Generic Construction of Certificateless Signcryption Scheme

Jayaprakash Kar[1] and Sagar Naik[2]

[1] Information Security Research Group,Department of Information Systems
Faculty of Computing & Information Technology
King Abdulaziz University, Kingdom of Saudi Arabia, Jeddah-21589 ⋆
[2] Department of Electrical and Computer Engineering
University of Waterloo, Waterloo, Ontario, Canada, N2L3G1

Abstract. Confidentiality and message authentication are the most important security goals that can be achieved simultaneously by Signcryption scheme. It is a cryptographic technique that performs both the functions of digital signature and public key encryption in a single logical step significantly at a lower cost than that of conventional method of signature-then-encryption. The paper proposes an efficient Certificateless Signcryption Scheme(`CLSC`) in random oracle model on bilinear mapping. It is provably secure under the assumptions of intractability of k-CAA, Inv-CDH, q-BDHI and CDH problems.

Keywords:Provable Security, Public Key Infrastructure, Insider Security, Chosen Message Attack

1 Introduction

Signcryption is a cryptographic primitive designed to operate both the function encryption and signing in a one single logical step significantly at a lower cost than that of conventional approaches. Hence, it provides both the security goals confidentiality and authentication simultaneously. In order to authenticate the user's public keys in public key cryptography, public key infrastructure (PKI) and identity based cryptography (IBC) are applied. Issuing of the certificate and managing can be carried out by setting a hierarchical framework called public key infrastructure (PKI).

In the PKI, a trusted third party called a certificate authority (CA) issues a certificate that provides an unforgeable and trusted link between the public key and the identity of a user by the signature of the CA. However certificate management includes storage, revocation, distribution of certificates is complex in traditional PKI. Further, the validity of the certificate is verified prior to use them. Hence there is a key management problem in PKI. This is resolved by identity-based cryptography (IBC) that was introduced by Shamir [2] in 1984. At IBC, the public key of the users is in the form of binary string that can identify the user through the certificates. The binary strings may be the IP address, e-mail address, etc. Since 1984, many identities-based signature schemes (IBS) have been proposed [3, 4]. But the efficient identity-based encryption (IBE) was proposed by [23] in 2001 using bilinear pairing over super singular curves. Afterward, numerous identify-based signcryption scheme (IBSC) are proposed [6, 8–12]. The advantage of IBC is that it reduces the requirement of public key certificates with the help of a trusted third party known as a public key generator (PKG). The role of PKG is generated and issue the private key of all of its users so that only these users can decrypt the ciphertext that provides the implicit in certification. Hence, it reduces the space and time complexity. However, this leads the key escrow problem to the IBC. Also in IBS scheme, PKG might forge any user's signature participating in the protocol. Generally in all the traditional signcryption schemes, the user's public key is the pseudo random bit string to be chosen from a particular given set. So, the user's authorization can be achieved by signcryption scheme.

In order to solve this key escrow problem in IBC, a new paradigm is introduced by Al-Riyami and Paterson [13] which is known as certificateless cryptography which does not require the use of the certificate. However, it does not solve fully key escrow problem of IBC. In order to serve in between PKI and IBC, the public key cryptography is framed with certificateless setting. There exist a trusted third party known as a key generator center (KGC) is to be fixed and is not be allowed to access the user's private key in IBC. KGC takes the user's identity and master secret key as input and generates a partial private key. Then the user chooses a secret value, combines with the partial private key and computes full private key. Since the public key is no longer computable from the identity

⋆ Corresponding author: e-mail: `jayaprakashkar@yahoo.com` of the user, it is not identity-based technique. If a sender would send a message to the receiver in certificateless setting, she has to obtain the public key of the receiver. However it does not require the authentication of receiver's public key and no need of the certificates.

In this paper, we propose a provably secure certificateless signcryption schme in random oracle. Security of the scheme relies on k-CAA, Inv-CDH, q-BDHI and CDH problem. We prove that, the proposed scheme has the indistiguishability property against adaptive chosen ciphertext attack and existential unforgeable against chosen message attack under the defined security model.

