced with $P_i^{'}$, then the challenger $\mathscr{C}$ returns the valid signature with the new public key $P_i^{'}$ and secret key $x_i^{'}$. If the list L does not contain the entry corresponding to ID_i, then a symbol $\perp$ is returned.

Game I (for $\mathscr{A}_1$ adversary): In this game, adversary $\mathscr{A}_1$ interacts with the challenger $\mathscr{C}$.

Phase I-1: The Setup algorithm is run by challenger $\mathscr{C}$, which takes the input security parameter ℓ to generate the *master key* and system parameter list *params*. The parameter list *params* is sent by challenger $\mathscr{C}$ to the adversary $\mathscr{A}_1$ while keeps the *master key* as secret to itself.

Phase I-2: The polynomially bounded number of oracle-query operations are performed by the adversary $\mathscr{A}_1$. The adversary $\mathscr{A}_1$ can make *RevealPartialKey*, *RevealSecretKey*, *RevealPublicKey*, *RevealPseudonym*, *ReplacePublicKey* and *Sign* queries onto oracle during this stage of simulation.

Phase I-3: Finally, $\mathscr{A}_1$ outputs a message and signature pair $< m_i^*, \sigma_i^* >$ corresponding to the identity ID_i^* with a public key P_i^*.

Now, $\mathscr{A}_1$ wins Game I if;

- σ_i^* is a valid signature on message m_i^* with identity ID_i^* and pubic key P_i^*.
- The identity ID_i^* has not queried partial private key pp_i^* during *RevealPartialKey* queries. Moreover, oracle *Sign* has never been queried with ID_i^* and m_i^*.

Game II (for $\mathscr{A}_2$ adversary): In this game, adversary $\mathscr{A}_2$ interacts with the challenger $\mathscr{C}$.

Phase II-1: The Setup algorithm is run by challenger $\mathscr{C}$, which takes the input security parameter ℓ for the generation of the *master key* and system parameter list *params*. The parameter list *params* and the *master key* both are sent by challenger $\mathscr{C}$ to the adversary $\mathscr{A}_2$.

Phase II-2: The polynomially bounded number of oracle-query operations are performed by the adversary $\mathscr{A}_2$. The adversary $\mathscr{A}_2$ can make *RevealSecretKey*, *RevealPublicKey*, *RevealPseudonym* and *Sign* queries onto oracle during this stage of simulation. The oracle *RevealPartialKey* is no longer needed by $\mathscr{A}_2$ as $\mathscr{A}_2$ has the access to the master key.

Phase II-3: Finally, $\mathscr{A}_2$ outputs a message and signature pair $< m_i^*, \sigma_i^* >$ corresponding to the identity ID_i^* with a public key P_i^*.

Now, $\mathscr{A}_2$ wins Game II if;

- σ_i^* is a valid signature on message m_i^* with identity ID_i^* and public key P_i^*.
- The identity ID_i^* has not queried secret key x_i^* during *RevealSecretKey* queries. Moreover, oracle *Sign* has never been queried with ID_i^* and m_i^*.

Definition 1: A Certificateless Signature scheme is existentially unforgeable under adaptively chosen message attack if the success probability $succ_{\mathscr{A}}(\ell)$ of any probabilistic polynomial time (PPT) adversary $\mathscr{A}$ in any of the above two games is negligible.

4 New Efficient Certificateless Signature Scheme for VANETs

Here, an efficient certificateless signature scheme based on bilinear pairings is presented. The proposed certificateless signature scheme comprises the following seven algorithms.

Setup

There is one Key Generation Center (KGC) located in the region under one Regional Transportation Authority (RTA). The input is a security parameter 1^ℓ where $\ell \in N$, firstly a cyclic additive group $\mathbb{G}_1$ of prime order q is chosen by KGC. And finally, the cyclic multiplicative group $\mathbb{G}_2$ of the same prime order q is chosen. The bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ is defined by it. Choose a generator point of the group $\mathbb{G}_1$ as $P \in \mathbb{G}_1$. A master key $s \in_R \mathbb{Z}_q^*$ is uniformally selected and the public key of KGC as $P_{pub} = s \cdot P$ is selected. The two distinct hash functions $H_2 : \{0,1\}^* \longrightarrow \mathbb{Z}_q^*$ and $H_3 : \{0,1\}^* \longrightarrow \mathbb{Z}_q^*$ are chosen. The message space is defined as $\mathscr{M} = \{0,1\}^*$. Each RSU in the region under RTA sets $y_i \in \mathbb{Z}_q^*$ as the secret key of RSU_i and $P_{rsu_i} = y_i.P$ as the public key of RSU_i. The public keys of all the RSUs $P_{rsu_1}, P_{rsu_2}, P_{rsu_3}, \ldots, P_{rsu_n}$ under the region of RTA are sent to the KGC and published under *params* list. The system parameter list is defined as $params = \{\mathbb{G}_1, \mathbb{G}_2, e, P, P_{pub}, H_2, H_3, P_{rsu_1}, P_{rsu_2}, P_{rsu_3}, \ldots, P_{rsu_n}\}$.

Registration

This algorithm is run by RTA to register the vehicle with Identity ID_i. RTA maps the relationship between ID_i and Q_{ID_i} as the actual identity of the vehicle ID_i is concealed and Q_{ID_i} is only used as the identity by the vehicle for all the communications. Whenever the law enforcement authorities want to trace the vehicle for liability issues, then RTA can reveal the actual identity of the vehicle. When the vehicle enters the region of another RTA, then the vehicle again registers its ID_i to get the new pseudo identity.

- Choose the distinct hash function $H_1 : \{0,1\}^* \longrightarrow \mathbb{G}_1$.
- The vehicle's identity space is $ID_i \in \{0,1\}^*$. The vehicle with identity ID_i registers itself with RTA (Regional Transportation Authority) as $Q_{ID_i} = H_1(ID_i) \in \mathbb{G}_1$.
- The RTA sends the Q_{ID_i} to the vehicle.

PartialKeyGen

In the region under single RTA, there is single KGC and partial private key is generated once for each vehicle under the region of one RTA. All the RSUs in the region are directly under the control of RTA of that region. It is assumed that the KGC and RTA do not collude as both of these are different authorities and RTA does not have any authority over KGC. It implies that the KGC and RSUs do not collude with each other.

- KGC runs this algorithm, and takes the inputs parameter list *params*, master key s and $Q_{ID_i} \in \mathbb{G}_1$.
- KGC then generates the partial private key of the vehicle as $pp_i = s.Q_{ID_i}$ by using the identity Q_{ID_i}.

It can be seen that the partial private key pp_i of vehicle is a signature on Q_{ID_i} with the public/private key pair (P_{pub}, s) and the correctness of the signature is checked by vehicle by checking whether $e(pp_i, P) = e(Q_{ID_i}, P_{pub})$ which can be verified as

$$e(pp_i, P) = e(s.Q_{ID_i}, P) = e(Q_{ID_i}, s.P) = e(Q_{ID_i}, P_{pub})$$

UserKeyGen

This algorithm is used to generate the secret as well as public keys of the vehicles for vehicular communications in VANETs.

- The vehicle updates the secret and public key each time it enters the region of new RTA as it gets the new pseudo identity Q_{ID_i}.
- The vehicle and RSU takes as input *params*, to generate the secret and public keys. The vehicle with identity Q_{ID_i} selects $x_i \in_R \mathbb{Z}_q^*$ at random and sets x_i as the secret key of vehicle and sets the public key of vehicle as $P_i = x_i.P$.

PseudonymGen

This algorithm is run by the corresponding RSU_i under whose coverage is the vehicle requesting for the pseudonym. The pseudonyms are allocated to vehicles each time a new RSU is encountered. The frequent updation of pseudonyms under each RSU may lead to consumption of network bandwidth and signalling overhead problem. Therefore, the Road Side Units (RSU) may combine to form the autonomous networks to help solve the signalling overhead problem and bandwidth consumption which may be caused due to the frequently updating of the pseudonyms under each RSU.

