Privacy-Enhancing Signcryption Scheme

Sara Ricci, Petr Dzurenda, Jan Hajny and Lukas Malina

Brno University of Technology
Technicka 12, Brno
Czech Republic
Email: {ricci, dzurenda, hajny, malina}@feec.vutbr.cz

September 30, 2019

Abstract

Many signcryption schemes have been proposed for different privacy-enhancing purposes, but currently none of them guarantees sender whole anoninymity. In this paper, we propose a new privacy-enhancing signcryption scheme which provides as main privacy-enhancing features: ciphertext and sender anonymity, traceability and unlinkability. To the best of our knowledge, this is the only signcryption scheme where sender identity is hidden not only to an outsider but also to the receiver, and sender and receiver identity is hidden to an outsider.

Moreover, security analysis of the scheme is also provided. Our proposal is proven to be strongly existentially unforgeable under an adaptive chosen message attack, indistinguishable under adaptive chosen ciphertext attack, and to provide ciphertext anonymity under adaptive chosen ciphertext attack.

The experimental results show that our scheme is efficient even on computationally restricted devices and can be therefore used in many IoT applications. `Signcrypt` protocol on smart card takes less then 1 s (including communication overhead). Time of `Unsigncrypt` protocol on current ARM devices is negligible (less than 40 ms).

Keywords— signcryption protocol, anonymity, privacy-enhancing technology, group signature, embedded devices, smart cards.

1 Introduction

A signcryption scheme [38] combines a digital signature with a public-key encryption scheme. Most of the traditional signcryption protocols are based on Diffie-Hellman problem. These schemes guarantee data confidentiality and integrity, as well as signature unforgeability. For instance, Mohanty et al. [27] present a signcryption protocol based on Diffie-Hellman problem. This scheme is one of the first scheme that try to deal with user's anonymity. In particular, the identity of the sender is hidden to the receiver and an outsider. In order to achieve this kind-of-anonymity, the scheme requires a really active group manager who is involved in the signcryption phase. This manager's involvement can lead to privacy leakage and slow down the computation, and therefore, it is normally avoided.

In signcryption protocols, users' privacy is basically achieved by *ciphertext anonymity* which means that the ciphertext reveals no information about who created it nor about whom it is intended to [38]. In other words, the problem is to hide sender's and receiver's identity to an outsider. The use of bilinear pairing in a signcryption protocol allows to achieve ciphertext anonymity property at the expense of speed, e.g. see [9, 28, 32]. However, many schemes require an even stronger anonymity. For instance, in case of e-voting the voter's (sender's) identity has to be hidden also to the receiver as well as in the case of video streaming applications where anonymous users (senders) broadcast live video to the Internet.

Observe that if the data are encrypted using standard data encryption (e.g. plain AES), *sender's anonymity*, i.e. sender's identity is hidden not only to outsiders but also to receivers, is completely guaranteed. However, this does not allow to provide data authenticity. In other words, we should be able to identify malicious users, e.g. users which broadcast a video with prohibited content, while keeping honest-user identity hidden. Group signature allows to provide data authenticity without disclosing users' identity. In particular, a user can anonymously sign a message on behalf of the group. Our scheme uses group signature and bilinear maps in order to provide ciphertext anonymity plus sender anonymity.

Our proposal has a wide range of possible usage. For instance, it can be applied in smart cities which try to offer to their citiziens always better services. In particular, smart cities can refer to: (1) smart mobility (vehicles sharing, traffic and road conditioning monitoring); (2) smart grid (energy management, smart metering); (3) public safety (territorial monitoring, smart air quality); and (4) smart healthcare (electronic health records, remote patients monitoring). In order to provide all these services, smart cities depend on new technologies as global infrastructure network, sensors and wireless devices. These interconnected objects are exchanging in real time data and information which may be sensitive.

Accordingly, this increasing connectivity can expose devices and, therefore, citiziens to cybersecurity attacks, i.e. attackers may penetrate on computer networks, disclose and alter the information they contain. For instance, Popescul and Radu [31] shows that attackers can have access to on-board computers of cars. Coventry and Branley [10] identify healthcare as one of the most targeted sectors of cybersecurity attacks. These breaches are not only about stealing medical health records or denying access to health services, but even remotely manipulating devices to alter operation or send fatal drug doses. In this scenario, the safety, privacy and trust of citiziens must be maintained by enforcing security and anonymization in the used technologies.

2 State of the art

There is a sizeable body of literature devoted to signcryption protocols as depicted in Table 1. Observe that in the table two anonymity types are considered: ciphertext anonymity, i.e. sender's identity is hidden to an outsider as well as receiver's identity to an outsider, and, sender anonymity, i.e. sender's identity is hidden not only to an outsider but also to the receiver. The level of privacy achieved by the below schemes depends on how many anonymity types they cover. Another important remark is that the property "receiver's identity is hidden to the sender" is not contemplated in the previous list, because it is normally supposed that the sender knows the identity of the receveir of its message.

Most of signcryption protocols propose a bilinear pairing strategy in order to reach stronger anonymity property. In fact, the use of bilinear pairing in a signcryption protocol allows to achieve ciphertext anonymity property at the expense of speed. Libert and Quisquater [22, 38] propose a scheme based on pairing which is only partially anonymous. In fact, an outsider cannot identify who was the sender but knows who is the receiver and the receiver needs sender's public-key to unsigncrypt the message. Chaudhari and Das [9] introduce a pairing-based scheme where the sender *and the receivers* identities are protected against an outsider, i.e. the scheme guarantees ciphertext anonymity. This proposal can be suitable for a multi-receiver environment, where only authorized receivers can decrypt the ciphertext and verify the signature. The pay-TV system is an example where multi-receiver signcryption scheme can be suitable. Here, only authorized receivers can decrypt the TV program (ciphertext) broadcasted by a service provider (sender) and learn who was the sender if necessary.

Scheme	Problem	Sender anon.	Ciphertext anon.		Multi-rec.	# pairing
		S-to-R	S-to-O	R-to-O		
Mohanty [27]	DH	✓	✓	✗	✗	-
Libert [22]	pairing	✗	✓	✗	✗	$2R$
Chaudhari [9]	pairing	✗	✓	✓	✓	$3S + 7R$
Pang [29]	interp-pairing	✓	✓	✗	✓	$4R$
Pang [28]	ring-pairing	✗	✓	✓	✓	$nS + 3R$
Huang [16]	ring-pairing	✓	✓	✗	✗	$(n + 1)S + 3R$
Saraswat [32]	ring-pairing	✗	✓	✓	✓	$1S + 3R$
Li [19]	pairing	✗	✓	✓	✗	$2R$
Braeken [6]	EC	✗	✓	✓	✗	-
our proposal	group-pairing	✓	✓	✓	✓	$2R$

Table 1: Main features of related work on anonymous signcryption schemes: scheme, security assumption, anonymity property, multi-receiver scenario support and number of bilinear pairings used in the corresponding scheme. "DH" stands for "Diffie-Hellman problem", "pairing" for "bilinear Diffie-Hellman problem", "interp-pairing" for "bilinear Diffie-Hellman problem combined with Lagrange interpolation polynomial method", "ring-pairing" for "bilinear Diffie-Hellman problem combined with ring signature", "EC" for "Elliptic curve dicrete logarithm problem", "group-pairing" for "bilinear Diffie-Hellman problem combined with group signature", "S-to-R" for "sender's identity is hidden to the receiver", similarly for "S-to-O" and "R-to-O" where "O" stands for "outsider". The number of pairings is given depending on who is computing it. For instance, "3S+7R" stands for "3 pairings computed by the sender and 7 by the receiver"; "n" is the number of users in the ring.

Most of the multi-receiver signcryption schemes generate different encryptions of the same message, in particular, one ciphertext for each authorized receiver. These ciphertexts are then concatenated in one, which is broadcasted. Therefore, if some part of the ciphertext goes wrong during transmission, only some of the authorized receivers can decrypt the message correctly while the rest cannot. This leads to the unfair decryption problem [29]. Pang et al. [28] present a pairing-based scheme where each receiver needs the whole ciphertext for decryption. However, the identity of the sender is disclosed by any authorized receiver after decryption.

In order to hide receivers' identity to an outsider, the Lagrange interpolation polynomial was also considered [18]. The unique ciphertext can be decrypted by any authorized receiver who owns a root of the interpolation polynomial. Later, this method was used in several anonymous multi-receiver signcryption schemes [21, 29, 36]. Unluckily, Li and Pang [20] pointed out that any scheme based on Lagrange interpolation polynomial methodology cannot achieve the receiver's anonymity and, accordingly, ciphertext anonymity. In fact, every authorized receiver can determine whether the other is one of the authorized receivers. To our knowledge, no current signcryption scheme could combine ciphertext anonymity and fair decryption.

Huang et al. [16] propose to combine pairing-based signcryption scheme with ring signature. In this case, a sender can anonymously signcrypt a message on behalf of the group. However, the receiver identity is not hidden to an outsider. Saraswat et al. [32] also present an anonymous proxy signcryption scheme based on pairing and ring signature. This scheme works in a different scenario, it is only required ciphertext anonymity. Li et al. [19] also propose a scheme where sender and receiver's identity are hidden to an outsider. Their scheme is designed to be efficient on the sender side and suitable for wireless body area networks.

Braeken and Touhafi [6] propose a fast signcryption scheme based on elliptic curve discrete logarithm problem. As in any non-pairing schemes, the anonymity is only partially achieved, i.e. the sender's identity is known by the receiver.

2.1 Contribution and paper structure

In [15], we proposed a novel group signature scheme based on the weak Boneh-Boyen signature [5] and the efficient proofs of knowledge [8]. This scheme has fast signature generation and provides all the main privacy-enhancing signature features, i.e. anonymity, unlinkability and traceability. The present article is an extension of this work, where the proposed signature is included in our signcryption scheme. Accordingly, the new signcryption scheme holds all the properties of the aforementioned group signature scheme.

Our novel signcryption scheme guarantees ciphertext and sender anonymity. As shown in Table 1, our scheme is the only one among the existing signcryption schemes which provides this combination. This is achieved by combining the Elliptic Curve Integrated Encryption Scheme (ECIES) [13] with our group signature [15].
The main properties of the scheme are summarized below.

Privacy-enhancing features:

- *ciphertext anonymity*, i.e. sender and receiver identity is hidden to an outsider;
- *sender anonymity*, i.e. the sender's identity is hidden not only to an outsider but also to the receiver. In this way, instead of sender authentication, group authentication is provided to achieve message integrity and verification of sender;
- *traceability*, i.e. the manager is able to trace which user issued the signature. This leads in the possibility to revoke malicious users;
- *unlinkability*, i.e. two or more signatures cannot be addressed to the same or different senders;

Other features:

- the `Signcrypt` algorithm is *fast*: it requires no bilinear pairing and only 6 exponentiations;
- the `Unsigncrypt` algorithm is efficient: it requires only 2 pairings and $\mathcal{O}(|rl|)$ exponentiations, where $|rl|$ is the length of the revocation list;
- the scheme is built by using primitives with formal security proofs;
- it is possible to revoke malicious users;
- the scheme can be adapted to a multi-receiver scenario;
- security analyses of the scheme are provided.

The rest of this article is organized as follows. Section 3 discusses some preliminaries. Section 4 shows the basic structure of the proposed scheme. Section 5 presents the proposed scheme. Section 6 shows how the scheme can be adapt to a multi-receiver scenario. Section 7 provides the security analysis of the scheme. Section 8 discusses a possible use case for our proposal. Section 9 reports the experimental results. The final section contains the conclusions.

3 Preliminaries

¿From now on, the symbol ":" means "such that" and "$|x|$" is the bitlength of x. We write $a \xleftarrow{\$} A$ when a is sampled uniformly at random from A. A secure hash function is denoted as $\mathcal{H}$. We describe the proof of knowledge protocols (PK) using the notation introduced by Camenisch and Stadler [7]. The protocol for proving the knowledge of discrete logarithm of c with respect to g is denoted as $\mathrm{PK}\{\alpha : c = g^{\alpha}\}$.

3.1 Bilinear Pairing

Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be groups of prime order q. A bilinear map $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ must satisfy:

- bilinearity: $\mathbf{e}(g^x, g_2^y) = \mathbf{e}(g, g_2)^{xy}$ for all $x, y \in \mathbb{Z}_q$;
- non-degeneracy: for all generators $g \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$, $\mathbf{e}(g, g_2)$ generates $\mathbb{G}_T$;

- computability: there exists an efficient algorithm $\mathcal{G}(1^k)$ to compute $\mathbf{e}(g, g_2)$ for all $g \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_2$.

By definition $(q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathbf{e}, g, g_2)$ is a bilinear group if it satisfies all above properties.

3.2 Weak Boneh-Boyen signature

The weak Boneh-Boyen (wBB) signature scheme is a pairing-based short signature scheme. This signature was proven existentially unforgeable against a weak (non-adaptive) chosen message attack under the Strong Diffie-Hellman assumption [5]. The scheme can be used to efficiently sign messages and can be also integrated with the zero-knowledge proofs [8]. In this way, the knowledge of signed messages can be proven anonymously, and unlinkably.

The wBB signature is briefly depicted below.

- $(pk_s, sk, par) \leftarrow$ `KeyGen`$(1^{\mathcal{K}})$: on the input of the system security parameter $\mathcal{K}$, the algorithm generates a bilinear group $par = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathbf{e}, g, g_2)$, computes $pk_s = g_2^{sk}$ where $sk \xleftarrow{\$} \mathbb{Z}_q$, and outputs sk as private key and (pk_s, par) as public key.

- $(\sigma) \leftarrow$ `Sign`(m, par, sk): on the input of the message $m \in \mathbb{Z}_q$, the system security parameters par and the secret key sk, the algorithm outputs the signature of the message $\sigma = g^{\frac{1}{sk+m}}$.