The paper is organized as follows. Section-1 and 3 present introduction and related works on CLSC. In section-3, we define mathematical assumption and properties of the admissible bilinear map. The framework of the scheme and security model are described in section-4 and 5 respectively. The proposed scheme is presented in section-6 and finally we conclude in section-7.

2 Previous Works

The notion of certificateless signcryption(CLSC) was introduced by Barbosa and Farshim [21] in 2008. Al-Riyami and Paterson [13], Selvi et al. [14] and Xie et al. [25] proposed three different provably secure schemes individually in the random oracle model. In the random oracle model, it is assumed that, the hash function is substituted by a random function called random oracle. The random function is allowed to access publicly. As a result, in the random oracle model, the hash value cannot be computed by the adversary. Liu et al. [24] proposed a CLSC scheme in the standard model in 2009. In the standard model, the adversary gets a limited amount of time and computing power to access the system. The vulnerability of the scheme has proven by Selvi et al. [14]. Also, she has proven these schemes are not publicly verifiable and have forward security. The technique of tag-Key Encapsulation method signcryption scheme was proposed by Li et al. [26] using the certificateless setting. However, Selvi emph et al. [14] proved that the hybrid scheme is not secure against existentially unforgeable attack and proposed an improved version of the scheme. In an identity-based setting Malone-Lee constructed a signcryption scheme [18]. Boyen extended this work and formulated the security framework that achieved different security goals that could be constructed by an identity-based signcryption scheme [8]. Subsequently, by Libert et al. in [19] and Chen et al. in [9] proposed a secure and efficient signcryption scheme.

Numerous of CLSC [21] [24] have been proposed based on bilinear mapping and discuss the complexity in the computation of pairing operations. However the computation time and cost in the computation of pairing operations remains same and did not reduce. Selvi et al. [15] and Xie et al. [16] have proposed two CLS without pairing. This motivated to construct pairing free CLSC and proposed in [17]. Furthermore, in these schemes there are some modular exponent operation which results poor performance in computation due to high computational time.

Due to less computational cost and communication overhead, our scheme is most suited to implement on low-end constrained devices, such as wireless sensor network, smart phone, PDA etc. The proposed scheme is provably secure in the random oracle model. In many cryptographic applications such as secure board cast, mobile adhoc network, etc where both authentication and confidentiality is required at a time. Therefore Signcryption technique can be used in these application. Further, sicne our scheme is based on identity-based cryptography, no need to authenticate a public key . This results to reduce in computational cost.

3 Preliminaries

3.1 Bilinear Pairings and Complexity Assumptions

Bilinear pairing is a map between two groups. There are two form of bilinear pairings on elliptic curves known as Weil and Tate pairings. It is defined as:
Let $\mathbb{G}_1$ be a cyclic additive group of prime order q and $\mathbb{G}_2$ be a cyclic multiplicative group of the same prime order p. Let $\hat{e}$ be a bilinear map which is non-degenerated and computable.

$$\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$$

holds following

– **Bilinearity**: Let $a, b \in \mathbb{Z}_q^*$ and $P, Q \in \mathbb{G}_1$

   1. $\hat{e}(aP, bQ) = \hat{e}(P, Q)^{ab}$.
   2. $\hat{e}(P + Q, R) = \hat{e}(P, R)\hat{e}(Q, R)$, for $P, Q, R \in \mathbb{G}_1$.
- **Non-degenerate**: Generator of $\mathbb{G}_2$ is $\hat{e}(P, P)$, if the generator of $\mathbb{G}_1$ is P. $\hat{e}(P, Q) \neq 1_{\mathbb{G}_2}$ for $P, Q \in \mathbb{G}_1$.
- **Computability**: $\hat{e}(P, Q)$ can be compute efficiently for all $P, Q \in \mathbb{G}_1$.

$\hat{e}$ is the bilinear map and is considered as admissible.
We consider the pairing is the modified Tate pairing and weil pairing on super singular elliptic curve.