- It is assumed that autonomous network is comprised of 4 RSUs in scarcely populated areas and 2 RSUs in densely populated areas. Pseudonym is generated once under one autonomous network. This solution provides privacy and liability at the same time.
- The inputs taken by this algorithm are *params* and vehicle identity Q_{ID_i} and generates the pseudonym of the vehicle in two parts $PS1_j$ and $PS2_j$ such that $PS_j = PS1_j + PS2_j$.
- The autonomous network RSU_i selects $a_j \in_R \mathbb{Z}_q^*$ at random and sets $PS1_j = a_j.Q_{ID_i}$.
- Then it calculates the hash value of $PS1_j$ as $T_j = H_3(PS1_j) \in \mathbb{Z}_q^*$.
- The second part of pseudonym $PS2_j$ is calculated as $PS2_j = a_j.T_j$.
- Finally the pseudonym PS_j is calculated as $PS_j = PS1_j + PS2_j$.

Sign

To sign a message $m_k \in \mathscr{M}$ using the partial private key and private key pair (pp_i, x_i) with vehicle identity Q_{ID_i} and public key P_i, the following steps are performed:

- Choose a random $r_i \in_R \mathbb{Z}_q^*$ and compute $U_i = r_i.P \in \mathbb{G}_1$.
- Compute $h_{ijk} = H_2(m_k, PS1_j, P_i, U_i) \in \mathbb{Z}_q^*$.
- Compute $V_{ijk} = pp_i.PS2_j + h_{ijk}.r_i.P_{pub} + h_{ijk}.x_i.P_{rsu_i}$.
- Output the signature on m_k as $\sigma_{ijk} = (U_i, V_{ijk})$.

The first scalar multiplication $(pp_i.PS2_j)$ in V_{ijk} can be pre-computed whenever the pseudonym is generated for Q_{ID_i} in the current autonomous network as it saves one scalar multiplication during signature generation.

Verify

To verify the signature $\sigma_{ijk} = (U_i, V_{ijk})$ signed by the vehicle with pseudonym PS_j, given are the *params*, pseudonym $PS1_j$, the public key P_i, the message m_k and the signature $\sigma_{ijk} = (U_i, V_{ijk})$, the vehicle:

- Computes $h_{ijk} = H_2(m_k, PS1_j, P_i, U_i)$.
- Computes $T_j = H_3(PS1_j)$.

- The signatures are accepted if the following equation holds:

$$e(V_{ijk},P) = e(PS1_j.T_j + h_{ijk}.U_i, P_{pub})e(h_{ijk}.P_i, P_{rsu_i})$$

- If the equation holds, output is true; otherwise, output is $\perp$.

The correctness of the scheme follows from the fact:

$$\begin{aligned} e(V_{ijk},P) &= e(pp_i.PS2_j + h_{ijk}.r_i.P_{pub} + h_{ijk}.x_i.P_{rsu_i}, P) \\ &= e(pp_i.PS2_j, P)e(h_{ijk}.r_i.P_{pub}, P)e(h_{ijk}.x_i.P_{rsu_i}, P) \\ &= e(s.Q_{ID_i}.a_j.T_j, P)e(h_{ijk}.r_i.s.P, P)e(h_{ijk}.x_i.y_i.P, P) \\ &= e(a_j.Q_{ID_i}.T_j, s.P)e(h_{ijk}.r_i.P, s.P)e(h_{ijk}.x_i.P, y_i.P) \\ &= e(PS1_j.T_j, P_{pub})e(h_{ijk}.U_i, P_{pub})e(h_{ijk}.P_i, P_{rsu_i}) \\ &= e(PS1_j.T_j + h_{ijk}.U_i, P_{pub})e(h_{ijk}.P_i, P_{rsu_i}) \end{aligned}$$

The current setup allows the KGC to choose a secret key $x_i^{'} \in \mathbb{Z}_q^*$ and compute new public key of user $P_i^{'} = x_i^{'}.P$ with identity Q_{ID_i}. Therefore, KGC is able to know both the partial private key and the private key of the vehicle which implies that both KGC and vehicle may deny signature generation. But in this scheme, the pseudonyms are generated by the RSUs and thus RSU verify the user identity and then generates the pseudonym corresponding to that identity. The signatures are generated with the pseudonyms used for those identities. So KGC does not have access to pseudonyms and it cannot forge the signatures. Moreover, if public key is replaced by KGC, it can easily be detected by law enforcement authorities as the KGC is the only entity having that capability.

5 Security Proof

Assuming the hardness of Computational Diffie-Hellman problem, the security of the Certificateless Signature Scheme is hereby shown.

Theorem 1 *In the random oracle model, an adversary $\mathscr{A}_1$ exists having an advantage ε to forge a signature in a game I modelled attack within a time span t and performs queries to various oracles by making q_i queries to H_i for $i = 1,2,3$, q_k queries to RevealPartialKey, q_s queries to the RevealSecretKey, q_p queries to RevealPublicKey, q_{ps} queries to RevealPseudonym and q_{sig} queries to sign, then CDH problem can be solved in time*

$$t + \varphi(q_1 + q_2 + q_3 + q_k + q_s + q_p + q_{ps} + q_{sig})t_m$$

where t_m is computation time for scalar multiplication in $\mathbb{G}_1$ with probability

$$\varepsilon^{'} \geq \frac{1}{(q_k+1).e}\varepsilon$$

Proof: Let $\mathscr{C}$ be an attacker receiving a random instance (P, aP, bP) of the CDH problem in cyclic group $\mathbb{G}_1$. Point P is a generator of $\mathbb{G}_1$ having prime order q. Now, $X = a.P$ and $Y = b.P$ where a and b are randomly chosen in ${}_R\mathbb{Z}_q^*$. A type I adversary $\mathscr{A}_1$ interacts with $\mathscr{C}$ as modelled in Game I. $\mathscr{C}$ uses $\mathscr{A}_1$ for

solving the CDH problem by computing abP in $\mathbb{G}_1$ with the construction of an algorithm S_1. $\mathscr{C}$ sends the params = $(\mathbb{G}_1, \mathbb{G}_2, e, P, P_{pub}, H_2, H_3, P_{rsu})$ to $\mathscr{A}_1$.

S_1 chooses random $c \in_R \mathbb{Z}_q^*$ and sets $P_{pub} = X$ and $P_{rsu} = c.P$ and then start performing oracle queries. The hash functions H_1, H_2 and H_3 are considered random oracles and $\mathscr{A}_1$ performs the following queries. It is assumed that $H_1(.)$ oracle query has been previously made on that identity for which key extraction or signature query has been made. A list $L = (ID_i, pp_i, x_i, P_i, PS_j)$ is maintained by S_1 while $\mathscr{A}_1$ makes queries throughout the Game I and $\mathscr{C}$ maintains S_1 algorithm. S_1 responds to all the $\mathscr{A}_1$ queries.

H_1 **queries:** After an identity ID_i is being submitted to oracle H_1, same answer will be given if the request has been asked before. Otherwise, S_1 flips a coin $c_i \in \{0,1\}$ yielding 0 with probability ζ and yielding 1 with probability $(1-\zeta)$. It then randomly picks $\alpha_i \in_R \mathbb{Z}_q^*$. If $c_i = 0$, the $H_1(ID_i)$ is defined as $Q_i = \alpha_i.Y \in \mathbb{G}_1$. If $c_i = 1$, then $H_1(ID_i) = \alpha_i.P \in \mathbb{G}_1$. In both cases, S_1 inserts a tuple $(ID_i, \alpha_i, c_i, Q_i)$ in a list $L_{H_1} = (ID_i, \alpha_i, c_i, Q_i)$ to keep the track of the queries.