- $(1/0) \leftarrow$ `Verify`(σ, m, pk_s, par): on the input of the system security parameters par, the public key pk, a signature σ and a message m, the algorithm returns 1 if and only if $\mathbf{e}(\sigma, pk_s) \cdot \mathbf{e}(\sigma^m, g_2) = \mathbf{e}(g, g_2)$ holds, i.e. the signature is valid, and 0 otherwise.

3.3 Hard problems

In this section, we describe some security assumptions used in the proposed scheme. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be groups of prime order q, g be a generator of $\mathbb{G}_1$, and g_2 be a generator of $\mathbb{G}_2$. In the first assumption, $\mathbb{G}_1$ is taken equal to $\mathbb{G}_2$ and, therefore, $g = g_2$.

Decisional Diffie-Hellman (DDH) Problem: Given $\langle g, g^a, g^b, g^c \rangle$ for some $a, b, c \in \mathbb{Z}_q^*$, determine whether $c \equiv ab \mod q$.

Definition 3.1 (DDH Assumption). *Let B be an algorithm with output in $\{0,1\}$, which has advantage $Adv_B^{DDH} = \mathbf{Pr}[B(g, g^a, g^b, g^c, ab) = 1] -$*

$\mathbf{Pr}[B(g, g^a, g^b, g^c, c) = 1]$ *in solving the DDH problem. If for any t-time algorithm the advantage Adv_B^{DDH} is negligible ($\leq \epsilon$), we say that the (q, t, ϵ)-DDH assumption holds.*

Strong Diffie-Hellman (p-SDH) Problem: Given as input a $(p + 3)$-tuple of elements

$$(g, g^x, g^{x^2}, \ldots, g^{x^p}, g_2, g_2^x) \in \mathbb{G}_1^{p+1} \times \mathbb{G}_2^2,$$

compute a pair $(c, g^{1/(x+c)}) \in \mathbb{Z}_q \times \mathbb{G}_1$ for some value $c \in \mathbb{Z}_q \setminus \{x\}$.

Definition 3.2 (p-SDH Assumption)**.** *Let B be an algorithm with advantage $Adv_B^{p\text{-}SDH} = \mathbf{Pr}[B(g, g^x, g^{x^2}, \ldots, g^{x^p}, g_2, g_2^x) = (c, g^{1/(x+c)})]$ in solving the p-SDH problem. If for any t-time algorithm the advantage $Adv_B^{p\text{-}SDH}$ is negligible ($\leq \epsilon$), we say that the (p, t, ϵ)-SDH assumption holds.*

3.4 Security models

It is generally accepted that the "correct" notion of security for a signcryption protocol combines unforgeability of the signature and indistinguishability of the encryption scheme [22, 28, 32, 38]. The notion of ciphertext anonymity [38] is also considered since it is an important privacy-enhancing property characterizing our proposal.

We briefly refresh the structure of a signcryption protocol. A traditional signcryption protocol consists of at least four basic algorithms: `Setup`, `KeyGen`, `Signcrypt`, `Unsigncrypt`. In particular, for a fixed security parameter, these algorithms work as follows:

$(pk_s, par) \leftarrow$ `Setup`$(1^{\mathcal{K}})$: on the input of the security parameter $\mathcal{K}$, the algorithm outputs the public system security parameters par and the group public key pk_s.

$(sk_s, pk_r, sk_r) \leftarrow$ `KeyGen`(par): on the input of par, generates sender's secret key sk_s and receiver's key pair (pk_r, sk_r).

$(c, \sigma) \leftarrow$ `Signcrypt`(par, sk_s, pk_r, m): on the input of par, sk_s and pk_r and a message m, outputs a ciphertext c and a signature σ.

$(1/0, m) \leftarrow$ `Unsigncrypt`$(par, c, \sigma, pk_s, sk_r)$: on the input of par, c, σ, pk_s and sk_r, verifies the signature σ and decrypts the ciphertext c. It returns 1 and m iff the signature is valid and 0 otherwise.

3.4.1 Strong Existential Unforgeability (sUF)

We consider the notion of strong existential unforgeability under adaptive chosen message attack (sUF-CMA) [5, 38], which is defined by using the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

> **Setup**: $\mathcal{C}$ runs algorithms `Setup` and `KeyGen` to generate the public system security parameters par, sender's key pair (pk_s, sk_s) and receiver's key pair (pk_r, sk_r). $\mathcal{A}$ is given par, pk_r and pk_s.
>
> **Signcryption-Queries**: $\mathcal{A}$ requests signcryption of at most q_s messages of its choice $m_1, \ldots, m_{q_s} \in \{0,1\}^*$. $\mathcal{C}$ responds to each query with a ciphertext and a signature $(c_i, \sigma_i) \leftarrow \texttt{Signcrypt}(par, sk_s, pk_r, m_i)$.
>
> **Unsigncryption-Queries**: Proceeding adaptively, $\mathcal{A}$ requests unsigncryption of at most q_s ciphertexts $c_1, \ldots, c_{q_s}$, under pk_s and pk_r. Then, $\mathcal{C}$ responds to each query with 1 and a signed message $(1, m_i) \leftarrow$ `Unsigncrypt` $(par, c_i, \sigma_i, pk_s, sk_r)$ if the obtained signed plaintext is valid and with 0 otherwise.
>
> **Output**: $\mathcal{A}$ eventually outputs a pair (m, σ) and wins the game if:
>
> 1. $(1, m) \leftarrow$ `Unsigncrypt` $(par, c, \sigma, pk_s, sk_r)$ is a valid signature.
> 2. c was not the output of a signcryption query `Signcrypt` (par, sk_s, pk_r, m_i) during the game.

We define $Adv_{\mathcal{A}}^{sUF}$ to be the probability that the adversary $\mathcal{A}$ wins in the above game, taken over the coin tosses made by $\mathcal{A}$ and $\mathcal{C}$.

Definition 3.3. *A forger $\mathcal{A}$ is said to (t, q_s, ϵ)-break a signcryption scheme if $\mathcal{A}$ runs in time at most t, $\mathcal{A}$ makes at most q_s signcryption queries and q_s unsigncryption queries, and $Adv_{\mathcal{A}}^{sUF}$ is at least ϵ. A signcryption scheme is (t, q_s, ϵ)-secure against strongly existentially unforgeable under adaptive chosen message attack if there exists no forger that (t, q_s, ϵ)-breaks it.*

3.4.2 Indistinguishability (IND)

We consider the notion of indistinguishability under adaptive chosen ciphertext attack (IND-CCA2) [34, 38], which is defined by using the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

> **Setup**: $\mathcal{C}$ runs algorithms `Setup` and `KeyGen` to generates the public system security parameters par, sender's key pair (pk_s, sk_s) and receiver's key pair (pk_r, sk_r). $\mathcal{A}$ is given par, pk_r and pk_s.

Queries-1: $\mathcal{A}$ requests signcryption of at most $q_{s'}$ messages of its choice $m_1, \ldots, m_{q_{s'}} \in \{0,1\}^*$. $\mathcal{C}$ responds to each query with a ciphertext and a signature $(c_i, \sigma_i) \leftarrow \texttt{Signcrypt}(par, sk_s, pk_r, m_i)$. Then, proceeding adaptively, $\mathcal{A}$ requests unsigncryption of at most $q_{s'}$ ciphertexts $c_1, \ldots, c_{q_{s'}}$, under pk_s and pk_r. $\mathcal{C}$ responds to each query with 1 and a signed message $(1, m_i) \leftarrow \texttt{Unsigncrypt}\ (par, pk_s, pk_r, c_i, \sigma_i)$ if the obtained signed plaintext is valid and with 0 otherwise.

Challenge: $\mathcal{A}$ outputs two equal-length messages m'_0 and $m'_1 \in \{0,1\}^*$ on which it wishes to be challenged. Then, hidden from $\mathcal{A}$ view, $\mathcal{C}$ chooses $b \leftarrow \{0,1\}$ and computes the challenge ciphertext $(c'_*, \sigma_*) \leftarrow \texttt{Signcrypt}(par, sk_s, pk_r, m'_b)$

Queries-2: $\mathcal{A}$ may request at most $q_{s''}$ signcryption and unsigncryption quieries as in Queries-1 phase but with the restriction that $\mathcal{A}$ can not query for c'_*.

Guess: $\mathcal{A}$ produces its guess b' of b. $\mathcal{A}$ is successful if $b' = b$, i.e. the guess is correct.

We define $Adv_{\mathcal{A}}^{IND}$ to be the probability that the adversary $\mathcal{A}$ wins in the above game, and it is defined as

$$Adv_{\mathcal{A}}^{IND} = |2\mathbf{Pr}[b' = b] - 1|$$

Definition 3.4. *An adversary $\mathcal{A}$ is said to $(t, q_s, \mu, m, \epsilon)$-break a signcryption scheme if $\mathcal{A}$ runs in time at most t, $\mathcal{A}$ makes at most $q_s = q_{s'} + q_{s''}$ signcryption queries and q_s unsigncryption queries, the size of the decryption queries is at most μ bits, the size of the challenge messages m'_0 and m'_1 is at most m bits, and $Adv_{\mathcal{A}}^{IND}$ is at least ϵ. A signcryption scheme is $(t, q_s, \mu, m, \epsilon)$-secure against indistinguishability under adaptive chosen ciphertext attack if there exists no adversary that $(t, q_s, \mu, m, \epsilon)$-breaks it.*

3.4.3 Ciphertext anonymity (ANON)

We consider the notion of ciphertext anonymity under adaptive chosen ciphertext attack (ANON-CCA) [38]. This property is satisfied if ciphertexts reveal no information about who created them nor about whom they are intended to. ANON-CCA is defined by using the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$:

Setup: $\mathcal{C}$ runs algorithms `Setup` and `KeyGen` to generates the public system security parameters par, sender's key pair (pk_s, sk_s), and two distinct receiver's key pair (pk_{r_0}, sk_{r_0}) and (pk_{r_1}, sk_{r_1}). $\mathcal{A}$ is given par, pk_{r_0} and pk_{r_1}.

Queries-1: $\mathcal{A}$ requests signcryption of at most $q_{s'}$ messages of its choice $m_1, \ldots, m_{q_{s'}} \in \{0,1\}^*$ for the key pairs (pk_{r_0}, sk_{r_0}) and (pk_{r_1}, sk_{r_1}). $\mathcal{C}$ responds to each query with a ciphertext and a signature $(c_{i_j}, \sigma_i) \leftarrow$ `Signcrypt`$(par, sk_s, pk_{r_j}, m_i)$, where $j = 0, 1$. Then, proceeding adaptively, $\mathcal{A}$ requests unsigncryption of at most $q_{s'}$ ciphertexts $c_1, \ldots, c_{q_{s'}}$, under pk_s and pk_{r_j} with $j = 0, 1$. $\mathcal{C}$ responds to each query with 1 and a signed message $(1, m_i) \leftarrow$ `Unsigncrypt` $(par, pk_s, pk_{r_j}, c_{i_j}, \sigma_i)$ if the obtained signed plaintext is valid and with 0 otherwise.

Challenge: $\mathcal{A}$ eventually outputs two sender's private keys sk_{s_0} and sk_{s_1}, and a a message $m \in \{0,1\}^*$ on which it wishes to be challenged. Then, hidden from $\mathcal{A}$ view, $\mathcal{C}$ chooses $b, d \leftarrow \{0,1\}$ and computes the challenge ciphertext $(c'_*, \sigma_*) \leftarrow$ `Signcrypt`$(par, sk_{s_b}, pk_{r_d}, m)$.

Queries-2: $\mathcal{A}$ may request at most $q_{s''}$ signcryption and unsigncryption quieries as in Queries-1 phase but with the restriction that $\mathcal{A}$ can not query for (c'_*, pk_{s_j}), where $j = 0, 1$.

Guess: $\mathcal{A}$ produces its guess b' of b and d' of d. $\mathcal{A}$ is successful if $b' = b$ and $d = d'$, i.e. the guess is correct.

We define $Adv_{\mathcal{A}}^{ANON}$ to be the probability that the adversary $\mathcal{A}$ wins in the above game, and it is defined as

$$Adv_{\mathcal{A}}^{IND} = |4\mathbf{Pr}[(b', d') = (b, d)] - 1|$$

Definition 3.5. *An adversary $\mathcal{A}$ is said to $(t, q_s, \mu, m, \epsilon)$-break a signcryption scheme if $\mathcal{A}$ runs in time at most t, $\mathcal{A}$ makes at most $q_s = q_{s'} + q_{s''}$ signcryption queries and q_s unsigncryption queries, the size of the decryption queries is at most μ bits, the size of the challenge messages m'_0 and m'_1 is at most m bits, and $Adv_{\mathcal{A}}^{ANON}$ is at least ϵ. A signcryption scheme is $(t, q_s, \mu, m, \epsilon)$-secure against anonymity under adaptive chosen ciphertext attack if there exists no adversary that $(t, q_s, \mu, m, \epsilon)$-breaks it.*

4 Architecture

Three types of entities interact in our signcryption scheme: a manager, a sender and a receiver.