Definition 1. **k-Collusion Attack Algorithm Assumption(k-CAA)** *Let k be an integer, $s \xleftarrow{R} \mathbb{Z}_p^*$ and P be the generator of an additive group $<\mathbb{G}, +>$. k-CAA problem in the group $\mathbb{G}$ is defined as given*

$P, \beta P$ and k-pairs $H_1, (\beta{+}H_1)^{-1}P), (H_2, (\beta{+}H_2)^{-1}P) \ldots (H_k, (\beta{+}H_k)^{-1}P)$, it is computationally infeasible to compute pair $(H_1^*, (\beta + H_2^*)^{-1}P)$ for some $H^* \notin \{H_1, H_2 \ldots H_k\}$ and β.

Definition 2. **Inverse Computational Diffie-Hellman Problem(Inv-CDH)** *Inverse Computational Diffie-Hellman Problem(Inv-CDH) is given $P, \beta P$ is to compute $\frac{1}{\beta}P$, where $\beta \xleftarrow{R} \mathbb{Z}^*$ is an unknown quantity.*

Definition 3. **q-Strong Diffie-Hellman Problem(q-SDHP)** *Let $\mathbb{G}_1$ and $\mathbb{G}_2$ are two groups of same prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be the bilinear map. P be a generator of the group $\mathbb{G}_1$. q-Strong Diffie-Hellman Problem(q-SDHP)in $(\mathbb{G}_1, \mathbb{G}_2, e)$ is given $(P, Q\beta Q, \beta^2 Q \ldots \beta^q Q)$, $q + 2$ tuples as input, to compute $(c, \frac{1}{c+\beta}P)$, where $c \xleftarrow{R} \mathbb{Z}_q^*$ be an unknown quantity.*

Definition 4. **q-Bilinear Diffie-Hellman Inverse Problem(q-BDHIP)** *Let $\mathbb{G}_1$ and $\mathbb{G}_2$ are two groups of same prime order p and $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be the bilinear map. P be a generator of the group $\mathbb{G}_1$. q-Bilinear Diffie-Hellman Inverse Problem(q-BDHIP) in $(\mathbb{G}_1, \mathbb{G}_2, e)$ is given $(P, \beta P, \beta^2 P \ldots \beta^q P)$ is to compute $(P, P)^{\frac{1}{\beta}}$.*

4 Framework of Certificateless Signcryption(CLSC)

Certificateless Signcryption Scheme(CLSC) comprises following seven probabilistic polynomial time solvable algorithms.

- Setup: It is a global probabilistic polynomial time solvable algorithm run by KGC. Let 1^k be the security parameter is given as input. It outputs Msk as a KGC's master secret key and params as system parameters which consists of Mpk, $\mathbb{M}$, $\mathbb{C}$ and $\mathbb{R}$ as master public key, descriptions of message space, ciphertext space and randomness space respectively. Formally

$$(\texttt{params, Msk}) \leftarrow \texttt{Setup}(1^k)$$

- Extract-Partial-Private-Key: The algorithm constructs user 's partial private key S_{ID}. It takes the system parameters params, master secret key Msk , identity of the corresponding user ID $\in \{0, 1\}^*$. Formally we can write

$$S_{ID} \leftarrow \texttt{Extract-Partial-Private-Key(params, Msk, ID)}$$

- Generate-User-Keys: This algorithm generates a secret value μ and a public key PK_{ID}. It takes params and an identity ID as input. The secret value generated is used to construct the full private key by the following algorithm and the public key generated is published without certification. Formally we can write :

$$\mu \leftarrow \texttt{Generate-User-Keys(params, ID)}$$

- Set-Private-Key: The algorithm constructs the user's full private key d_{ID}. It take the two parameters user's partial private key S_{ID} and a secret value μ as input . Formally

$$\texttt{Set-Private-Key}(S_{ID}, \mu)$$

- CL-Signcrypt: This algorithm constructs the certificateless signcrypt text ciphertext $c \in \mathcal{C}$. It run taking the system parameter params, plaintext message $m \in \mathbb{M}$, the sender's full private key d_{ID_s}, user's identity ID_s and sender's public key PK_{ID_s} , and the receiver's identity ID_r and receiver's public key PK_{ID_r}.

$$c \leftarrow \texttt{CL-Signcrypt}(\texttt{params}, m, d_{ID_s}, ID_s, PK_{ID_s}, ID_r, PK_{ID_r})$$