H_2 **queries:** S_1 maintains a list $L_{H_2} = (m_k, PS1_j, P_i, U_i, h_{ijk})$ which is initially empty. When $\mathscr{A}_1$ issues a query $H_2(m_k, PS1_j, P_i, U_i)$, the same answer from the list L_{H_2} will be given if the request has been previously made. Otherwise S_1 selects random element $h_{ijk} \in_R \mathbb{Z}_q^*$ and adds the tuple $(m_k, PS1_j, P_i, U_i, h_{ijk})$ to the list L_{H_2} and returns h_{ijk} as the answer to the hash value of $H_2(m_k, PS1_j, P_i, U_i)$ to $\mathscr{A}_1$.

H_3 **queries:** S_1 maintains a list $L_{H_3} = (PS1_j, t1_j)$ which is initially empty. When $\mathscr{A}_1$ issues a query $H_3(PS1_j)$, same answer will be returned if the query has been previously made. Otherwise, S_1 selects a random $t1_j \in_R \mathbb{Z}_q^*$ and adds $(PS1_j, t1_j)$ to the list L_{H_3} and return $t1_j$ as answer to $\mathscr{A}_1$.

RevealPseudonym queries: The request is issued on an identity ID_i.

- The corresponding tuple $(ID_i, pp_i, x_i, P_i, PS_j)$ is recovered from the list L. S_1 checks if PS_j is $\perp$. If $PS_j \neq \perp, S_1$ returns PS_j to $\mathscr{A}_1$. Otherwise, S_1 randomly chooses $k_j \in_R \mathbb{Z}_q^*$ and computes $PS1_j = k_j.Q_i \in \mathbb{G}_1$ with ID_i corresponding to the list $L_{H_1} = (ID_i, \alpha_i, c_i, Q_i)$ if $c_i = 1$ and $PS2_j = k_j.\ t1_j$ where $t1_j$ corresponds to the list $L_{H_3} = (PS1_j, t1_j)$. S_1 returns $PS_j = (PS1_j + PS2_j)$ to the adversary $\mathscr{A}_1$ and adds $(ID_i, pp_i, x_i, P_i, PS_j)$ to the list L.

- If the list L does not contain $(ID_i, pp_i, x_i, P_i, PS_j)$, then S_1 sets $PS_j = \perp$ and then randomly chooses $k_j \in_R \mathbb{Z}_q^*$ and computes $PS1_j = k_j.Q_i$ and $PS2_j = k_j.t1_j$ from corresponding lists $L_{H_1} = (ID_i, \alpha_i, c_i, Q_i)$ and $L_{H_3} = (PS1_j, t1_j)$ respectively. S_1 answers $PS_j = (PS1_j + PS2_j)$ to the adversary $\mathscr{A}_1$ and adds $(ID_i, pp_i, x_i, P_i, PS_j)$ to the list L.

RevealPartialKey queries: The request is issued on an identity ID_i.

- The corresponding tuple $(ID_i, \alpha_i, c_i, Q_i)$ is recovered from the list L_{H_1}. If $c_i = 0$, then S_1 returns failure.

- If $c_i = 1$, and if list L contains $(ID_i, pp_i, x_i, P_i, PS_j)$, S_1 checks if $pp_i = \perp$. If $pp_i \neq \perp$, then S_1 returns pp_i to S_1. If $pp_i = \perp$, S_1 recovers the tuple $(ID_i, \alpha_i, c_i, Q_i)$ from the list L_{H_1}. Now, $c_i = 1$ i.e. $H_1(ID_i)$ is defined as $\alpha_i.P \in \mathbb{G}_1$ and $pp_i = \alpha_i.P_{pub} = \alpha_i.X \in \mathbb{G}_1$. S_1 returns partial private key pp_i to $\mathscr{A}_1$ and adds tuple $(ID_i, pp_i, x_i, P_i, PS_j)$ into list L.

- Again if $c_i = 1$, and list L does not contain the tuple $(ID_i, pp_i, x_i, P_i, PS_j)$, S_1 sets $pp_i = \perp$ and recovers $(ID_i, \alpha_i, c_i, Q_i)$ from the list L_{H_1}. S_1 sets $pp_i = \alpha_i.P_{pub} = \alpha_i.X \in \mathbb{G}_1$ and returns pp_i to $\mathscr{A}_1$ and adds $(ID_i, pp_i, x_i, P_i, PS_j)$ into the list L.

RevealPublicKey queries: The request is issued on an identity ID_i.

- If the list L contains $(ID_i, pp_i, x_i, P_i, PS_j)$, S_1 checks whether $P_i = \perp$. If $P_i \neq \perp$, then S_1 returns P_i to S_1. Otherwise if $P_i = \perp$, S_1 randomly selects $v_i \in_R \mathbb{Z}_q^*$ and $P_i = v_i.P \in \mathbb{G}_1$ and $x_i = v_i$. S_1 returns P_i as answer to $\mathscr{A}_1$ and saves (P_i, x_i) into the list L.
- If the list L does not contain $(ID_i, pp_i, x_i, P_i, PS_j)$, S_1 sets $P_i = \perp$ and sets $x_i = v_i$, $P_i = v_i.P \in \mathbb{G}_1$; $v_i \in_R \mathbb{Z}_q^*$ and returns P_i to $\mathscr{A}_1$ and adds the tuple $(ID_i, pp_i, x_i, P_i, PS_j)$ to the list L.

RevealSecretKey queries: The request is issued on an identity ID_i.

- If the list L contains $(ID_i, pp_i, x_i, P_i, PS_j)$, then S_1 checks if $x_i = \perp$. If $x_i \neq \perp$, then S_1 returns x_i to $\mathscr{A}_1$. Otherwise, S_1 selects at random $v_i \in_R \mathbb{Z}_q^*$ and sets $x_i = v_i$ and $P_i = v_i.P \in \mathbb{G}_1$. It returns x_i to $\mathscr{A}_1$ and adds $(ID_i, pp_i, x_i, P_i, PS_j)$ into the list L.
- If the list L does not contain $(ID_i, pp_i, x_i, P_i, PS_j)$, then S_1 sets $x_i = \perp$ and selects randomly $v_i \in_R \mathbb{Z}_q^*$ and $P_i = v_i.P \in \mathbb{G}_1$ and $x_i = v_i$. It returns x_i to $\mathscr{A}_1$ and adds $(ID_i, pp_i, x_i, P_i, PS_j)$ into the list L.

ReplacePublicKey queries: Suppose $\mathscr{A}_1$ chooses new public key $P_i^{'}$ for an identity ID_i.

- S_1 searches the list L and if L contains an element $(ID_i, pp_i, x_i, P_i, PS_j)$, S_1 replaces P_i with $P_i^{'}$ and sets $x_i = \perp$.
- If the list L does not contain $(ID_i, pp_i, x_i, P_i, PS_j)$, then S_1 sets $P_i = P_i^{'}$ and $x_i = \perp$. It also sets $pp_i = \perp$ and adds $(ID_i, pp_i, x_i, P_i, PS_j)$ into the list L.