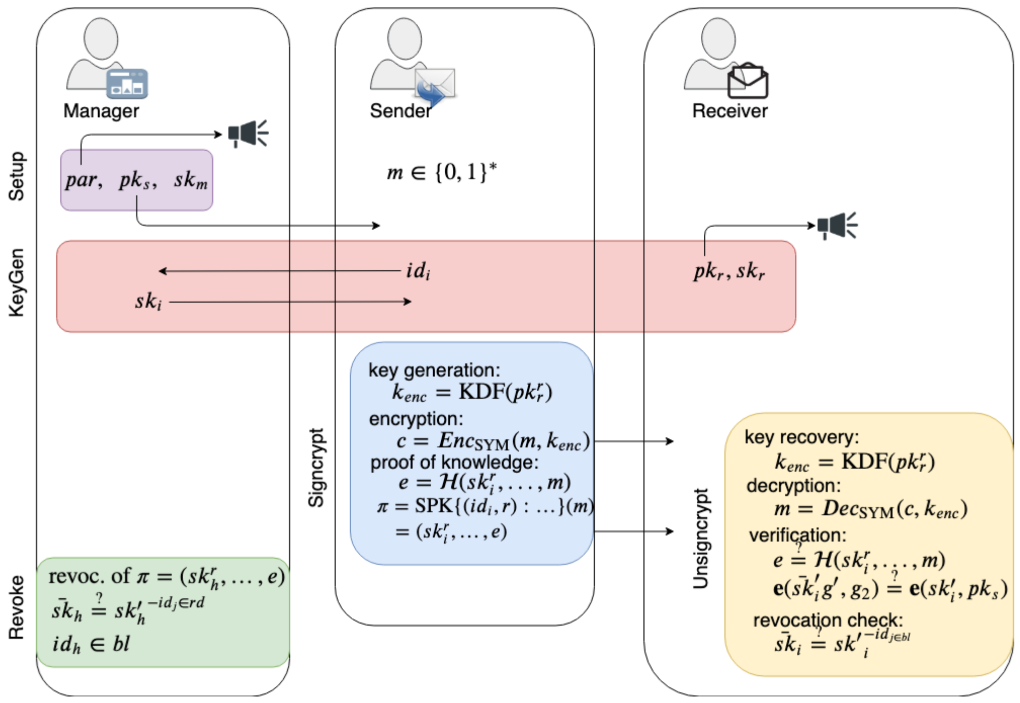

Manager
Sender
Receiver
Setup
KeyGen
Revoke
Signcrypt
Unsigncrypt
$par,\ pk_s,\ sk_m$
$m \in \{0,1\}^*$
id_i
sk_i
pk_r, sk_r
key generation:
$k_{enc} = \mathrm{KDF}(pk_r^r)$
encryption:
$c = Enc_{\mathrm{SYM}}(m, k_{enc})$
proof of knowledge:
$e = \mathcal{H}(sk_i^r, \ldots, m)$
$\pi = \mathrm{SPK}\{(id_i, r) : \ldots\}(m)$
$= (sk_i^r, \ldots, e)$
key recovery:
$k_{enc} = \mathrm{KDF}(pk_r^r)$
decryption:
$m = Dec_{\mathrm{SYM}}(c, k_{enc})$
verification:
$e \overset{?}{=} \mathcal{H}(sk_i^r, \ldots, m)$
$\mathbf{e}(\bar{sk}_i' g', g_2) \overset{?}{=} \mathbf{e}(sk_i', pk_s)$
revocation check:
$\bar{sk}_i \overset{?}{=} {sk'}_i^{-id_{j\in bl}}$
revoc. of $\pi = (sk_h^r, \ldots, e)$
$\bar{sk}_h \overset{?}{=} {sk'}_h^{-id_j \in rd}$
$id_h \in bl$

Figure 1: Sketch of our proposal.

- **Manager (or Public Key Generator)**: the manager generates system security parameters and cryptographic keys, enrolls new senders and revokes invalid ones.
- **Sender**: the sender signcrypts the data and sends them to the receiver.
- **Receiver**: the receiver receives the encrypted data, decrypts and checks the validity of the signature of the plaintext.

The signcryption scheme consists of the following five algorithms (which are sketched in Figure 1):

- $(pk_s, sk_m, par) \leftarrow$ `Setup`$(1^{\mathcal{K}})$: on the input of security parameter $\mathcal{K}$, the algorithm generates and publishes the public system parameters $par = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathbf{e}, g, g_2, \mathcal{H}, \mathrm{SYM})$, chooses and publishes the public key shared by all senders pk_s, and chooses the manager's private key sk_m which is kept secret. In particular, $\mathcal{H}$ is a predefined hash function and SYM is a predefined symmetric encryption scheme. The `Setup` algorithm is run by the manager.
- $(sk_i, rd, (pk_r, sk_r)) \leftarrow$ `KeyGen`(par, id_i, sk_m): this algorithm works in two phases. At first, on the input of the public system parameters par,

the manager's private key sk_m and the sender identifier id_i, this protocol outputs the sender's private key sk_i and the revocation database rd. The `KeyGen` algorithm first phase is run as an interactive protocol between the manager and the sender. At second, on input of the public system parameters par, this protocol outputs the receiver's private key sk_r and publishes the receiver's public key pk_r. This `KeyGen` algorithm second phase is run by the receiver.

- $(\pi, c) \leftarrow$ `Signcrypt`$(par, m, id_i, sk_i, pk_r)$: on the input of the public system parameters par, the message m, receiver's public key pk_r, sender's identifier id_i and its private key sk_i, the `Signcrypt` algorithm outputs the signature proof of knowledge π on the message m, and the ciphertext c of the message m. This algorithm is run by the sender.

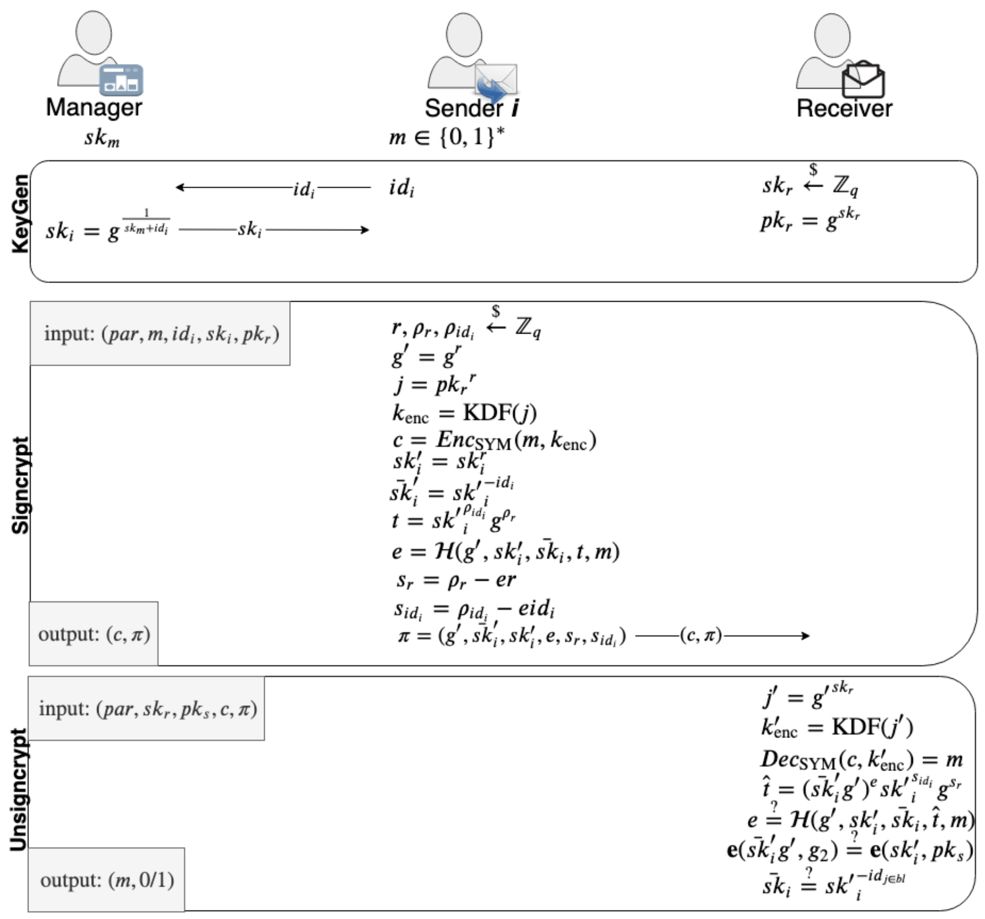
- $(m, 0/1) \leftarrow$ `Unsigncrypt`$(par, sk_r, pk_s, c, \sigma)$: on the input of the public system parameters par, receiver's private key sk_r, the public key pk_s, the ciphertext c and the signature σ, the `Unsigncryption` algorithm decrypts the ciphertext c and returns the message m, then verifies the signature σ and returns 1 and the message m iff the signature is valid and 0 otherwise. This algorithm is run by the receiver.

- $(id_i, bl) \leftarrow$ `Revoke`(rd, σ): on the input of the manager's revocation database rd and a signature σ, the algorithm outputs the identifier id_i of the signer and updates the black list bl. The `Revoke` algorithm is run by the manager.

5 Proposed scheme

In this section, the scheme is presented in details. Regarding the group signature scheme, we follow the proposal of Hajny et al. [15]. In particular, Weak Boneh-Boyen signature [5] and its efficient proof of knowledge [8] is used to sign messages. The wBB signatures were proven existentially unforgeable against a weak (non-adaptive) chosen message attack under the p-SDH assumption [5]. For the encryption, we take inspiration from the elliptic curve integrated encryption scheme proposed by Gayoso et al. [13]. The full notation specifications of our scheme are depicted in Figure 2.

In our proposal, a key derivation function KDF is needed. In particular, KDF maps elements of G_2 to the key space of SYM. The concrete algorithms can be found below.



Manager
sk_m
Sender i
$m \in \{0,1\}^*$
Receiver
KeyGen
id_i
id_i
$sk_r \xleftarrow{\$} \mathbb{Z}_q$
$pk_r = g^{sk_r}$
$sk_i = g^{\frac{1}{sk_m + id_i}}$
sk_i
Signcrypt
input: $(par, m, id_i, sk_i, pk_r)$
$r, \rho_r, \rho_{id_i} \xleftarrow{\$} \mathbb{Z}_q$
$g' = g^r$
$j = pk_r{}^r$
$k_{\text{enc}} = \text{KDF}(j)$
$c = Enc_{\text{SYM}}(m, k_{\text{enc}})$
$sk'_i = sk_i^r$
$\bar{sk}'_i = sk'^{-id_i}_i$
$t = sk'^{\rho_{id_i}}_i g^{\rho_r}$
$e = \mathcal{H}(g', sk'_i, \bar{sk}_i, t, m)$
$s_r = \rho_r - er$
$s_{id_i} = \rho_{id_i} - eid_i$
$\pi = (g', \bar{sk}'_i, sk'_i, e, s_r, s_{id_i})$
(c, π)
output: (c, π)
Unsigncrypt
input: $(par, sk_r, pk_s, c, \pi)$
$j' = g'^{sk_r}$
$k'_{\text{enc}} = \text{KDF}(j')$
$Dec_{\text{SYM}}(c, k'_{\text{enc}}) = m$
$\hat{t} = (\bar{sk}'_i g')^e sk'^{s_{id_i}}_i g^{s_r}$
$e \stackrel{?}{=} \mathcal{H}(g', sk'_i, \bar{sk}_i, \hat{t}, m)$
$\mathbf{e}(\bar{sk}'_i g', g_2) \stackrel{?}{=} \mathbf{e}(sk'_i, pk_s)$
$\bar{sk}_i \stackrel{?}{=} sk'^{-id_{j \in bl}}_i$
output: $(m, 0/1)$

Figure 2: Full notation of `KeyGen`, `Signcrypt` and `Unsigncrypt` algorithms.

5.1 `Setup` algorithm

The manager performs the five following steps:

1. Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be groups of the same prime order q. Let g be a generator of $\mathbb{G}_1$ and g_2 a generator of $\mathbb{G}_2$. Choose a bilinear map $\mathbf{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$.

2. Define a secure hash funtion $\mathcal{H} : \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{G}_1 \times \{0,1\}^{|m|} \to \mathbb{Z}_q^*$, where $|m|$ is the length of the plaintext message.

3. Choose a symmetric encryption scheme $\text{SYM} = (Enc_{\text{SYM}}, Dec_{\text{SYM}})$.

4. Choose $sk_m \xleftarrow{\$} \mathbb{Z}_q^*$ as manager's private key, and set $pk_s = g_2^{sk_m}$ as senders' public key.

5. Publish the public system security parameters $par = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathbf{e}, g, g_2, \mathcal{H}, \mathrm{SYM})$ and keep sk_m secret.

5.2 `KeyGen` algorithm

First phase. With public system security parameters par, manager's secret key sk_m and sender's identifier id_i as input, the manager performs the following steps:

1. Compute the sender's private key $sk_i = g^{\frac{1}{sk_m + id_i}}$ and send it to sender i.

2. Update the manager's revocation database rd by storing id_i.

Second phase. With public system parameters par, the receiver performs the following steps:

1. Randomly choose a private key $sk_r \xleftarrow{\$} \mathbb{Z}_q^*$.

2. Compute and publish its public key $pk_r = g^{sk_r}$.

5.3 `Signcrypt` algorithm

With public system security parameters par, the message m, receiver's public key pk_r, sender's identifier id_i and its private key sk_i, sender i generates the ciphertext c and the signature proof of knowledge π on m as follows:

1. Randomly choose randomizers $r, \rho_r, \rho_{id_i} \xleftarrow{\$} \mathbb{Z}_q^*$, and compute $g' = g^r$ and $j = pk_r^r$.

2. Generate a symmetric key $k_{\text{enc}} = \mathrm{KDF}(j)$.

3. Encrypt the message $c = Enc_{\mathrm{SYM}}(m, k_{\text{enc}})$ by a symmetric encryption scheme.

4. Compute the values $sk_i' = sk_i^r$, $\bar{sk}_i' = {sk'}_i^{-id_i}$, $t = {sk'}_i^{\rho_{id_i}} g^{\rho_r}$, $e = \mathcal{H}(g', sk_i', \bar{sk}_i, t, m)$, $s_r = \rho_r - er$, and $s_{id_i} = \rho_{id_i} - eid_i$ necessary to generate the signature proof of knowledge $\pi = \{(id_i, r) : \bar{sk}_i' = {sk'}_i^{-id_i} \wedge g' = g^r\}$.