- `CL-Unsigncrypt`: This algorithm returns the plaintext message m and symbol $\perp$ for failure if plaintext message is not valid. It takes the system parameter `params`, a ciphertext c, the sender's identity ID_s and public key PK_{ID_s}, and the receiver's full private key S_{ID_r}, identity ID_r and public key PK_{ID_r}. Formally

$$(\ m\ /\ \perp\ ) \leftarrow \texttt{CL-Unsigncrypt}(\texttt{params}, ID_s, PK_{ID_s}, S_{ID_r}, ID_r, PK_{ID_r})$$

5 Security Model

5.1 Security Discussions

The security of signcryption have two issues: the security goal that we want to achieve and the attack model where to evaluate the capabilities of the adversary. The notion of security was defined by Barbosa and Farshim [21]. Confidentiality and Unforgeability are the two most important security requirement for a CLSC scheme. In CLSC confentiality is defined by the model as indistinguishability against adaptive chosen ciphertext attacks (`IND-CCA2`) and Unforgeability is defined by the model as existential unforgeability against adaptive chosen messages attacks (`UF-CMA`).

Consider the strong notion of insider security. So the notion of strong existential unforgeability against adaptive chosen message attacks is denoted by (`sUF-CMA`). Here, the adversary wins the game, if it returns a valid message and signcryption (m, σ) provided the signcryption oracle does not returns the signcryption σ on the message m in before. Similar to the author proposed in [8, 9], the attacks targeting to signcryption does not consider for $ID_s = ID_r$. These type of queries are not not allowed for significant oracles and are not taken the signcryption as a valid forgery. Following two types of adversaries exists in the model.
Type-I and II attacker

- `Type-I`: The adversary constructs an attacker who is considered as a common user of the system and not in possession of the master secret key generated by `KGC`. But he can replace the public key of the users with valid public keys of his choice in an adaptive manner.
- `Type-II`: The adversary constructs an honest-but-curious `KGC` who knows the `KGC`'s master secret key. But he cannot able to replace public keys of the users.

`Game for Confidentiality`:
The security model is defined by the game play between the adversary and challenger. With respect to the two types of attacker, there are games "`IND-CCA2-I`" and "`IND-CCA2-II for confidentiality` . Let the adversary of both types are denoted by $\mathcal{A}_I$ and $\mathcal{A}_{II}$ respectively. During the game they interact with their "challenger" and maintain a record to store all history of "query-answer".

`IND-CCA2-I:`
`Initial:` In initialization stage, the challenger generates master secret key msk and the system parameters by running the algorithm `Setup` .

$$(params, msk) \leftarrow \texttt{Setup}(1^k)$$

The challenger provides the system parameter $params$ to $\mathcal{A}_I$ and keeps secret the master secret key msk . In `Phase 1:`, the adversary $\mathcal{A}_I$ submits polynomials bounded number of queries adaptively.

- `Extract partial private key:` In the extraction of partial private key, an user's identity ID is chosen by the adversary $\mathcal{A}_I$ transmits to the challenger. The challenger runs the algorithms `Extract-Partial-Private-Key` and computes the partial private key D_{ID} of the corresponding user.

$$D_{ID} \leftarrow \texttt{Extract-Partial-Private-Key}(params, msk, ID)$$

  Then sends D_{ID} to $\mathcal{A}_I$.
- `Extract-Secret-value` : The attacker $\mathcal{A}_I$ chooses an identity ID and sends to the challenger. Then the challenger first run the `Extract-Partial-Private-Key` algorithm taking the identity and system parameter as input and constructs the secret value and the public key of the corresponding user. Formally, we can write as

$$(x_{ID}, PK_{ID}) \leftarrow \texttt{Generate-User-Keys}(params, ID).$$

Then he provides this secret value x_{ID} to $\mathcal{A}_I$. The previous queries are stored by the challenger. $\mathcal{A}_I$ provides the corresponding secret value x_{ID} to the user with condition that his public key is being replaced and $\mathcal{A}_I$ is not allowed to query the challenge receiver's partial private key. This constraint is imposed , since it is difficult to expect that the challenger is able to provide a full private key for a user for which it does not know the secret value.