Sign queries: On receiving a sign query on ID_i, S_1 firstly recovers L_{H_1} list as $(ID_i, \alpha_i, c_i, Q_i)$, then list L as $(ID_i, pp_i, x_i, P_i, PS_j)$, list L_{H_2} as $(m_k, PS1_j, P_i, U_i, h_{ijk})$ and list L_{H_3} as $(PS1_j, t1_j)$ and generates the signature as follows:

- If $c_i = 1$ and if the list L contains $(ID_i, pp_i, x_i, P_i, PS_j)$, S_1 checks if $x_i = \perp$. If $x_i = \perp$, S_1 makes *RevealPublicKey* query to generate $x_i = v_i$ and $P_i = v_i.P$.
- If L does not contain $(ID_i, pp_i, x_i, P_i, PS_j)$, S_1 makes *RevealPublicKey* query to generate x_i and P_i and adds it to list L.
- To generate the signature, if $c_i = 0$, choose r_i, $\gamma_i \in_R \mathbb{Z}_q^*$, set $U_i = r_i.P - h_{ijk}^{-1}.PS1_j.t1_j$ and computes

$$\begin{aligned} V_{ijk} &= pp_i.PS2_j + h_{ijk}.r_i.P_{pub} + h_{ijk}.x_i.P_{rsu} \\ &= h_{ijk}.r_i.X + h_{ijk}.x_i.P_{rsu} \end{aligned}$$

  The output σ_{ijk} can be expressed as $\sigma_{ijk} = (U_i, V_{ijk})$

- If $c_i = 1$, to generate the signature, choose at random $r_i \in_R \mathbb{Z}_q^*$ and $U_i = r_i.P \in \mathbb{G}_1$ and set the values $pp_i = \alpha_i.X$, $PS2_j = k_j.t1_j$, $P_{rsu} = c.P$. It computes the value of V_{ijk} as

$$V_{ijk} = \alpha_i.X.k_j.t1_j + h_{ijk}.r_i.X + h_{ijk}.x_i.c.P$$

If the tuple containing h_{ijk} already appears in the list L_{H_2}, then S_1 selects another $r_i, h_{ijk} \in_R \mathbb{Z}_q^*$ and tries again. Finally S_1 responds to $\mathscr{A}_1$ with $\sigma_{ijk} = (U_i, V_{ijk})$. All the responses to *Sign* queries are valid, i.e. the output (U_i, V_{ijk}) of sign query is a valid signature generated on message m_k.

Now, when $c_i = 0$, $V_{ijk} = h_{ijk}.r_i.X + h_{ijk}.x_i.P_{rsu}$

$$\begin{aligned} e(V,P) &= e(h_{ijk}.r_i.X + h_{ijk}.x_i.P_{rsu}, P) \\ &= e(h_{ijk}.r_i.a.P, P)e(h_{ijk}.x_i.c.P, P) \\ &= e(h_{ijk}.r_i.P, aP)e(h_{ijk}.x_i.P, c.P) \\ &= e(h_{ijk}.U_i + PS1_j.t_{1_j}, X)e(h_{ijk}.P_i, P_{rsu}) \end{aligned}$$

By forking lemma, Shamir (1985), replaying $\mathscr{A}_1$ with some random tape, S_1 obtains two valid signatures $\sigma^*_{ijk}(ID^*_i, m^*_k, h^*_{ijk}, U^*_i, V^*_{ijk})$ and $\sigma^{'}_{ijk}(ID^*_i, m^*_k, h^{'}_{ijk}, U^*_i, V^{'}_{ijk})$ within polynomial time, where

$$V^*_{ijk} = pp^*_i.PS2^*_j + h^*_{ijk}.r^*_i.P^*_{pub} + h^*_{ijk}.x^*_i.P^*_{rsu} \quad (1)$$

$$V^{'*}_{ijk} = pp^*_i.PS2^*_j + h^{'*}_{ijk}.r^*_i.P^*_{pub} + h^{'*}_{ijk}.x^*_i.P^*_{rsu} \quad (2)$$

Multiplying both sides of equation (1) by $(h^*_{ijk})^{-1}$ and both sides of equation (2) by $(h^{'*}_{ijk})^{-1}$

$$(h^*_{ijk})^{-1}V^*_{ijk} = (h^*_{ijk})^{-1}pp^*_i.PS2^*_j + (h^*_{ijk})^{-1}h^*_{ijk}(r^*_i.P^*_{pub} + x^*_i.P^*_{rsu}) \quad (3)$$

$$(h^{'*}_{ijk})^{-1}V^{'*}_{ijk} = (h^{'*}_{ijk})^{-1}pp^*_i.PS2^*_j + (h^{'*}_{ijk})^{-1}h^{'*}_{ijk}(r^*_i.P^*_{pub} + x^*_i.P^*_{rsu}) \quad (4)$$

Subtracting (4) from (3)

$$(h^*_{ijk})^{-1}V^*_{ijk} - (h^{'*}_{ijk})^{-1}V^{'*}_{ijk} = [(h^*_{ijk})^{-1} - (h^{'*}_{ijk})^{-1}]pp^*_i.PS2^*_j \quad (5)$$

Then, S_1 recovers the corresponding $(ID_i, \alpha_i, c_i, Q_i)$ from the list L_{H_1}. If $c_i = 1$, S_1 aborts. Otherwise, if $c_i = 0$, $Q_i = \alpha_i.Y = \alpha_i.b.P$, $PS2_j = k_j.t_{1_j}$. Now $P_{pub} = a.P$, where a is the secret key of KGC. Then, $pp_i = a.Q_i = a(\alpha_i.b.P) = \alpha_i.abP$. Now, in equation (5),

$$(h^*_{ijk})^{-1}V^*_{ijk} - (h^{'*}_{ijk})^{-1}V^{'*}_{ijk} = [(h^*_{ijk})^{-1} - (h^{'*}_{ijk})^{-1}]\alpha_i.abP.k_j.t_{1_j}.$$

$$abP = [(h^*_{ijk})^{-1}V^*_{ijk} - (h^{'*}_{ijk})^{-1}V^{'*}_{ijk}][\alpha_i.k_j.t_{1_j}.((h^*_{ijk})^{-1} - (h^{'*}_{ijk})^{-1})]^{-1}$$

Thus, algorithm S_1 outputs abP as the solution to Computational Diffie Hellman problem.

The proof is completed by showing that S_1 solves the given instance of CDH problem with the probability

$$\varepsilon^{'} \geq \frac{1}{(q_k+1).e}\varepsilon$$

S_1 needs three events in order to succeed:

E_1**:** The result of any $\mathscr{A}_1$'s *RevealPartialKey* queries does not abort S_1.

E_2**:** A valid and non trivial signature is generated by $\mathscr{A}_1$.

E_3**:** Probability that $\mathscr{A}_1$ outputs a valid and nontrivial forgery and S_1 does not abort.

Probability that S_1 succeeds after all these events happen is

$$P[E_1 \wedge E_2 \wedge E_3] = P[E_1] \cdot P[E_2|E_1] \cdot P[E_3|E_1 \wedge E_2]$$

- Claim 1: Probability that result of any $\mathscr{A}_1$'s *RevealPartialKey* queries does not abort S_1 is atleast $(1-\zeta)^{q_k}$. Hence, P$[E_1] \geq (1-\zeta)^{q_k}$
  Proof: $P[c_i = 1] = (1-\zeta)$, therefore probability that S_1 does not abort is $(1-\zeta)$. As it takes atmost q_k times *RevealPartialKey* queries, probability of S_1 not aborting after the queries of $\mathscr{A}_1$ is atleast $(1-\zeta)^{q_k}$.
- Claim 2: The probability of S_1 not aborting with $\mathscr{A}_1$'s signature queries and key extraction queries is ε. $P[E_2|E_1] \geq \varepsilon$.
- Claim 3: Probability that $\mathscr{A}_1$ outputs a valid and nontrivial forgery and S_1 does not abort is ζ. $P[E_3|E_1 \wedge E_2] \geq \zeta$
  Proof: Suppose $\mathscr{A}_1$ generated a valid and nontrivial forgery after the events E_1 and E_2 occurred. Thus, $P[E_3|E_1 \wedge E2] \geq \zeta$.