5. Send (π, c) to the receiver, where π is $(g', \bar{sk}_i', sk_i', e, s_r, s_{id_i})$.

5.4 `Unsigncrypt` algorithm

When receiving (π, c), the receiver decrypts the message and, then, verifies the signature:

Decrypt

1. Compute $j' = g'^{sk_r}$ and k'_{enc} =KDF(j').
2. Recover the message $m = Dec_{\text{SYM}}(c, k'_{\text{enc}})$.

Verify
The receiver computes $\hat{t} = (\bar{sk}'_i g')^e {sk'}_i^{s_{id_i}} g^{s_r}$, and checks if the equations $e \overset{?}{=} \mathcal{H}(g', sk'_i, \bar{sk}_i, \hat{t}, m)$ and $\mathbf{e}(\bar{sk}'_i g', g_2) \overset{?}{=} \mathbf{e}(sk'_i, pk_s)$ hold. If yes, the receiver also performs the revocation check $\bar{sk}_i \overset{?}{=} {sk'}_i^{-id_j}$ for all id_j values stored on the blacklist bl. If the revocation check equation holds for any value on the blacklist, the signature is rejected. Otherwise, the signature is accepted.

5.5 `Revoke` algorithm

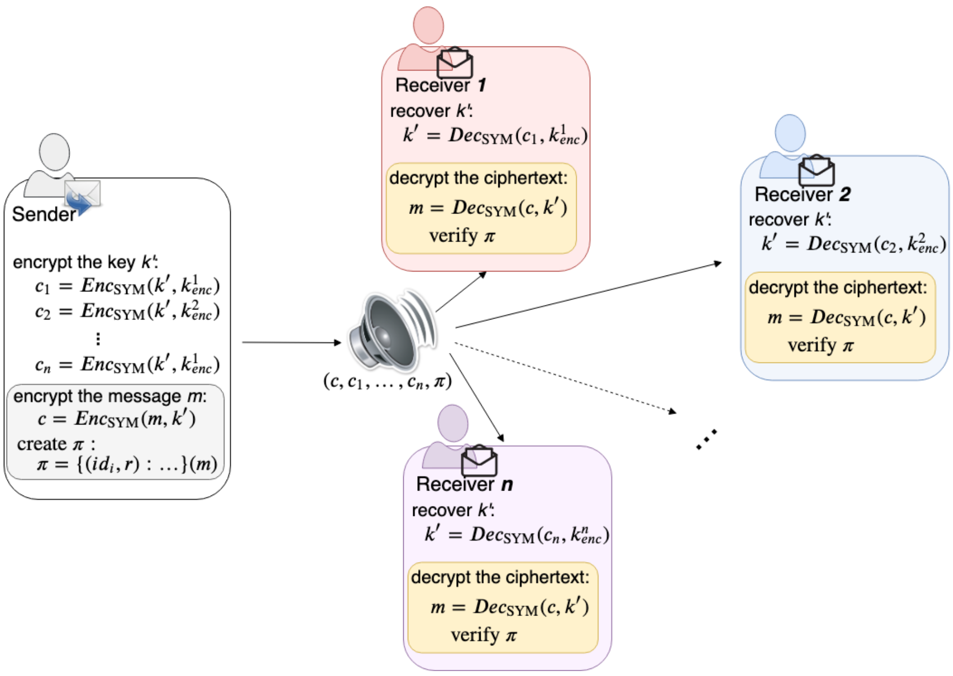
With revocation database rd and signature proof of knowledge π, the manager checks if the equation $\bar{sk}_i \overset{?}{=} {sk'}_i^{-id_j}$ holds for all id_js in rd. If there exists an id_j for which this equation holds, id_j is put on the public blacklist bl.

6 Multi-receiver scenario

The proposed signcryption scheme can be easily adapted to a multi-reciever scenario. A sketch of our protocol is depicted in Figure 3.

In this case, the message m is encrypted using a common key k' which is sent encrypted to each authorized receiver. In particular, the sender has to encrypt k' accordingly to receivers' public key and, therefore, produce n different encryptions of k' in case of n receivers. `Setup`, `KeyGen` and `Revoke` algorithms remain unchanged, while `Signcrypt` and `Decrypt` algorithms are slightly modified (the modifications are marked in bold).

Let $k' \xleftarrow{\$} \{0,1\}^*$ be our common key for SYM and let us suppose to have n receivers. Receiver i holds the key pair (pk_{r_i}, sk_{r_i}).



Sender
encrypt the key k':
$c_1 = Enc_{\mathrm{SYM}}(k', k^1_{enc})$
$c_2 = Enc_{\mathrm{SYM}}(k', k^2_{enc})$
$\vdots$
$c_n = Enc_{\mathrm{SYM}}(k', k^1_{enc})$
encrypt the message m:
$c = Enc_{\mathrm{SYM}}(m, k')$
create π :
$\pi = \{(id_i, r) : \ldots\}(m)$
$(c, c_1, \ldots, c_n, \pi)$
Receiver *1*
recover k':
$k' = Dec_{\mathrm{SYM}}(c_1, k^1_{enc})$
decrypt the ciphertext:
$m = Dec_{\mathrm{SYM}}(c, k')$
verify π
Receiver *2*
recover k':
$k' = Dec_{\mathrm{SYM}}(c_2, k^2_{enc})$
decrypt the ciphertext:
$m = Dec_{\mathrm{SYM}}(c, k')$
verify π
Receiver *n*
recover k':
$k' = Dec_{\mathrm{SYM}}(c_n, k^n_{enc})$
decrypt the ciphertext:
$m = Dec_{\mathrm{SYM}}(c, k')$
verify π

Figure 3: Our multi-receiver signcryption proposal. Observe that the decryption of c and the verification of π are equal for all the receivers.

6.1 Signcrypt algorithm

1. **For** $w = 1$ **to** n:

    1. Randomly choose secret integers $r, \rho_r, \rho_{id_i} \xleftarrow{\$} \mathbb{Z}_q^*$, and compute $g' = g^r$ and $\mathbf{j_w} = pk^r_{\mathbf{r_w}}$ where pk_{r_w} is the public key of receiver w.
    2. Generate a symmetric key $\mathbf{k^w_{enc}} = \mathrm{KDF}(\mathbf{j_w})$, where KDF states for Key Derivation Function.

    2b. **encrypt the key** $\mathbf{c_w} = \mathbf{Enc_{SYM}}(\mathbf{k'}, \mathbf{k^w_{enc}})$.

3. Encrypt the message, $c = Enc_{\mathrm{SYM}}(m, \mathbf{k'})$.

4. Compute the values $sk'_i = sk^r_i$, $\bar{sk}'_i = sk'^{-id_i}_i$, $t = sk'^{\rho_{id_i}}_i g^{\rho_r}$, $e = \mathcal{H}(g', sk'_i, \bar{sk}_i, t, m)$, $s_r = \rho_r - er$, and $s_{id_i} = \rho_{id_i} - eid_i$ necessary to generate the signature proof of knowledge π.

5. Send $(c, \sigma, \mathbf{c_1}, \ldots, \mathbf{c_n})$, where σ is $(g', \bar{sk}'_i, sk'_i, e, s_r, s_{id_i})$.

Therefore, if we have n receivers, the scheme requires n additional encryptions of k' compared to the traditional scheme and the sender will publish $(c, \sigma, c_1, \ldots, c_n)$.

6.2 `Unsigncrypt` algorithm

In `Unsigncrypt` algorithm, the `Verify` algorithm remain unchanged, while the `Decrypt` protocol needs to be modified. This algorithm is run by each receiver. In case of receiver w, it works as follows:

Decrypt

1. Compute $j' = g'^{sk_{r_w}}$ and k'^{w}_{enc} =KDF(j').

2a. **Recover the key** $\mathbf{k}' = \mathbf{Dec_{SYM}}(\mathbf{c_w}, \mathbf{k^w_{enc}})$

2. Recover the message $m = Dec_{\text{SYM}}(c, \mathbf{k}')$.

The verification part remains unchanged.

7 Security Analysis

In this section, we prove that the proposed scheme satisfies correctness, confidentiality (IND-CCA2), unforgeability (sUF-CMA) and ciphertext anonymity (ANON-CCA) [38]. Moreover, we remark that our signcryption scheme also guarantees unlinkability, traceability and sender anonymity thanks to the considered group signature [15].

7.1 Corretness

Theorem 1. *The decryption process in Section 5.4 is correct.*

Proof. Since a symmetric cryptographic scheme is used to encrypt the message, at first we show that the receiver can reconstruct the sender's key. In fact,

$$\begin{aligned} j' &= g'^{sk_r} = (g^r)^{sk_r} = pk_r^r = (g^{sk_r})^r = j \\ k'_{\text{enc}} &= \text{KDF}(j') = \text{KDF}(j) = k_{\text{enc}} \end{aligned}$$

and, therefore, $Dec_{\text{SYM}}(c, k'_{\text{enc}}) = Dec_{\text{SYM}}(c, k_{\text{enc}}) = m$. Accordingly, the decryption process is correct. □

Theorem 2. *The verification process in Section 5.4 is correct.*

Proof. Once the message is correctly decrypted, we need to show that $\hat{t}$ is equal to t. This can be proven as follows,

$$\begin{aligned}
\hat{t} &= (\bar{sk}'_i g')^e {sk'}_i^{s_{id_i}} g^{s_r} \\
&= ({sk'}_i^{-id_i} g^r)^e {sk'}_i^{s_{id_i}} g^{s_r} \\
&= {sk'}_i^{-eid_i} g^{er} {sk'}_i^{s_{id_i}} g^{s_r} \\
&= {sk'}_i^{-eid_i} g^{er} {sk'}_i^{\rho_{id_i} - eid_i} g^{s_r} \\
&= g^{er} {sk'}_i^{\rho_{id_i}} g^{s_r} \\
&= g^{er} {sk'}_i^{\rho_{id_i}} g^{\rho_r - er} \\
&= {sk'}_i^{\rho_{id_i}} g^{\rho_r} = t.
\end{aligned}$$

Therefore, $e = \mathcal{H}(g', sk'_i, \bar{sk}_i, t, m) = \mathcal{H}(g', sk'_i, \bar{sk}_i, \hat{t}, m)$. In order to accept the signature, the receiver also needs that $\mathbf{e}(\bar{sk}'_i g', g_2) \overset{?}{=} \mathbf{e}(sk'_i, pk_s)$ holds. For a valid signature, we have that

$$\begin{aligned}
\mathbf{e}(\bar{sk}_i g', g_2) &= \mathbf{e}(sk'_i, pk_s) \\
\mathbf{e}(sk_i^{-id_i r} g^r, g_2) &= \mathbf{e}(sk_i^r, g_2^{sk_m}) \\
\mathbf{e}(g^{\frac{-id_i r}{sk_m + id_i}} g^r, g_2) &= \mathbf{e}(sk_i^r, g_2^{sk_m}) \\
\mathbf{e}(g^{\frac{sk_m r + id_i r - id_i r}{sk_m + id_i}}, g_2) &= \mathbf{e}(sk_i^r, g_2^{sk_m}) \\
\mathbf{e}(sk_i^{sk_m r}, g_2) &= \mathbf{e}(sk_i^r, g_2^{sk_m}) \\
\mathbf{e}(sk_i, g_2)^{sk_m r} &= \mathbf{e}(sk_i, g_2)^{sk_m r}.
\end{aligned}$$

At last, the receiver needs that $\bar{sk} \overset{?}{=} sk'^{-id_i}$ does not hold for all id_i stored on the blacklist bl. This works if and only if $id_i \notin bl$ by definition, i.e. the sender is a valid one. Therefore, the correctness of message and signature is proven. □

7.2 Unforgeability

Boneh and Boyen [5] prove that the wBB signature scheme is strong existentially unforgeable against an adaptive chosen message attack under the p-SDH assumption. The sUF-CMA of our scheme follows from the unforgeability of wBB signature (see Lemma 9 in [5]) and uses the same proof technique. We consider an attacker who makes up to q_s adaptive signcryption and unsigncryption queries, and reduce the forgery to the resolution of a random p-SDH instance for $p = q_s$.

Theorem 3. *Suppose the (p, t', ϵ)-SDH assumption holds in $(\mathbb{G}_1, \mathbb{G}_2)$. Then the signcryption scheme proposed in Section 5 is (t, q_s, ϵ)-secure against existential forgery under adaptive chosen message attack with*

$$q_s \leq p \text{ and } t \leq t' - \Theta(pT)$$

where T is the maximum time for an exponentiation in $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{Z}_q$.

Proof. We prove that if $\mathcal{A}$ can (t, q_s, ϵ)-break the signcryption scheme, then there exists an algorithm $\mathcal{B}$ such that, by interacting with $\mathcal{A}$, solves the p-SDH problem in time t' with advantage ϵ. Let $(g, d_1, d_2, \ldots, d_p, g_2, h)$ be a random instance of the p-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$, where $d_i = g^{x^i} \in \mathbb{G}_1$ for $i = 1, \ldots, p$ and $h = g_2^x \in \mathbb{G}_2$ for some unkown $x \in \mathbb{Z}_q$. Let $g = d_0$ and $x = sk_m$ for convenience. The goal of $\mathcal{B}$ is to compute the pair $(c, g^{1/(x+c)}) \in \mathbb{Z}_q \times \mathbb{G}_1$ for some value $c \in \mathbb{Z}_q \setminus \{x\}$ of its choice.
$\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

Query: $\mathcal{A}$ outputs a list of $q_s \leq p$ messages $m_1, \ldots, m_{q_s} \in \mathbb{Z}_q$. We can suppose that $q_s = p$ for simplicity. If less queries are made, we can always reduce the value of p to $p' = q_s$.