Thus,

$$\begin{aligned} P[E_1 \wedge E_2 \wedge E_3] &= P[E_1] \cdot P[E_2|E_1] \cdot P[E_3|E_1 \wedge E_2] \\ &= (1-\zeta)^{q_k} \cdot \varepsilon \cdot \zeta \\ &= \zeta(1-\zeta)^{q_k} \cdot \varepsilon \end{aligned}$$

Now, $\zeta_{opt} = \frac{1}{q_k+1}$. Thus,

$$\begin{aligned} \varepsilon' &\geq \zeta(1-\zeta)^{q_k} \cdot \varepsilon \\ \varepsilon' &\geq \frac{1}{q_k+1}[1-\frac{1}{q_k+1}]^{q_k} \cdot \varepsilon \end{aligned}$$

With sufficiently large q_k, the term $\left[1-\frac{1}{q_k+1}\right]^{q_k}$ tends to $\frac{1}{e}$. Thus the probability is,

$$\varepsilon' \geq \frac{1}{(q_k+1).e}\varepsilon$$

□

Theorem 2 *In the random oracle model, an adversary $\mathscr{A}_2$ exists having an advantage ε to forge a signature in a game II modelled attack within a time span t and performs queries to various oracles by making q_2 queries to H_2, q_3 queries to H_3, q_p queries to RevealPublicKey, q_s queries to RevealSecretKey, q_{ps} queries to RevealPseudonym and q_{sig} queries to sign, then the CDH problem in $\mathbb{G}_1$ can be solved in time*

$$t + \varphi(q_2+q_3+q_s+q_p+q_{ps}+q_{sig})t_m$$

where t_m is the computational time for scalar multiplication in $\mathbb{G}_1$ with probability

$$\varepsilon' \geq \frac{1}{(q_p+1).e}\varepsilon$$

Proof: Let $\mathscr{C}$ be an attacker receiving a random instance (P,aP,bP) of the CDH problem in cyclic group $\mathbb{G}_1$. Point P is a generator of $\mathbb{G}_1$ having prime order q. X = a.P and Y = b.P where a and b are randomly chosen in ${}_R\mathbb{Z}_q^*$. A type II adversary $\mathscr{A}_2$ interacts with $\mathscr{C}$ as modelled in Game II. $\mathscr{C}$ uses $\mathscr{A}_2$ for solving the CDH problem by computing abP in $\mathbb{G}_1$ with the construction of an algorithm S_2. $\mathscr{C}$ sets $P_{rsu} = aP = X$. S_2 randomly sets the master key of KGC as $\lambda \in {}_R\mathbb{Z}_q^*$ and sets the public key of KGC as $P_{pub} = \lambda.P$. S_2 then sends the system parameters *params* $(\mathbb{G}_1, \mathbb{G}_2, e, P, P_{pub}, H_2, H_3, P_{rsu})$ to $\mathscr{A}_2$. The adversary $\mathscr{A}_2$ is also provided the master key λ by S_2. Now, $\mathscr{A}_2$ has the master key so it can generate the partial public key itself. $\mathscr{C}$ maintains S_2 algorithm and S_2 responds to all the $\mathscr{A}_2$ queries. Since, the partial private key $pp_i = s.H_1(ID_i)$ can be computed by both S_2 and $\mathscr{A}_2$. So, there is no need to model the hash function H_1 as random oracle. S_2 maintains a list $(ID_i, x_i, P_i, PS_j, c_i)$ in list L.

H_2 **queries:** S_2 maintains a list $L_{H_2} = (m_k, PS1_j, P_i, U_i, h_{ijk})$ which is empty initially. Suppose $(m_k, PS1_j, P_i, U_i, h_{ijk})$ is submitted to $H_2(.)$. S_2 first checks if the request has been asked before on the same input and if so, the previous value is returned. Otherwise, S_2 selects random $h_{ijk} \in {}_R\mathbb{Z}_q^*$ and adds the tuple $(m_k, PS1_j, P_i, U_i, h_{ijk})$ to L_{H_2} and returns h_{ijk} as answer to the query made by $\mathscr{A}_2$.

H_3 **queries:** S_2 maintains a list $L_{H_3} = (PS1_j, t_{1_j})$ which is empty initially. When $\mathscr{A}_2$ issues a query on $H_3(.)$ oracle, S_2 checks if the request has been asked previously on same input and returns the same answer. Otherwise, S_2 selects a random $t_{1_j} \in {}_R\mathbb{Z}_q^*$ and adds $(PS1_j, t_{1_j})$ to the list L_{H_3} and return t_{1_j} as an answer to $\mathscr{A}_2$.

RevealPseudonym queries: The request is issued on an identity ID_i.

- S_2 recovers the corresponding $(ID_i, x_i, P_i, PS_j, c_i)$ and checks if $PS_j = \perp$. If $PS_j \neq \perp$, S_2 returns PS_j to $\mathscr{A}_2$. Otherwise, S_2 randomly selects $k_j \in {}_R\mathbb{Z}_q^*$ and computes $PS1_j = k_j.Q_i$ where $Q_i = H_1(ID_i)$ and $PS2_j = k_j.t_{1_j}$ where the corresponding list is $L_{H_3} = (PS1_j, t_{1_j})$. S_2 returns $PS_j = (PS1_j + PS2_j)$ to $\mathscr{A}_2$ and adds $(ID_i, x_i, P_i, PS_j, c_i)$ to the list L.
- If the list L does not contain $(ID_i, x_i, P_i, PS_j, c_i)$, then S_2 sets $PS_j = \perp$ and randomly selects $k_j \in {}_R\mathbb{Z}_q^*$, computes $PS1_j = k_j.Q_i$ and $PS2_j = k_j.t_{1_j}$ where the corresponding list is $L_{H_3} = (PS1_j, t_{1_j})$ and $Q_i = H_1(ID_i)$. S_2 returns $PS_j = (PS1_j + PS2_j)$ to $\mathscr{A}_2$ and adds $(ID_i, x_i, P_i, PS_j, c_i)$ to the list L.

RevealPublicKey queries: When the RevealPublicKey query is issued on an identity ID_i, the following three cases hold:

- S_2 checks the list L and if it contains $(ID_i, x_i, P_i, PS_j, c_i)$, S_2 returns P_i to $\mathscr{A}_2$.
- If the list L does not contain $(ID_i, x_i, P_i, PS_j, c_i)$, then a coin $c_i \in \{0,1\}$ is flipped by S_2 yielding 0 with probability ζ and yielding 1 with probability $(1-\zeta)$. If $c_i = 0$, S_2 randomly selects $\gamma_i \in {}_R\mathbb{Z}_q^*$ and sets $P_i = \gamma_i.P \in \mathbb{G}_1$. S_2 sets $x_i = \gamma_i$ and inserts a tuple $(ID_i, x_i, P_i, PS_j, c_i)$ to the list L and P_i is returned to $\mathscr{A}_2$.
- If $c_i = 1$, S_2 returns $P_i = \gamma_i.Y \in \mathbb{G}_1$ to $\mathscr{A}_2$ where a random $\gamma_i \in_R \mathbb{Z}_q^*$ and sets $x_i = \gamma_i$ and inserts a tuple $(ID_i, x_i, P_i, PS_j, c_i)$ to the list L.

RevealSecretKey queries: The request is issued on an identity ID_i, the following three cases hold:

- It checks if the list L contains $(ID_i, x_i, P_i, PS_j, c_i)$, then it returns x_i to $\mathscr{A}_2$.

- If the list does not contain $(ID_i, x_i, P_i, PS_j, c_i)$, then if $c_i = 1$, it halts.
- If $c_i = 0$, S_2 makes *RevealPublicKey* query and the tuple $(ID_i, x_i, P_i, PS_j, c_i)$ is added to the list L and returns x_i to $\mathscr{A}_2$.

Sign Oracle: On receiving a sign query on ID_i, S_2 firstly recovers the list $L = (ID_i, x_i, P_i, PS_j, c_i)$, then $L_{H_2} = (m_k, PS1_j, P_i, U_i, h_{ijk})$ and $L_{H_3} = (PS1_j, t1_j)$ and generates the signature as follows:

- If $c_i = 1$, then if the list contains $(ID_i, x_i, P_i, PS_j, c_i)$, S_2 checks as $x_i = \gamma_i$ and $P_i = \gamma_i.Y \in \mathbb{G}_1$.
- If $c_i = 0$, then list contains $(ID_i, x_i, P_i, PS_j, c_i)$, S_2 checks as $x_i = \gamma_i$ and $P_i = \gamma_i.P \in \mathbb{G}_1$.

Now, for generating the signature, if $c_i = 1$, then S_2 checks if the adversary $\mathscr{A}_2$ has not made the *sign* query on $(ID_i, x_i, P_i, PS_j, c_i)$. In addition, the forged signature must satisfy

$$\begin{aligned} e(V,P) &= e(PS1_j.t1_j + h_{ijk}.U_i, P_{pub})e(h_{ijk}.P_i, P_{rsu}) \\ &= e(k_j.Q_i.t1_j + h_{ijk}.U_i, P_{pub})e(h_{ijk}.P_i, P_{rsu}) \end{aligned}$$

The above equation holds for a valid signature. Otherwise, S_2 aborts. If the equation holds i.e. if S_2 does not abort, then the signature on $(ID_i^*, x_i^*, P_i^*, PS_j^*, c_i)$ is

$$e(V^*,P) = e(PS1_j^* \cdot t1_j^* + h_{ijk}^* \cdot U_i^*, P_{pub})e(h_{ijk}^* \cdot P_i^*, P_{rsu})$$

By setting the values as $PS1_j^* = k_j \cdot Q_i^*$, $Q_i^* = H_1(ID_i^*)$, $P_{pub} = \lambda \cdot P$, $P_{rsu} = a \cdot P$, $P_i = \gamma_i^* \cdot Y = \gamma_i^* \cdot bP$, and $t1_j^* = H_3(PS1_j^*)$:

$$\begin{aligned} e(V^*,P) &= e(k_j \cdot Q_i^* \cdot t1_j^* + h_{ijk}^* \cdot U_i^*, \lambda \cdot P)e(h_{ijk}^* \cdot \gamma_i^* \cdot bP, aP) \\ e(h_{ijk}^* \cdot \gamma_i^* \cdot bP, aP) &= e(V^*,P)e(k_j \cdot Q_i^* \cdot t1_j^* + h_{ijk}^* \cdot U_i^*, \lambda \cdot P)^{-1} \\ e(h_{ijk}^* \cdot \gamma_i^* \cdot abP, P) &= e(V^*,P)e((k_j \cdot Q_i^* \cdot t1_j^* + h_{ijk}^* \cdot U_i^*)\lambda, P)^{-1} \\ e(h_{ijk}^* \cdot \gamma_i^* \cdot abP, P) &= e(V^* - \lambda(k_j \cdot Q_i^* \cdot t1_j^* + h_{ijk}^* \cdot U_i^*), P) \end{aligned}$$

Now,

$$\begin{aligned} h_{ijk}^* \cdot \gamma_i^* \cdot abP &= V^* - \lambda(k_j \cdot Q_i^* \cdot t1_j^* + h_{ijk}^* \cdot U_i^*) \\ abP &= (h_{ijk}^* \cdot \gamma_i^*)^{-1}[V^* - \lambda(k_j \cdot Q_i^* \cdot t1_j^* + h_{ijk}^* \cdot U_i^*)] \end{aligned}$$

Thus, S_2 outputs abP as the solution of the Computational Diffie Hellman problem. Similar to Theorem 1, it can be shown that the given instance of CDH problem is solved by S_2 with probability

$$\varepsilon' \geq \frac{1}{(q_p+1).e}\varepsilon$$

□

6 New Certificateless Aggregate Signature Scheme

This section proposes a new certificateless aggregate signature scheme based on the proposed Certificateless signature scheme.

6.1 Proposed Certificateless Aggregate Signature Scheme

This section explains the construction of a new certificateless aggregate signature scheme for VANETs. All the six algorithms *Setup*, *Registration*, *PartialKeyGen*, *UserKeyGen*, *PseudonymGen* and *Sign* algorithms remain same as proposed in the basic CLS scheme. The remaining two algorithms are described as:

Aggregate

This algorithm aggregates the collection of individual signatures. The vehicle acts as an aggregate signature generator aggregating a collection of n signatures form n users $\{\mathscr{U}_1 \dots \mathscr{U}_n\}$ with pseudonyms $\{\mathscr{PS}_1 \dots \mathscr{PS}_n\}$, public keys $\{\mathscr{P}_1 \dots \mathscr{P}_n\}$ with messages $\{\mathscr{M}_1 \dots \mathscr{M}_n\}$ and signatures $\{\sigma_1 = (U_1, V_1), \dots, \sigma_n = (U_n, V_n)\}$ from corresponding users $\{\mathscr{U}_1 \dots \mathscr{U}_n\}$ respectively. It aggregates all the signatures signed under the same Roadside Unit (RSU). The aggregate signatures can be computed as $V = \sum_{i=1}^{n} V_i$ and $\sigma = (U_1 .... U_n, V)$ is an output for an aggregate signature pair.

AggregateVerify

An aggregate signature $\sigma = (U_1 \dots U_n, V)$ which is signed by n users $\{\mathscr{U}_1 .... \mathscr{U}_n\}$ on n messages $\{\mathscr{M}_1 ..... \mathscr{M}_n\}$ is verified by AggregateVerify algorithm with the help of following steps:

- Compute $h_i = H_2(m_i, PS1_i, P_i, U_i)$ for i = 1 to n.
- Compute $T_i = H_3(PS1_i)$ for i = 1 to n.
- Check the following equation if it holds or not:

$$e(V,P) = e(\sum_{i=1}^{n}[PS1_i.T_i + h_i.U_i], P_{pub}) e(\sum_{i=1}^{n} h_i.P_i, P_{rsu})$$

for all the messages signed under single RSU. The vehicle accepts the signature if the equation holds, otherwise outputs $\perp$.

6.2 Security Proof

Theorem 3 *The certificateless aggregate signature scheme is considered secure against existential forgery if the proposed certificateless signature scheme is secure against adaptive chosen message attacks in the aggregate model.*

Proof: There exists an adversary $\mathscr{A}$ having an advantage ε in forging a signature in certificateless aggregate signature scheme in the random oracle model. An algorithm S is constructed to output the forgery of the certificateless aggregate signature scheme. S chooses random $c \in_R \mathbb{Z}_q^*$ and sets $P_{pub} = X$ and $P_{rsu} = c.P$ and then start performing oracle queries. Algorithm S maintains a list $L = (ID_i, pp_i, x_i, P_i, PS_i)$ while $\mathscr{A}$ makes queries throughout the Game and algorithm S responds to all the queries of $\mathscr{A}$.

H_1 **queries:** On submitting an identity ID_i to oracle H_1, it returns same answer if the request has been asked before. Otherwise, a coin $c_i \in \{0,1\}$ is flipped by S yielding 0 with probability ζ and yielding 1 with probability $(1-\zeta)$ and picks α_i randomly from ${}_R\mathbb{Z}_q^*$. If $c_i = 0, H_1(ID_i)$ is defined as $Q_i = \alpha_i.P_{pub} \in \mathbb{G}_1$. If $c_i = 1$, then the hash value $H_1(ID_i) = \alpha_i.P \in \mathbb{G}_1$. A tuple $(ID_i, \alpha_i, c_i, Q_i)$ is inserted in the list $L_{H_1} = (ID_i, \alpha_i, c_i, Q_i)$ by algorithm S in both cases to keep the track of all the queries. Now, ultimately adversary $\mathscr{A}$ has to output n users set with identities from $L^*_{ID} = \{ID^*_1 \dots ID^*_n\}$, public keys from $L^*_P = \{P^*_1 \dots P^*_n\}$, pseudonyms from $L^*_{PS} = \{PS^*_1 \dots PS^*_n\}$, n messages from the set $L^*_m = \{m^*_1 \dots m^*_n\}$ and aggregate signature $\sigma^* = (U^*_1 \dots U^*_n, V^*)$. Now, S finds the corresponding n tuples $(ID_i, \alpha_i, c_i, Q_i)$ for i = 1 to n from L_{H_1} and precedes only when $c_k = 0$ and $c_j = 1$ for $j = 1$ to n and $j \neq k$. The signature $(m^*_k, PS1^*_k, P^*_k, U^*_k, h^*_k)$ has never been requested before. Otherwise, S will fail and halt. When $Q_k = \alpha_k.P_{pub}$ and $Q_j = \alpha_j.P$ for j = 1 to n and $j \neq k$, S succeeds. The generated aggregate signature is $\sigma^* = (U^*_1 \dots U^*_n, V^*)$ satisfying the aggregate verification equation:

$$e(V^*, P) = e(\sum_{i=1}^{n} [PS1^*_i.T^*_i + h^*_i.U^*_i], P_{pub}) e(\sum_{i=1}^{n} h^*_i.P^*_i, P_{rsu})$$