Response: $\mathcal{B}$ responds with p pairs $(c_i, \pi_i) \leftarrow \texttt{Signcrypt}(par, sk_s, pk_r, m_i)$ and p "signed" messages $(m_i, 0/1) \leftarrow \texttt{Unsigncrypt}\ (par, pk_s, pk_r, c_i, \sigma_i)$. Therefore, $\mathcal{A}$ obtains p signature proofs of knowledge on its input messages.

Let f be the univariate polynomial defined as $f(X) = X + id_i$. $\mathcal{B}$ chooses $\theta \in \mathbb{Z}_q^*$ and computes

$$g_1 = g^{\theta f(x)}$$

Therefore, $\mathcal{A}$ receives identity id_i, parameters $\widehat{par} = (q, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \mathbf{e}, g_1, g_2, \mathcal{H}, \text{SYM})$ and public key $pk_s = h$.
If $f(x) = 0$, then $x = -id_i$ and $\mathcal{B}$ can easily recover the secret key x and solve the p-SDH problem. If $f(x) \neq 0$, then g_1 and g_2 are independently and uniformly distributed random generators for the respective groups due to the action of ϕ. In this case, $\mathcal{B}$ has to apply both `Signcrypt` and `Unsigncrypt` algorithms and generate a valid signature proof of knowledge π_j on each message m_j, for $j = 1, \ldots, p$. To do so, by following `Signcrypt` algorithm, $\mathcal{B}$ chooses at random r, ρ_r, ρ_{id_i}, encrypts m_j and creates the proof: $\pi_j = (g', \bar{sk}_i', sk_i', e, s_r, s_{id_i})$ where all the computations are made using g_1 instead of g. In fact, if g_1 is used, then $sk_i = g_1^{1/f(x)} = g^\theta$ and $\mathcal{B}$ can compute each other component of the signature easily. This is repeated for each message m_j, where $j = 1, \ldots, p$.

Observe that π_j is a valide proof for m_j under $\widehat{par}$, since

$$\begin{aligned}\mathbf{e}(\bar{sk}'_i g', g_2) =& \mathbf{e}(g_1^{-id_i r/f(x)} g_1^r, g_2) = \mathbf{e}(g^{-\theta id_i r f(x)/f(x)} g^{r\theta f(x)}, g_2) = \\ & \mathbf{e}(g^{\theta r(f(x)-id_i)}, g_2) = \mathbf{e}(g^{\theta r x}, g_2) = \mathbf{e}(g^{\theta r x}, g_2) = \\ & \mathbf{e}(g^{\theta f(x) r/f(x)}, g_2^x) = \mathbf{e}(sk'_i, pk_s)\end{aligned}$$

The fact that $e = \mathcal{H}(g', sk'_i, \bar{sk}_i, \hat{t}, m_j)$ follows straightforward from the correctness of our scheme, see Theorem 2. These are exactly the verification steps performed by $\mathcal{B}$ when it applies `Unsigncrypt` algorithm and, therefore, links each message m_j to its proof π_j. Since each message admits only a unique signature proof of knowledge, the output distribution is trivially correct.

Output: $\mathcal{A}$ returns for a user's identity id_* a forgery (m_*, π_*) such that π is a valid proof of $m_* \notin \{m_1, \ldots, m_p\}$. The signature proof of knowledge π_* is a vector $\pi_* = (g', \bar{sk}'_i, sk'_i, e, s_r, s_{id_i})$ computed using the parameters $\widehat{par}$. We suppose that $id_* \neq id_i$ since $\mathcal{A}$ can choose id_* knowing id_i. By construction and uniqueness of the proof, we know that the component sk_i is equal to $\sigma_* := g_1^{\frac{1}{x+id_*}} = g^{\frac{\theta f(x)}{x+id_*}}$ where $f(x) = x + id_i$. If $x = -id_*$, then $\mathcal{B}$ can easily recover the secret key x and solve the p-SDH problem. Otherwise, note that the polynomial f can be rewritten as $f(x) = x + id_* + \gamma_*$ where $\gamma_* = id_i - id_* \in \mathbb{Z}_q$. Therefore, the ratio $f(x)/(x + id_*)$ can be written as $f(x)/(x + id_*) = 1 + \frac{\gamma_*}{x+id_*}$ and the espression of σ^* becomes

$$\sigma_* = g^{\theta(1+\frac{\gamma_*}{x+id_*})}$$

Taking roots of order θ and $\gamma_* \mod q$, $\mathcal{B}$ can compute

$$\omega = (\sigma_*^{1/\theta} g^{-\theta})^{1/\gamma^*} = g^{1/x+id_*} \in \mathbb{G}_1 \tag{1}$$

and obtain the pair (id_*, ω) as solution to the submitted instance of the p-SDH problem.

The claimed bound is obvious by construction of the reduction. □

7.3 Indistinguishability

Smart [34] analyzes the security of a generic ECIES scheme, in particular, he focuses on the indistinguishability under adaptive chosen ciphertext attacks. The IND-CCA2 of our scheme follows the same proof technique of ECIES indistinguishability (see Section 4 in [34]). We consider an attacker who makes up to q_s adaptive signcryption and unsigncryption queries.

Theorem 4. *Suppose the (q, t', ϵ')-DDH and (p, t'', ϵ'')-SDH assumptions hold in $\mathbb{G}_1$ and $(\mathbb{G}_1, \mathbb{G}_2)$, respectively. Then the signcryption scheme proposed in Section 5 is (t, q_s, ϵ)-secure against indistinguishability under adaptive chosen ciphertext attacks with*

$$q_s \leq p \text{ and } t \leq 2t''' + t'' + \frac{2q_s^2}{q} - \Theta(q_{s''}T)$$

where T is the maximum time for an exponentiation in $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{Z}_q$.

Proof. We prove this theorem using Lemma 5 which allows bounding the advantage of winning the IND-CCA2 game. Since $Adv_{\mathcal{B}}^{DDH} = \frac{q_s^2}{p}$, where q_s is the number of queries that $\mathcal{A}$ makes (as proven by Shoup [35], Theorem 4), the claimed bound is obvious by construction.

$\square$

Lemma 5. *For any adversary $\mathcal{A}$ running in time t and making at most $q_s = q_{s'} + q_{s''}$ signcryption and unsigncryption queries, the advantage of winning the IND-CCA2 game is*

$$Adv_{\mathcal{A}}^{IND}(t, q_s) \leq 2Adv_{\mathcal{B}}^{DDH}(t', q) + 2Adv_{\mathcal{B}}^{SDH}(t'', p) + Adv_{\mathcal{B}}^{SYM}(t''', |\mathcal{K}|)$$

where

- $Adv_{\mathcal{B}}^{DDH}(t', q)$ *is the maximal probability of solving the DDH assumption in time t'.*
- $Adv_{\mathcal{B}}^{SDH}(t'', p)$ *is the maximal probability of solving the SDH assumption in time t''.*
- $Adv_{\mathcal{B}}^{SYM}(t''', |\mathcal{K}|) = 2\mathbf{Pr}[b' = b] - 1$ *is the maximal advantage of any adversary mounting a chosen plaintext attack on SYM in time t''' with key size $|\mathcal{K}|$.*

Proof. We wish to use $\mathcal{A}$ to attack the security of DDH problem, the underlying SYM and proposed group signature (GS) schemes. During the proof the bitlength of the messages is bounded by μ.
Game 1. Following the definition of IND-CCA2 game (Section 3.4.2), the below game is used to break the encryption scheme. $\mathcal{C}$ and $\mathcal{A}$ do as follows:

> **Setup**: $\mathcal{C}$ runs algorithms `Setup` and `KeyGen` to generates the public system security parameters par, senders' public key pk_s, manager's private key sk_m, sender's i private key sk_i and receiver's key pair (sk_r, pk_r). $\mathcal{A}$ is given par, pk_r and pk_s.

Queries-1: $\mathcal{A}$ requests signcryption of at most $q_{s'}$ messages of its choice $m_1, \dots, m_{q_{s'}} \in \{0,1\}^*$. $\mathcal{C}$ responds to each query with a signature and a ciphertext $(c_i, \pi_i) \leftarrow \texttt{Signcrypt}(par, sk_s, pk_r, m_i)$. Then, proceeding adaptively, $\mathcal{A}$ requests unsigncryption of at most $q_{s'}$ ciphertexts $c_1, \dots, c_{q_{s'}}$, under pk_s and pk_r. $\mathcal{C}$ responds to each query with 1 and a message $(m_i, 0/1) \leftarrow \texttt{Unsigncrypt}\ (par, pk_s, pk_r, c_i, \pi_i)$ if the obtained plaintext is valid and with 0 otherwise.

Challenge: $\mathcal{A}$ outputs two equal-length messages m'_0 and $m'_1 \in \{0,1\}^*$ on which it wishes to be challenged. Then, hidden from $\mathcal{A}$ view, $\mathcal{C}$ chooses $b \leftarrow \{0,1\}$ and computes the challenge ciphertext $(c'_*, \pi'_*) \leftarrow \texttt{Signcrypt}(sk_s, pk_r, m'_b)$.

Queries-2: $\mathcal{A}$ may request at most $q_{s''}$ signcryption and unsigncryption quieries as in Queries-1 phase but with the restriction that $\mathcal{A}$ can not query for c'_*.

Guess: $\mathcal{A}$ produces its guess b' of b. $\mathcal{A}$ is successful if $b' = b$, i.e. the guess is correct.

Therefore, $Adv_{\mathcal{A}}^{IND}(t, q_s) = 2\mathbf{Pr}[b' = b] - 1$ represents the probability that $\mathcal{A}$ wins in the above game in time t with at most q_s signcryption and unsigncryption queries.

Since $\mathcal{A}$ is not allowed querying the unsigncryption protocol for the target cipher c'_*, namely **Type $\mathbf{Q}_\perp$** query, $\mathcal{A}$ cannot have access to Dec_{SYM} for the key k_{enc} corresponding to c'_*. In case a **Type $\mathbf{Q}_\perp$** query is made, Dec_{SYM} will output $\gamma \in \{0,1\}$. Let **Type $\mathbf{Q_v}$** be any valid query different from **Type $\mathbf{Q}_\perp$**.

Game 2. In this game, we prove that if $\mathcal{A}$ can $(t, q_s, \mu, m, \epsilon)$-break the signcryption scheme, then there exists an algorithm $\mathcal{B}$ such that, by interacting with $\mathcal{A}$, solves the DDH problem in time t' with advantage ϵ'. Let $\langle g, g^a, g^b, g^c \rangle$ be a random instance of DDH problem in $\mathbb{G}_2$, where $a, b, c \in \mathbb{Z}_q^*$. The goal is to determine whether $c \equiv ab \mod q$.

Therefore, *Game 2* is the same as *Game 1* but $\mathcal{B}$ has as input the following values: $(sk_r = b, pk_r = g^b)$, $r = a$, i.e. $g' = g^a$, and $j = g^c$ (see Figure 2 for more details on the protocol). In this way, k_{enc} and k'_{enc} are equal if and only if $c \equiv ab \mod q$. In other words, $\mathcal{A}$ believes that c is equal to $ab \mod q$ if $\mathcal{A}$ is successful in the game, i.e. $b = b'$.

We have three different situations depending on chosen DDH problem instance and query type.

1. When a valid DDH problem instance is given as input, $\mathcal{A}$ runs $\mathcal{B}$ as if one wants to mount an attack against the proposed signcryption

protocol. Therefore,

$$Adv_B^{\mathrm{DDH}}(t', q) = \frac{1 + Adv_{\mathcal{A}}^{IND}(t, q_s)}{2} \tag{2}$$

2. If a non-valid DDH problem instance is given as input and $\mathcal{A}$ makes a **Type $\mathbf{Q}_\perp$** query, $\mathcal{A}$ runs $\mathcal{B}$ as if one wants to mount an attack against SYM. Therefore,

   $$Adv_B^{\mathrm{DDH}}[(t', q) \wedge \textbf{Type } \mathbf{Q}_\perp] \leq \frac{1 + Adv_{\mathcal{B}}^{\mathrm{SYM}}(t''', |\mathcal{K}|)}{2} \tag{3}$$

   where the inequality appears since $\mathcal{B}$ makes 0 signcryption queries in order to break SYM.

3. If a non-valid DDH problem instance is given as input and $\mathcal{A}$ makes a **Type $\mathbf{Q}_\mathbf{v}$** query, the game is the same of breaking GS and, therefore, breaking the p-SDH problem. The proof follows straightforward from the sUF-CMA (Theorem 3) of the signcryption scheme. Indeed, the fact that $k_{enc} \neq k'_{enc}$ does not affect the computation of σ'_* and `Verify` phase of `Unsigncrypt` algorithm since m'_0 and m'_1 are known.

   Therefore, we have

   $$Adv_B^{\mathrm{DDH}}[(t', q) \wedge \textbf{Type } \mathbf{Q}_\mathbf{v}] \leq Adv_{\mathcal{B}}^{SDH}(t'', p) \tag{4}$$

   where the inequality appears since $\mathcal{B}$ only requires one round of signcryption and unsigncryption queries, i.e. $q'_s \leq q_s$ signcryption and unsigncryption queries in order to break GS.

Finally, combining Equations 2, 3 and 4, we obtain

$$Adv_B^{\mathrm{DDH}}(t', q) \geq \frac{1 + Adv_{\mathcal{A}}^{IND}(t, q_s)}{2} - \frac{1 + Adv_{\mathcal{B}}^{\mathrm{SYM}}(t''', |\mathcal{K}|)}{2} - Adv_{\mathcal{B}}^{SDH}(t'', p)$$

and the claimed bound directly follows from the last inequality. □

7.4 Ciphertext anononymity

Ciphertext anonymity property is satisfied if ciphertexts reveal no information about who created them nor about whom they are intended to [38]. In particular, this exactly covers that sender's and receiver's identities are hidden to outsiders.