Then, S searches the tuple $(m^*_i, PS1^*_i, P^*_i, U^*_i, h^*_i)$ from the list L_{H_2}, $(PS1^*_i, t^*_{1_i})$ from the list L_{H_3} and the tuple $(ID^*_i, pp^*_i, x^*_i, P^*_i, PS^*_i)$ from the list L. It sets $V^*_i = \alpha_i.P_{pub}$, then $e(V^*_i, P) = e(Q^*_i, P_{pub})$ for i = 1…n, $j \neq k$. Finally S constructs $V^{'^*}_i$ as $V^* - \sum_{i=1, i\neq k}^{n} V^*_i$. Now,

$$V^{'^*}_i = pp^*_k.PS2^*_k + \sum_{i=1}^{n} h^*_i.r^*_i.P^*_{pub} + \sum_{i=1}^{n} h^*_i.x^*_i.P^*_{rsu}$$

As $U^*_i = r^*_i.P$ and $r^*_i \in {}_R\mathbb{Z}_q^* (1 \leq i \leq n)$. Algorithm S now chooses a random number $w^*_k \in {}_R\mathbb{Z}_q^*$ and computes $U^{'^*} = (w^*_k)^{-1} \sum_{i=1}^{n} h^*_i.U^*_i$. After this, S computes $P^{'^*}_k = (w^*_k)^{-1} \sum_{i=1}^{n} h^*_i.P^*_i$.

S updates P_i to $P^{'}_i$ by invoking *ReplacePublicKey* query. Then, S defines the hash value $H_2(m^*_k, PS1^*_k, P^*_k, U^*_k) = w^*_k$. If the tuple $(m^*_k, PS1^*_k, P^*_k, U^*_k)$ already appears in list L_{H_2} then it tries another w^*_k until there is no collision. Thereupon, $(U^{'^*}, V^{'^*})$ is a valid signature for identity ID^*_k on message m^*_k with pseudonym PS^*_k and corresponding public key P^*_k. Its verification equation is:

$$\begin{aligned}
& e(w^*_k.U^{'^*} + PS1^*_k.t^*_{1_k}, P_{pub}) e(w^*_k.P^{'^*}_k, P_{rsu}) \\
&= e(w^*_k.(w^*_k)^{-1} \sum_{i=1}^{n} h^*_i.U^*_i + PS1^*_k.t^*_{1_k}, P_{pub}) e(w^*_k.(w^*_k)^{-1} \sum_{i=1}^{n} h^*_i.P^*_i, P_{rsu}) \\
&= e(\sum_{i=1}^{n} h^*_i.U^*_i + PS1^*_k.t^*_{1_k}, P_{pub}) e(\sum_{i=1}^{n} h^*_i.P^*_i, P_{rsu}) \\
&= e(\sum_{i=1}^{n} h^*_i.r^*_i.P_{pub} + pp^*_i.PS2^*_k, P) e(\sum_{i=1}^{n} h^*_i.x^*_i.P_{rsu}, P) \\
&= e(V^{'^*}, P)
\end{aligned}$$

Thus, the output $(U^{'^*}, V^{'^*})$ is given by algorithm S which acts as a forgery of the basic certificateless signature scheme.

To complete the proof, it is to be shown that S's advantage in forging the basic certificateless signature is atleast

$$\varepsilon' \geq \frac{1}{(q_k+n).e} \cdot \varepsilon$$

There are three events needed for S to succeed:

E_1**:** The result of any $\mathscr{A}$'s *RevealPartialKey* queries does not abort S.

E_2**:** A valid and non trivial signature is generated by $\mathscr{A}$.

E_3**:** Probability that $\mathscr{A}$ outputs a valid and nontrivial forgery and S does not abort.

Probability that S succeeds after all these events happen is

$$P[E_1 \wedge E_2 \wedge E_3] = P[E_1] \cdot P[E_2|E_1] \cdot P[E_3|E_1 \wedge E_2]$$

- Claim 1: Probability that S does not abort as a result of the *RevealPartialKey* queries is atleast $(1-\zeta)^{q_k}$ Hence, $P[E_1] \geq (1-\zeta)^{q_k}$
  Proof: $P[c_i = 1] = (1-\zeta)$, for key extraction queries, probability that S does not abort is $(1-\zeta)$. As it takes atmost q_k times *RevealPartialKey* queries, probability that S does not abort as result of $\mathscr{A}$ queries is atleast $(1-\zeta)^{q_k}$.
- Claim 2: The probability of S not aborting with $\mathscr{A}$'s signature queries and key extraction queries is ε . $P[E_2|E_1] \geq \varepsilon$.
- Claim 3: Probability that $\mathscr{A}$ outputs a valid and nontrivial forgery and S does not abort is ζ. $P[E_3|E_1 \wedge E_2] \geq \zeta(1-\zeta)^{n-1}$
  Proof: Suppose events E_1 and E_2 have occurred and $\mathscr{A}$ has generated some valid and nontrivial forgery. Hence $P[E_3|E_1 \wedge E_2] \geq \zeta(1-\zeta)^{n-1}$

Thus,

$$\begin{aligned} P[E_1 \wedge E_2 \wedge E_3] &= P[E_1] \cdot P[E_2|E_1] \cdot P[E_3|E_1 \wedge E_2] \\ &= (1-\zeta)^{q_k} \cdot \varepsilon \cdot \zeta(1-\zeta)^{n-1} \\ &= \zeta(1-\zeta)^{q_k+n-1} \cdot \varepsilon \end{aligned}$$

Now, $\zeta_{opt} = \frac{1}{q_k+n}$. Thus,

$$\begin{aligned} \varepsilon' &\geq \zeta(1-\zeta)^{q_k+n-1} \cdot \varepsilon \\ &\geq \frac{1}{q_k+n}\left[1-\frac{1}{q_k+n}\right]^{q_k+n-1} \cdot \varepsilon \end{aligned}$$

With sufficiently large q_k, the term $\left[1-\frac{1}{q_k+n}\right]^{q_k+n-1}$ tends to $\frac{1}{e}$. Thus the probability is,

$$\varepsilon' \geq \frac{1}{(q_k+n)\cdot e} \cdot \varepsilon$$

□

Tab. 1: Comparison of the proposed scheme with other 4 schemes

CLS scheme	Type	Sign Cost	Verify Cost	Aggregate Verify Cost
Zhang and Zhang (2009)	*Sync*#	3S	4P	(n+3)P
Zhang et al. (2010)	Sync	5S	5P+2S	5P+2nS
First scheme in Gong et al. (2007)	Ad hoc*	2S	3P	(2n+1)P
Second scheme in Gong et al. (2007)	Sync	3S	3P	(n+2)P+nS
Our scheme	Ad hoc	3S	3P+3S	3P+3nS

#Sync means normal mode of transfer

*Ad hoc means temporary mode of transfer

7 Efficiency

In practice, element size in group $\mathbb{G}_1$ can be reduced by a factor of 2 using the various compression techniques. The certificateless aggregate signature scheme proposed here is a short CLAS scheme like BLS signature scheme by Boneh et al. (2001). Elliptic curve are used to choose the group and bilinear map resulting in a group size of 160 bits and thus, the signatures generated by this scheme are of length 160 bits approximately (half-size comparing to other proposed CLAS schemes). Therefore, the proposed scheme is much more efficient in terms of bandwidth which is a must requisite for the bandwidth limited networks such as VANETs. The comparison of the computational costs of the proposed scheme and the computational costs of already existing schemes, Zhang and Zhang (2009); Zhang et al. (2010); Gong et al. (2007) is made in Tab. 1. Here, the cost of operations which may be pre-computed by the signer such as $pp_i.PS2_i$ etc. are omitted.