We consider an attacker who makes up to q_s adaptive signcryption and unsigncryption queries.

Theorem 6. *Suppose the (q, t', ϵ')-DDH and (p, t'', ϵ'')-SDH assumptions hold in $\mathbb{G}_1$ and $(\mathbb{G}_1, \mathbb{G}_2)$, respectively. Then the signcryption scheme proposed in Section 5 is (t, q_s, ϵ)-secure against anonymity under adaptive chosen ciphertext attacks with*

$$q_s \leq p \text{ and } t \leq 4t''' + 2t'' + \frac{4q_s^2}{q} + \frac{1}{4} - \Theta(q_{s''}T)$$

where T is the maximum time for an exponentiation in $\mathbb{G}_1, \mathbb{G}_2$ and $\mathbb{Z}_q$.

Proof. We prove this theorem using Lemma 7 which allows bounding the advantage of winning the ANON-CCA game. Since $Adv_{\mathcal{B}}^{DDH} = \frac{q_s^2}{p}$, where q_s is the number of queries that $\mathcal{A}$ makes (as proven by Shoup [35], Theorem 4), the claimed bound is obvious by construction.

□

Lemma 7. *For any adversary $\mathcal{A}$ running in time t and making at most $q_s = q_{s'} + q_{s''}$ signcryption and unsigncryption queries, the advantage of winning the ANON-CCA game is*

$$Adv_{\mathcal{A}}^{ANON}(t, q_s) \leq 4Adv_{\mathcal{B}}^{DDH}(t', q) + 4Adv_{\mathcal{B}}^{SDH}(t'', p) + 2Adv_{\mathcal{B}}^{SYM}(t''', |\mathcal{K}|) + \frac{1}{4}$$

where $Adv_{\mathcal{B}}^{DDH}(t', q)$, $Adv_{\mathcal{B}}^{SDH}(t'', p)$ and $Adv_{\mathcal{B}}^{SYM}(t''', |\mathcal{K}|)$ are defined as in Lemma 5.

Proof. The proof of this lemma follows the same structure of Lemma 5. Since it would be redundant to rewrite the same proof two times, we just sketch it emphasizing the main difference.

As in Lemma 5, we wish to use $\mathcal{A}$ to attack the security of DDH problem, the underlying SYM and the proposed GS schemes. During the proof the bitlength of the messages is bounded by μ.

Game 1. In this case, we consider ANON-CCA game (Section 3.4.3), where

$$Adv_{\mathcal{A}}^{ANON} = |4\mathbf{Pr}[(b', d') = (b, d)] - 1| \quad (5)$$

is the probability that the adversary $\mathcal{A}$ wins the ANON-CCA game for our proposed signcryption scheme.

As above, $\mathcal{A}$ can do two different queries: **Type $\mathbf{Q}_\perp$** query, which is $\mathcal{A}$ querying for (c'_*, σ_*), and **Type $\mathbf{Q_v}$** query, that is any valid query.

Game 2. As above, if $\mathcal{A}$ can $(t, q_s, \mu, m, \epsilon)$-break the signcryption scheme, then there exists an algorithm $\mathcal{B}$ such that, by interacting with $\mathcal{A}$, solves the DDH problem in time t' with advantage ϵ'. Let $\langle g, g^a, g^b, g^c \rangle$ be a random instance of DDH problem in $\mathbb{G}_2$, where $a, b, c \in \mathbb{Z}_q^*$. The goal is to determine whether $c \equiv ab \mod q$. As in Lemma 5, we have three different situations, where only the first one is slightly different from the previous proof:

1. When a valid DDH problem instance is given as input, $\mathcal{A}$ runs $\mathcal{B}$ as if one wants to mount an attack against the proposed signcryption protocol, therefore,

$$Adv_B^{\mathrm{DDH}}(t', q) = \frac{1 + Adv_{\mathcal{A}}^{ANON}(t, q_s)}{4} \tag{6}$$

   Observe that the denominator is 4, since this equality is derived from Equation 5.

2. If a non-valid DDH problem instance is given as input and $\mathcal{A}$ makes a **Type $\mathbf{Q}_{\perp}$** query, we have Equation 3.

3. If a non-valid DDH problem instance is given as input and $\mathcal{A}$ makes a **Type $\mathbf{Q}_{\mathrm{v}}$** query, the game is the same of breaking GS and, therefore, we have Equation 4.

Finally, combining Equations 3, 4 and 6, we obtain the claimed bound.

□

7.5 Unlinkability, traceability and sender anonymity

It is important to notice that unlinkability, traceability and sender's anonymity are privacy-enhancing features achieved thanks to the usage of our previously proposed group signature [15].

This group signature is integrated with the zero-knowledge proofs, i.e. the sender proves the knowledge of messages and signatures themselves. In particular, without the knowledge of witnesses id_i and randomizers r, these proofs are provably *unlinkable*. Moreover, since the manager knows the witnesses id_i, it is able to efficiently link all the proofs (*traceability*). Regarding *sender anonymity*, any sender can sign message on behalf of a group, therefore, its identity is hidden inside the group. See [8] for more details.

8 Example of possible use case

In this section, a possible use case suitable for our proposal is presented. As mentioned before, smart cities depend on new technologies as global infrastructure network, sensors and wireless devices which exchange data which may be sensitive in real time. For instance, this is the case of sharing vehicles.

Sharing vehicles is a kind of rental where cars, bycicles or scooters are rented out for short periods of time in localized areas. These vehicles are connected to central cloud servers which identify, authenticate and autorize the usage of all these means of transport. In particular, this cloud provider stores all the information related to the vehicle such as acceleration data, GPS coordinates and speed information. Observe that modern cars are equipped with up to 70 embedded electronic sensors which are connected via various communication buses. This connectivity can expose smart vehicles and, therefore, citiziens to security and privacy threats such as location tracking or remote hijacking of the vehicle [4]. It is the case of Miller and Valasek [25] that could hack a car while a journalist was driving it.

Moreover, most of the current secure communication architectures do not consider user privacy [4, 24, 37]. In fact, the vehicle data are normally exchanged without the owner's permission. Conversely, Dorri et al. [11] propose a blockchain-based architecture to protect user privacy in vehicular environment. In their proposal, user's identity is hidden to anyone but the block manager who needs to know the connection between user's and their public keys. In this way, the block managers can reconstruct all the movements done by each user. Unluckily, even a sparse set of geolocated data, when coupled with user identifiers, can reveal personal user information to potentially untrusted third parties [30].

Our proposal guarantees an encrypted communication to securely exchange data as vehicle location, keys to unlock the vehicle, and payment details of the user. Moreover, the user's privacy is guaranteed due to the group signature usage. In fact, the user (sender) anonymously signes a message in behalf of the group and, therefore, hides its identity even to the cloud service provider (receiver). However, if a user refuses a payment or has other malicious behaviour, the provider of the services (receiver) in collaboration with the manager can retrieve the malicious user's identity.

9 Experimental results

This section provides the whole protocol implementation and the implementation aspects discussion. Current IoT networks consist of many resource-constrained devices with limited computational and storage capabilities. In order to cover the vast majority of possible use cases, we decide to employ these devices to our testing scenario. The main purpose is to demonstrate the efficiency and the practical potential of our scheme. In particular, we consider ARM-platform (Raspberry Pi) and smart card platforms (JavaCard & MultOS). Their specifications are described in Sections 9.1 and 9.2, while testing scenario and evaluations are presented in Section 9.3.

9.1 Smart Card Selection

Smart cards (SCs) are a closed platform. This means that is not usually possible to upgrade cryptographic libraries on the card. SC cryptographic support differs according to: 1) the SC platform (e.g. Java Card, MultOS and Basic Card), 2) the version of the operating system, and 3) the SC implementation itself.

Table 2: Technical specification of tested smart cards.

	J3D081	**Sm@rtCafe6**	**ZC7.6**	**ML4**	**ML3**
MCU	P5CD081	P5CD081	–	SC23Z018	SLE78CLXPM
OS	Java Card	Java Card	Basic Card	MultOS	MultOS
Version	3.0.1	3.0.1	ZC7	4.3.1	4.3.1
ROM	264KB	264KB	–	252KB	280KB
EEPROM	80KB	80KB	72KB	18KB	96KB
RAM	6KB	6KB	4.3KB	1.75KB	2KB

For our tests, the newest cards in the market (for each card platform one representative) were selected and their HW/SW properties and cryptographic support were compared. The technical specification of tested SCs is shown in Table 2. Current SCs usually have only 8-bit, 16-bit (or 32-bit in really special cases) processors, and small RAM and EEPROM memories. These limited resources make the development of novel cryptographic protocols very difficult. On the other hand, SCs are equipped with a co-processor, which allows developers to accelerate specific cryptographic operations and algorithms.

Note that our proposal requires 1) a symmetric encryption algorithm to encrypt data, and 2) algebraic operations over finite field and a secure hash algorithm to generate a signature. These simple requests are not of easy support for current SCs. The cryptographic support in accordance with our signcryption scheme requirements is shown in Table 3. It is important to note that nowadays there is no one smart card platform that supports bilinear pairing operations. In particular, MultOS and Basic Cards are the only platforms which allow the access to modular and elliptic curve operations.

EC support and speed are crucial for our implementation, and therefore we compared the speed of individual SC platforms. Figure 4 depicts the EC scalar multiplication `ecMul` (which is the most computationally demanding operation of `Signcrypt` protocol) cost for Brainpool curves for different elliptic curve sizes. MultOS card is 75% faster than Basic card (ZC7.6) and 35% faster than the fastest Java card (J3D081). Sm@rtCafe implementation shows a bit worse results than JCOP SC implementation.

Table 3: Cryptographic support on tested smart cards.

	Algorithm	**J3D081**	**Sm@rtCafe6**	**ZC7.6**	**ML4**	**ML3**
Crypto	`SHA1`	✔	✔	✔	✔	✔
	`SHA256`	✔	✔	✔	✔	✔
	`3DES`	✔	✔	✔	✔	✔
	`AES256`	✔	✔	✔	✗	✔
MAO	`mod`	✗	✗	✔	✔	✔
	`modMul`	✗	✗	✔	✔	✔
	`modExp`	✓	✓	✔	✔	✔
ECC	`ecAdd`	✓	✗	✔	✔	✗
	`ecMul`	✓	✗	✔	✔	✗
	`Pair`	✗	✗	✗	✗	✗

Note: ✔– algorithm is supported, ✓– algorithm is supported through a special API (e.g. NXP JCOP) or another function (e.g. RSA encryption), ✗– algorithm is not supported, MAO – Modular Arithmetic Operations, ECC – Elliptic Curve Cryptography.

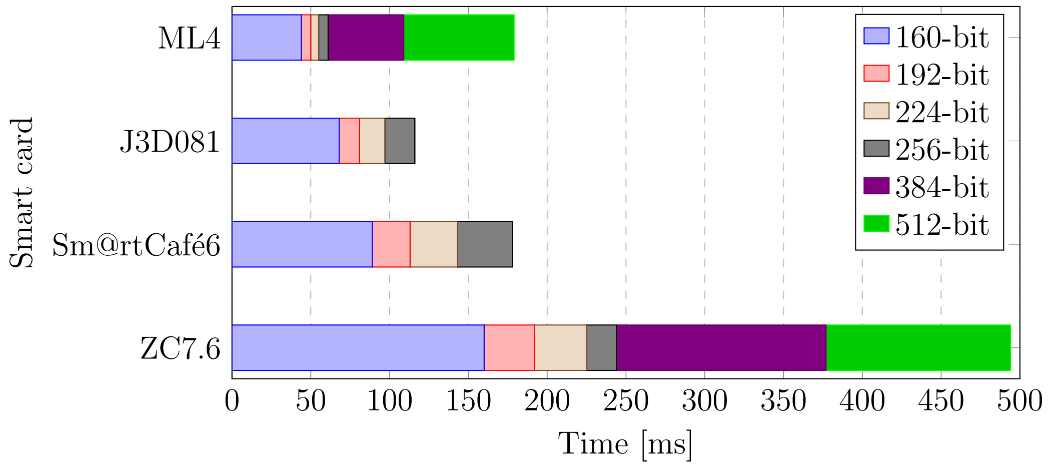


Figure 4: Efficiency of `ecMul` operation on different smart card platforms.

Furthermore, we also provided benchmarks of the employed cryptographic algorithms. SHA-1 algorithm is used in order to create non-interactive proof of knowledge (signing part) and as a part of key derivation function KDF for key establishment (encryption part). We use 3DES encryption algorithm to provide data confidentiality. The reason of this choice is a missing support of more secure AES algorithms on MultOS cards. Figure 5 shows the speed of SHA-1 and 3DES algorithms across platforms. The Java Card reports a bit better results than MultOS cards. However, we can assume that our data will not exceed 200 B, and therefore the difference between SCs is minimal (with the exception of the ML3 card, which reports much worse results in encryption), i.e. around 20 ms for SHA-1 and 40 ms for 3DES.

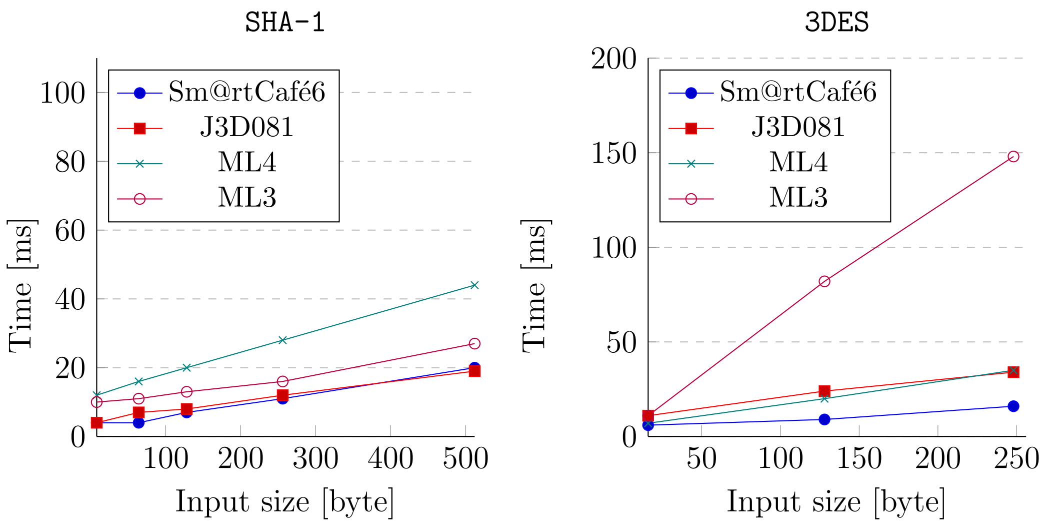


Figure 5: Message digest based on hash function SHA-1 and 3DES encryption on different smart card platforms.

9.2 ARM platform and software selection

ARM processors are widely used in smartphone, tablet, smartwatch and other IoT mobile devices. Raspberry Pi is an ARM-based single-board computer that runs Linux and has various communication interfaces, e.g. GPIO (General Purpose Input/Output) pins, Ethernet, HDMI, USB ports and bluetooth and WIFI adapters. These features allows Raspberry Pi to be part of many services in the IoT ecosystems. The technical specification of tested Raspberry Pi devices is shown in Table 4.

In public repositories, e.g. GitHub, there are several libraries with pairing-based cryptography support. The choice of the cryptographic library is crucial during the application development on resource-constrained devices. Since we are interested on the best performance, and therefore, the fastest pairing calculation, we focused on libraries implemented in C/C++ programming language. The selected libraries (Pairing Based Cryptography (PBC) [23], Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL) [26], University of Tsukuba Elliptic Curve and Pairing Library

Table 4: Technical specification of tested Raspberry Pi devices.

	ISA	**CPU**	**SDRAM**	**OS (32-bit)**
Raspberry Pi Model B+	ARMv6Z (32-bit)	1×ARM1176JZF-S 700 MHz	512 MB	Raspbian 9.3
Raspberry Pi Zero W	ARMv6Z (32-bit)	1 ARM1176JZF-S 1 GHz	512 MB	Raspbian 9.3
Raspberry Pi 3 Model B+	ARMv8-A (64/32-bit)	4 Cortex-A53 1.4 GHz	1 GB	Raspbian 9.3
Raspberry Pi 4 Model B	ARMv8-A (64/32-bit)	4 Cortex-A72 1.5 GHz	2 GB	Raspbian 9.3

Note: ISA – Instruction Set Architecture, CPU – central processing unit, SDRAM – Synchronous Dynamic Random Access Memory, OS – operating system, N/A – Not Available

(TEPLA) [17], Efficient LIbrary for Cryptography (RELIC) [1] and MCL [33]) were installed on an embedded device, i.e. ARM-based microcomputer (Raspberry Pi 3 Model B). The benchmarks were run by using the 256-bit BN (paring-friendly) curve and averaged over 10-run. The results are presented in Figure 6. We choose the MCL library, since it has support for ARM architecture (32 and also 64-bit version) and has the best computational speed results among the compared libraries.

Furthermore, Table 5 shows the comparison of the most time consuming operations for our protocol which are performed on the tested ARM devices.

Table 5: Computational capability of tested ARM devices for most demanding operations of our scheme.

	ecMul G1 [ms]	**ecMul G2** [ms]	**expGT** [ms]	**Pairing** [ms]
Raspberry Pi Model B+	6,2	12,9	19,0	38,7
Raspberry Pi Zero W	4,2	8,9	13,1	26,4
Raspberry Pi 3 Model B+	2,2	5,1	7,6	11,9
Raspberry Pi 4 Model B	0,8	1,6	2,5	5,1

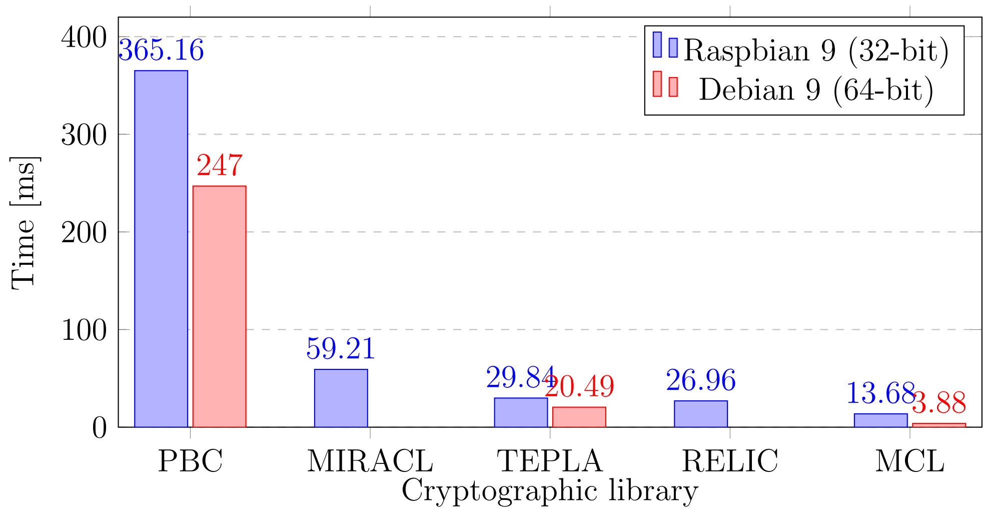


Figure 6: The comparison of different cryptographic libraries from the point of view of bilinear pairing performance over BN 256-bit elliptic curve on the ARMv8 processor (the Raspberry Pi 3, 32-bit and 64-bit OS).

9.3 Testing scenario and system parameters

In our testing scenario, receivers are represented by Raspberry Pi devices, and senders by SCs or Raspberry Pi devices. Normally, senders are represented by very resource-restricted devices (i.e., with processing and memory restrictions). For instance, a sender can be an user who owns a smartphone, a smart meter, an on-board unit built in cars (each of this device can by represented by Raspberry Pi which is using same ARM processors) or an access card (which is a SC). Accordingly, we choose a smart card platform that follows these constrained assumptions. Furthermore, the SC is a tamper-resistant device which securely allows the storage and the process of sensitive data such as cryptographic keys. In case of SC application development, we use only standard MultOS API and free public development environment (Eclipse IDE for C/C++ Developers, SmartDeck 3.0.1, MUtil 2.8). The application is written in MULTOS assembly code and C language.

Conversely, receivers can be a server, a PC or an embedded device which are less constrained and, therefore, they can be represented by a more powerful device. Tested SCs and Raspberry Pi hardware and software specifications are depicted in Tables 2 and 4, respectively. The Raspberries run Raspbian 9.0.3 operating system and C/C++ application. The application provides the communication with sender's smart card through PC/SC (Personal Computer/Smart Card) interface and executes `Unsigncrypt` (and `Signcrypt`) protocols. We use OpenSSL 1.1.1c library to perform cryptographic operations (i.e., hash and cipher), and MCL [33] library to perform operations over elliptic curves (i.e., EC point addition, EC scalar multiplication and bilinear pairing). The application for Raspberry Pi was developed in NetBeans IDE 8.2 development environment. The code was remotely built and executed on the targeted devices, i.e. Raspberry Pi B+/ZeroW/3B+/4B.

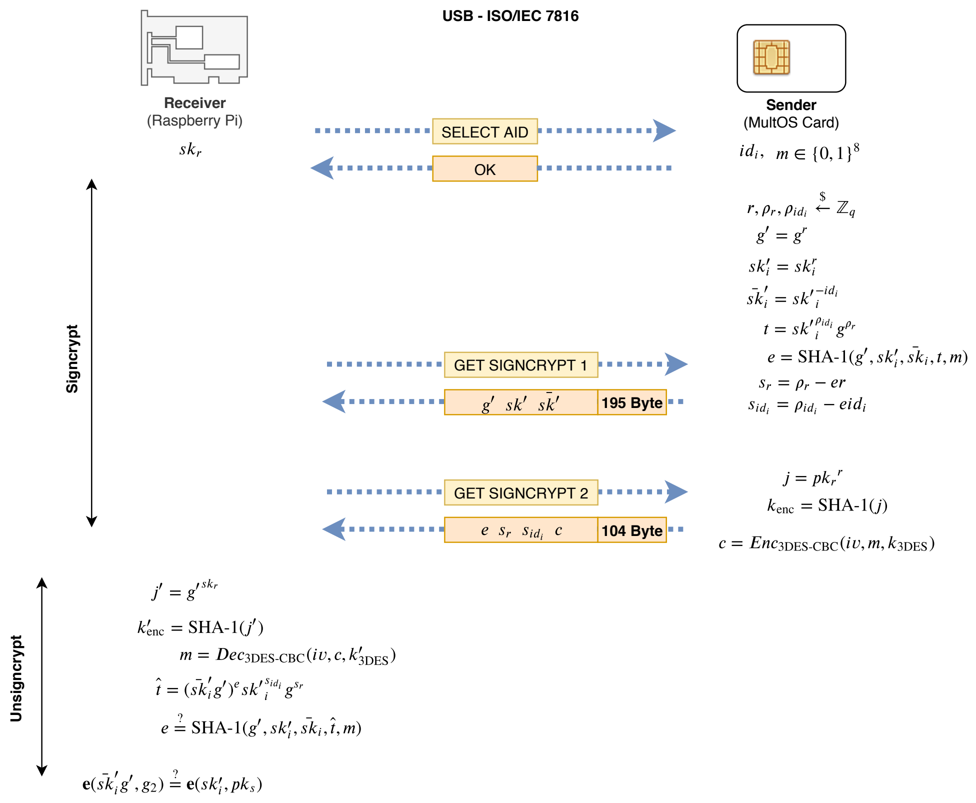
The signcryption scheme implementation follows the restrictions of current smart cards (see Table 3), and the most recent security requirements defined by National Institute of Standards and Technology (NIST), see [2] and [3] for more details. The security level of our implementation is 112 bits. This restriction is due to the use of 3DES cipher algorithm since the more secure AES-128 algorithm is not supported by our MultOS smart card. However, by replacing 3DES with AES-128 algorithm directly increase the scheme security to 128 bit, since our signcryption scheme already uses 256-bit elliptic curves with embedding degree 12 (i.e. Barreto–Naehrig EC) and SHA-1 hash algorithm. Table 6 shows the system parameters set in details.

Table 6: Cryptographic algorithms and elliptic curve domain parameters.

	Algorithm	**Size [Byte]**	**Description**
C	3DES-CBC	8 (block)	Data encryption
R		24 (Key)	
Y	SHA-1	20 (output)	KDF and FS
P	BN-curve	32	SPK
T			
	Parameter	**Size [Bites]**	**Hexadecimal Value**
E	p (characteristic)	254/[256]*	0x2523648240000001ba344d8000000008
C			6121000000000013a700000000000013
	a (constant)	0/[256]*	0x00000000000000000000000000000000
D			00000000000000000000000000000000
O	b (constant)	2/[256]*	0x00000000000000000000000000000000
M			00000000000000000000000000000002
A	G[x,y] (generator)	255/[513]*	0x042523648240000001BA344D8000000008
I			6121000000000013A700000000000012
N			00000000000000000000000000000000
			00000000000000000000000000000001
P	q (order of G)	254/[256]*	0x2523648240000001ba344d8000000007
A			ff9f800000000010a10000000000000d
R	h (cofactor)	1/[8]*	0x01

Note: KDF – Key Derivation Function, FS – Fiat–Shamir heuristic, SPK – Signature Proof of Knowledge, [Size]* – real allocated space on smart card.

Our implementation considers only single-receiver scenario with 8-byte message, where MultOS card acts as a sender and Raspberry Pi acts as both a sender and a receiver. A sketch of our implementation with involved smart card is depicted in Figure 7. MultOS ML4 smart card supports only T=0 transport protocol. Since we need to transfer 299 bytes in total and T=0 protocol allows us to transfer data payload of maximum 255 bytes, we need to use two APDU commands (`GET SIGNCRYPT 1` and `GET SIGNCRYPT 2`). While `GET SIGNCRYPT 1` performs group signature generation, `GET SIGNCRYPT 2` derives encryption key and encrypts data.



USB - ISO/IEC 7816
Receiver
(Raspberry Pi)
sk_r
Sender
(MultOS Card)
$id_i, \;\; m \in \{0,1\}^8$
SELECT AID
OK
$r, \rho_r, \rho_{id_i} \overset{\$}{\leftarrow} \mathbb{Z}_q$
$g' = g^r$
$sk'_i = sk_i^r$
$\bar{sk}'_i = {sk'}_i^{-id_i}$
$t = {sk'}_i^{\rho_{id_i}} g^{\rho_r}$
$e = \text{SHA-1}(g', sk'_i, \bar{sk}_i, t, m)$
$s_r = \rho_r - er$
$s_{id_i} = \rho_{id_i} - eid_i$
Signcrypt
GET SIGNCRYPT 1
$g' \;\; sk' \;\; \bar{sk}'$
195 Byte
GET SIGNCRYPT 2
$e \;\; s_r \;\; s_{id_i} \;\; c$
104 Byte
$j = {pk_r}^r$
$k_{\text{enc}} = \text{SHA-1}(j)$
$c = Enc_{\text{3DES-CBC}}(iv, m, k_{\text{3DES}})$
Unsigncrypt
$j' = {g'}^{sk_r}$
$k'_{\text{enc}} = \text{SHA-1}(j')$
$m = Dec_{\text{3DES-CBC}}(iv, c, k'_{\text{3DES}})$
$\hat{t} = (\bar{sk}'_i g')^e {sk'}_i^{s_{id_i}} g^{s_r}$
$e \overset{?}{=} \text{SHA-1}(g', sk'_i, \bar{sk}_i, \hat{t}, m)$
$\mathbf{e}(\bar{sk}'_i g', g_2) \overset{?}{=} \mathbf{e}(sk'_i, pk_s)$

Figure 7: Implementation of `Signcrypt` and `Unsigncrypt` algorithms.