Tab. 1 gives the detailed comparison of the proposed scheme with other schemes based on Type, Sign Cost, Verify Cost and Aggregate Verify Cost. The major goal is to reduce the signature verification cost to enhance the signature verification process. Here, the two main operations are considered on the basis of which comparison is made, scalar multiplication(S) in $\mathbb{G}_1$ and pairing operation(P). The pairing operation is the most costly operation, so there is need to minimize the pairing operations. The scheme presented by Zhang and Zhang (2009) takes (n+3) pairing operations for signature verification process, thus the pairing operations increase linearly as the number of signatures increase. The scheme by Zhang et al. (2010) takes 5 pairing operations and 2n scalar multiplications; therefore it takes 5 constant pairing operations and the scalar multiplications increase linearly. The first scheme in Gong et al. (2007) takes (2n+1) pairing operations and the second scheme in Gong et al. (2007) takes (n+2) pairing operations and n scalar multiplications. Both the schemes are highly costly in Gong et al. (2007) as the pairing operations increase linearly with the number of signatures. The Aggregate Verify procedure of the proposed scheme is much efficient as it takes just 3 pairing operations and 3n scalar multiplications which is less as compared to all other schemes whereas the sign procedure is comparable to other schemes. In vehicular ad-hoc networks, vehicle has high computational power where the vehicle needs to sign only one signature whereas needs to verify multiple signatures. Therefore, the signing cost can be compromised but one cannot compromise on the signature verification cost. It can be seen that the computational cost of the proposed scheme is more efficacious than the already existing schemes.

8 Conclusion

A new efficient Certificateless Aggregate Signature Scheme is proposed for vehicular communications. The proposed signature scheme is proven existentially unforgeable against the chosen message attack under the assumption that CDH problem is intractable in the random oracle model. The proposed CLAS scheme is adduced specifically for securing vehicular communications in vehicular ad-hoc networks by reducing the signature verification time drastically and helps in verifying more messages in the specific stipulated time, thus increasing the efficiency of the network. The propounded scheme has much less computational cost in terms of verifying signatures when compared with the already proposed works. This scheme is very adequate in networks which have limited bandwidth such as vehicular ad-hoc networks.

Acknowledgements

The work is fully supported by the grant under TCS (Tata Consultancy Services) Research Scholar Program.

References

S. S. Al-Riyami and K. G. Paterson. Certificateless public key cryptography. In *Advances in Cryptology-ASIACRYPT 2003*, pages 452–473. Springer, 2003.

J. Baek, R. Steinfeld, and Y. Zheng. Formal proofs for the security of signcryption. *Journal of cryptology*, 20(2):203–235, 2007.

A. Bagherzandi and S. Jarecki. Identity-based aggregate and multi-signature schemes based on rsa. In *Public Key Cryptography–PKC 2010*, pages 480–498. Springer, 2010.

M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM conference on Computer and communications security*, pages 62–73. ACM, 1993.

A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum. Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 276–285. ACM, 2007.

D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. In *Advances in Cryptology-ASIACRYPT*, pages 514–532. Springer, 2001.

D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Advances in cryptology-EUROCRYPT*, pages 416–432. Springer, 2003.

K. Y. Choi, J. H. Park, J. Y. Hwang, and D. H. Lee. Efficient certificateless signature schemes. In *Applied Cryptography and Network Security*, pages 443–458. Springer, 2007.

H. Du and Q. Wen. Efficient and provably-secure certificateless short signature scheme from bilinear pairings. *Computer Standards & Interfaces*, 31(2):390–394, 2009.

C. Gamage, B. Gras, B. Crispo, and A. S. Tanenbaum. An identity-based ring signature scheme with enhanced privacy. In *Securecomm and Workshops, 2006*, pages 1–5. IEEE, 2006.

Z. Gong, Y. Long, X. Hong, and K. Chen. Two certificateless aggregate signatures from bilinear maps. In *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on*, volume 3, pages 188–193. IEEE, 2007.

D. He, J. Chen, and J. Hu. A pairing-free certificateless authenticated key agreement protocol. *International Journal of Communication Systems*, 25(2):221–230, 2012.

B. C. Hu, D. S. Wong, Z. Zhang, and X. Deng. Key replacement attack against a generic construction of certificateless signature. In *Information Security and Privacy*, pages 235–246. Springer, 2006.

B. C. Hu, D. S. Wong, Z. Zhang, and X. Deng. Certificateless signature: a new security model and an improved generic construction. *Designs, Codes and Cryptography*, 42(2):109–126, 2007.

X. Huang, W. Susilo, Y. Mu, and F. Zhang. On the security of certificateless signature schemes from asiacrypt 2003. In *Cryptology and Network Security*, pages 13–25. Springer, 2005.

X. Huang, Y. Mu, W. Susilo, D. S. Wong, and W. Wu. Certificateless signature revisited. In *Information Security and Privacy*, pages 308–322. Springer, 2007.

P. Kamat, A. Baliga, and W. Trappe. An identity-based security framework for vanets. In *Proceedings of the 3rd international workshop on Vehicular ad hoc networks*, pages 94–95. ACM, 2006.

P. Kamat, A. Baliga, and W. Trappe. Secure, pseudonymous, and auditable communication in vehicular ad hoc networks. *Security and Communication Networks*, 1(3):233–244, 2008.

X. Lin, X. Sun, P.-H. Ho, and X. Shen. Gsis: a secure and privacy-preserving protocol for vehicular communications. *Vehicular Technology, IEEE Transactions on*, 56(6):3442–3456, 2007.

J. K. Liu, M. H. Au, and W. Susilo. Self-generated-certificate public key cryptography and certificateless signature/encryption scheme in the standard model. In *Proceedings of the 2nd ACM symposium on Information, computer and communications security*, pages 273–283. ACM, 2007.

R. Lu, X. Lin, H. Zhu, P.-H. Ho, and X. Shen. Ecpp: Efficient conditional privacy preservation protocol for secure vehicular communications. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*. IEEE, 2008.

A. Lysyanskaya, S. Micali, L. Reyzin, and H. Shacham. Sequential aggregate signatures from trapdoor permutations. In *Advances in Cryptology-Eurocrypt 2004*, pages 74–90. Springer, 2004.

S. Miao, F. Zhang, S. Li, and Y. Mu. On security of a certificateless signcryption scheme. *Information Sciences*, 232:475–481, 2013.

A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in cryptology*, pages 47–53. Springer, 1985.

A. Studer, E. Shi, F. Bai, and A. Perrig. Tacking together efficient authentication, revocation, and privacy in vanets. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2009. SECON'09. 6th Annual IEEE Communications Society Conference on*, pages 1–9. IEEE, 2009.

J. Sun, C. Zhang, and Y. Fang. An id-based framework achieving privacy and non-repudiation in vehicular ad hoc networks. In *Military Communications Conference, 2007. MILCOM 2007. IEEE*, pages 1–7. IEEE, 2007.

H. Xiong, F. Li, and Z. Qin. Certificateless threshold signature secure in the standard model. *Information Sciences*, 237:73–81, 2013.

D. H. Yum and P. J. Lee. Generic construction of certificateless signature. In *Information Security and Privacy*, pages 200–211. Springer, 2004.

L. Zhang and F. Zhang. A new certificateless aggregate signature scheme. *Computer Communications*, 32(6):1079–1085, 2009.

L. Zhang, B. Qin, Q. Wu, and F. Zhang. Efficient many-to-one authentication with certificateless aggregate signatures. *Computer Networks*, 54(14):2482–2491, 2010.