Figures 8 and 9 show the final computational times for `Signcrypt` and `Unsigncrypt` algoritms performed on Raspberries and MultOS card. In case of Raspberries, the times are negligible and under 200 ms for both `Signcrypt` and `Unsigncrypt` protocols. In case of Raspberry Pi 4, the whole `Signcryption` process takes less than 60 ms (without communication overhead). Generally, SCs are much more slower to process `Signcrypt` algorithm with respect to Raspberries. However, in our implementation the SC is enough fast (under 1 s including communication overhead) to be used in a real scenario.

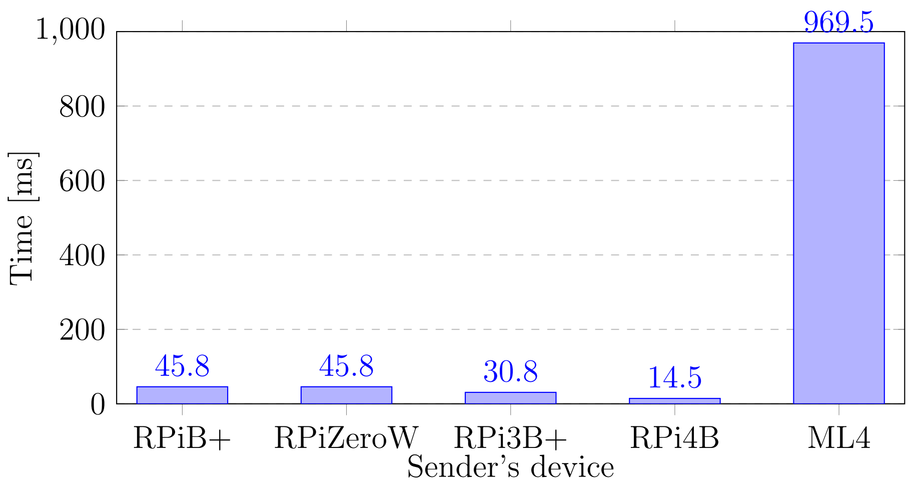


Figure 8: The performance comparison of `Signcrypt` algorithm performed on devices with different computing power.

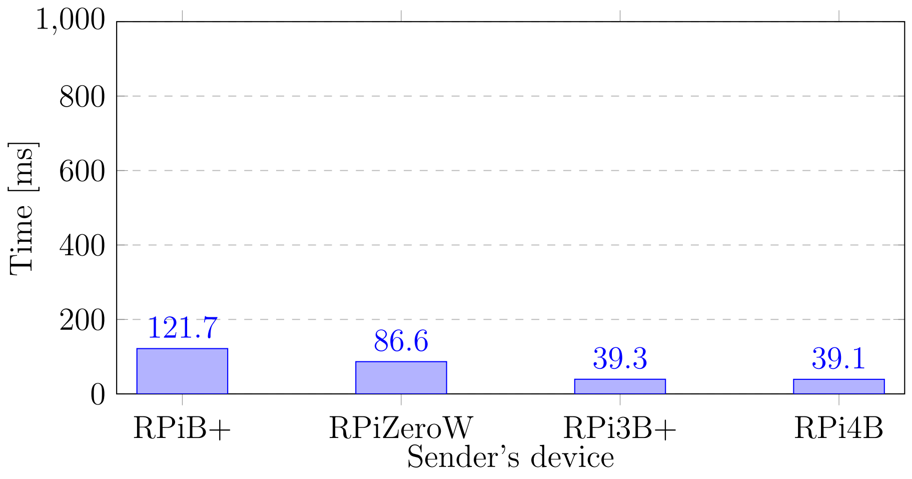


Figure 9: The performance comparison of `Unsigncrypt` algorithm performed on devices with different computing power.

In a multi-receiver scenario, additional $n\cdot$(`ecMul`+`SHA1`+`3DES`) operations are required, where n represents the number of added receivers to the system, `ecMul` the EC-256 scalar multiplication, `SHA1` the hash of 65 bytes (KDF from EC point $j = pk_r^r$), and `3DES` the encryption of 24 bytes (3DES master key encrypting message). Therefore, the time complexity on the sender's side grows linearly in n. Assuming that the cost of `ecMul` is 61 ms, `SHA1` is 19 ms and `3DES` is 15 ms, we can predict that the complexity of the protocol is

$969.5 + n \cdot 95$ ms. For instance, the time required to run `Signcrypt` algorithm is 1 919.5 ms in case of 10 receivers.

Moreover, the blacklist check requires $r \cdot ($`ecMul` operations, where r denotes the number of blacklisted users. Let Raspberry Pi 4 Model B be the receiver, then the blacklist check takes $r \cdot 0,8$ ms. For instance, it requires only 800 ms for 100 revoked users in the worst case.

10 Conclusions

In this article, we presented a new privacy-enhancing signcryption scheme which provides as main privacy-enhancing features: ciphertext and sender anonymity, traceability and unlinkability. To the best of our knowledge, this is the only signcryption scheme that holds both sender anoninymity, i.e. the sender's identity is hidden not only to an outsider but also to the receiver, and ciphertext anonymity, i.e. sender and receiver identity is hidden to an outsider. This is achieved by using our group signature scheme combined with an elliptic curve integrated encryption scheme.

Moreover, the security analysis of the scheme is also provided. Our proposal is proven to be strongly existentially unforgeable under an adaptive chosen message attack, indistinguishable under adaptive chosen ciphertext attack, and to provide ciphertext anonymity under adaptive chosen ciphertext attack. The used signature has also anonymity, traceability and unlinkability privacy features. At last, the scheme can be slightly modified in order to work in a multi-receiver scenario.

Our scheme has a wide range of possible usage in a smart cities, where the increasing connectivity can expose devices and, therefore, citiziens to cybersecurity attacks. This attack can be held not only by an outsider but also by a receiver. Therefore, users identity needs to be hidden even to the receiver of the message. For instance, we exploited the vehicles sharing uses case. Our proposal guarantees an encrypted communication to securely exchange data as vehicle location, keys to unlock the vehicle, and payment details of the user. Moreover, the user's privacy is guaranteed due to the group signature usage.

The experimental results show that our scheme is efficient even on computationally restricted devices and can be therefore used in many IoT applications. `Signcrypt` protocol on SCs takes less then 1 s (including communication overhead). `Unsigncrypt` protocol complexity time on current ARM devices is negligible (less than 40 ms).

References

[1] Aranha, D. F., Gouvêa, C. P. L.: RELIC is an Efficient LIbrary for Cryptography. `https://github.com/relic-toolkit/relic` (2018).

[2] Barker, E., Barker, W., Burr, W., Polk, W., Smid, M.: Recommendation for key management part 1: General (revision 4). NIST special publication part 1: general revision; pages 800 (2016).

[3] Barker, E., Chen, L., Roginsky, A. and Smid, M.: NIST Special Publication 800-56A Revision 2: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography. NIST special publication (2016).

[4] Bernardini, C., Asghar, M.R., Crispo, B.: Security and privacy in vehicular communications: Challenges and opportunities. In Vehicular Communications;10:13-28 (2017).

[5] Boneh, D., Boyen, X.: Short signatures without random oracles and the SDH assumption in bilinear groups. Journal of Cryptology; 21(2):149-77 (2008).

[6] Braeken, A., Touhafi, A.: AAA-autonomous anonymous user authentication and its application in V2G. Concurrency and Computation: Practice and Experience. 30(12):e4303 (2018).

[7] Camenisch, J., Stadler, M.: Efficient group signature schemes for large groups. In Kaliski, B., editor, Advances in Cryptology - CRYPTO '97, volume 1294 of Lecture Notes in Computer Science, pages 410–424. Springer Berlin / Heidelberg (1997).

[8] Camenisch, J., Drijvers, M., Hajny, J.: Scalable revocation scheme for anonymous credentials based on n-times unlinkable proofs. In Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society (pp. 123-133). ACM (2016).

[9] Chaudhari, P., Das, M.L.: Privacy Preserving Signcryption Scheme. In International Conference on Distributed Computing and Internet Technology(pp. 196-209). Springer, Cham (2017).

[10] Coventry, L., Branley, D.: Cybersecurity in healthcare: A narrative review of trends, threats and ways forward. Maturitas; 113:48-52 (2018).

[11] Dorri, A., Steger, M., Kanhere, S.S., Jurdak R.: Blockchain: A distributed solution to automotive security and privacy. IEEE Communications Magazine; 55(12):119-25 (2017).

[12] Efthymiou, C., Kalogridis, G.: Smart grid privacy via anonymization of smart metering data. In Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference (pp. 238-243). IEEE (2010).

[13] Gayoso Martínez, V., Hernández Encinas, L., Sánchez Ávila, C.: A survey of the elliptic curve integrated encryption scheme (2010).

[14] Greveler, U., Justus, B., Loehr, D.: Forensic content detection through power consumption. In Proceedings of the IEEE International Conference on Communications, Canada. IEEE Press, Psicataway, NJ, 6759–6763, (2012).

[15] Hajny, J., Dzurenda, P., Malina, L., Ricci, S.: Anonymous Data Collection Scheme from Short Group Signatures. In SECRYPT 2018 Proceedings: p. 1-10. ISBN: 978-989-758-319-3 (2018).

[16] Huang, X., Susilo, W., Mu, Y., Zhang, F.: Identity-based ring signcryption schemes: cryptographic primitives for preserving privacy and authenticity in the ubiquitous world. In Advanced Information Networking and Applications, 19th International Conference on 2005 Mar 28 (Vol. 2, pp. 649-654). IEEE (2005).

[17] Kanaoka, A.: TEPLA Elliptic Curve and Pairing Library. University of Tsukuba, `https://github.com/TEPLA/tepla-library` (2018).

[18] Lal, S., Kushwah, P.: Anonymous ID Based Signcryption Scheme for Multiple Receivers. IACR Cryptology ePrint Archive; 345 (2009).

[19] Li, F., Han, Y., Jin, C.: Cost-effective and anonymous access control for wireless body area networks. IEEE Systems Journal; 12(1):747-58 (2018).

[20] Li, H., Pang, L.: Cryptanalysis of Wang et al.'s improved anonymous multi-receiver identity-based encryption scheme. IET Information Security; 8(1):8-11 (2014).

[21] Li, F., Zheng, Z., Jin, C.: Secure and efficient data transmission in the Internet of Things. Telecommunication Systems; 62(1):111-22 (2016).

[22] Libert, B., Quisquater, J.J.: Efficient signcryption with key privacy from gap Diffie-Hellman groups. In International Workshop on Public Key Cryptography (pp. 187-200). Springer, Berlin, Heidelberg (2004).

[23] Lynn, B.: The pairing-based cryptography (PBC) library. `https://crypto.stanford.edu/pbc/` (2018).

[24] Midgley, P.: Bicycle-sharing schemes: enhancing sustainable mobility in urban areas. United Nations, Department of Economic and Social Affairs; 8:1-2 (2011).

[25] Miller, C., Valasek, C.: A survey of remote automotive attack surfaces. black hat USA; 2014:94 (2014).

[26] MIRACL UK Ltd: Multiprecision Integer and Rational Arithmetic Cryptographic Library – the MIRACL Crypto SDK. `https://github.com/miracl/MIRACL` (2018).

[27] Mohanty, S., Majhi, B., Das, S.: A secure electronic cash based on a certificateless group signcryption scheme. Mathematical and Computer Modelling, 58(1-2):186-95 (2013).

[28] Pang, L., Gao, L., Li, H., Wang, Y.: Anonymous multi-receiver ID-based signcryption scheme. IET Information Security, 9(3):194-201 (2015).

[29] Pang, L., Li, H.: nMIBAS: a novel multi-receiver ID-based anonymous signcryption with decryption fairness. Computing and Informatics. 32(3):441-60 (2013).

[30] Pontes, T., Vasconcelos, M., Almeida, J., Kumaraguru, P., Almeida, V.: We know where you live: privacy characterization of foursquare behavior. In Proceedings of the 2012 ACM conference on ubiquitous computing 2012 Sep 5 (pp. 898-905). ACM (2012).

[31] Popescul, D., Radu, L.D.: Data security in smart cities: challenges and solutions. Informatica Economica; 20(1):29 (2016).

[32] Saraswat, V., Sahu, R.A., Awasthi, A.K.: A secure anonymous proxy signcryption scheme. Journal of Mathematical Cryptology, 11(2):63-84 (2017).

[33] Shigeo, M.: Mcl library. `https://github.com/herumi/mcl` (2018).

[34] Smart, N.P.: The exact security of ECIES in the generic group model. In IMA International Conference on Cryptography and Coding, pp. 73-84. Springer, Berlin, Heidelberg, (2001).

[35] Shoup, V.: Lower bounds for discrete logarithms and related problems. In International Conference on the Theory and Applications of Cryptographic Techniques; pp. 256-266. Springer, Berlin, Heidelberg (1997).

[36] Wang, H., Zhang, Y., Xiong, H., Qin, B.: Cryptanalysis and improvements of an anonymous multi-receiver identity-based encryption scheme. IET Information Security; 6(1):20-7 (2012).

[37] Weikl, S., Bogenberger, K.: Relocation strategies and algorithms for free-floating car sharing systems. IEEE Intelligent Transportation Systems Magazine; 5(4):100-11 (2013).

[38] Yung, M.: Practical signcryption. Springer Science & Business Media (2010).