– `Request public key:` The adversary $\mathcal{A}_I$ chooses an identity ID. The challenger computes $(x_{ID}, PK_{ID}) \leftarrow$`Generate-User-Keys`$(params, ID)$ and provides the public key PK_{ID} of the corresponding user to $\mathcal{A}_I$ and store a record for all.
– `Replace public key:` $\mathcal{A}_I$ chooses randomly a valid public key PK for t he corresponding user and replaces a public key PK_{ID} with the chosen PK.
– `Signcryption queries:` $\mathcal{A}_I$ takes the plaintext message m, a sender's identity ID_s and a receiver's identity ID_r, the challenger recovers from his "query-answer" list for full private key S_{ID_s} and public key PK_{ID_s} of the sender and public key PK_{ID_r} of receiver. Then computes the signcryption as

$$\sigma \leftarrow \texttt{Signcrypt}(params, m, S_{ID_s}, ID_s, PK_{ID_s}, ID_r, PK_{ID_r}),$$

and returns σ to $\mathcal{A}_I$. All the queries are being stored. If the sender's public key has been replaced, then $\mathcal{A}_I$ provides the secret value of the sender to the challenger. Here the queries for $ID_s = ID_r$ is not allowed.
– `Unsigncryption queries:` $\mathcal{A}_I$ takes the signcrytion σ, a sender's identity ID_s and a receiver's identity ID_r, the challenger recovers S_{ID_r} from its "query-answer" list, computes

$$\texttt{Unsigncrypt}(params, \sigma, ID_s, PK_{ID_s}, S_{ID_r}, ID_r, PK_{ID_r}),$$

and returns the result to $\mathcal{A}_I$. The output is either a plaintext message m or $\perp$ if the verification does not hold. We can note that, when the public key of the receiver is replaced, the challenger is not conscious about the secret value of the receiver. In this circumstance, the secret value will not be provided by $\mathcal{A}_I$. So the queries for $ID_s = ID_r$ is not allowed.

`Challenge:` When Phase-I is completed, $\mathcal{A}_I$ would like to challenge for two equal length plaintexts (m_0, m_1), a sender's identity ID_s^*, and a receiver's identity ID_r^*. He generates these challenge parameters. We note that, $\mathcal{A}_I$ will be disallowed to query to extract partial private key for ID_r^*. Also that ID_r^* cannot be equal to an identity for which both the public key has been replaced and the partial private key has been extracted. The challenger picks a random bit δ from $\{0,1\}$, computes

$$\sigma^* \leftarrow \texttt{Signcrypt}(params, m_\delta, S_{ID_s^*}, ID_s^*, PK_{ID_s^*}, ID_r^*, PK_{ID_r^*}),$$

and returns σ^* to $\mathcal{A}_I$. `Phase 2:` The adversary $\mathcal{A}_I$ can ask a polynomially bounded number of queries adaptively again as in Phase -1. The restriction is applied here: $\mathcal{A}_I$ cannot extract the private key for ID_r^*. $\mathcal{A}_I$ cannot extract the partial private key for ID_r^* if the public key of this identity has been replaced before the challenge phase. In addition, $\mathcal{A}_I$ cannot make an unsigncryption query on σ^* under ID_s^* and ID_r^*, unless the public key $PK_{ID_s^*}$ or $PK_{ID_r^*}$ has been replaced after the challenge phase.

`Guess:` $\mathcal{A}_I$ produces a bit δ' and wins the game if $\delta' = \delta$. The advantage of $\mathcal{A}_I$ is defined to be

$$\mathrm{Adv}_{\texttt{CLSC}}^{\texttt{IND-CCA2-I}}(\mathcal{A}_I) = |2\Pr[\delta' = \delta] - 1|,$$

where $\Pr[\delta' = \delta]$ denotes the probability that $\delta' = \delta$.

`IND-CCA2-II:`
This is the game where the type-II attacker $\mathcal{A}_{II}$ interacts with the "challenger": `Initial:` The challenger runs the `Setup` algorithm and generates the system parameters and master secret key as

$$(params, msk) \leftarrow \texttt{Setup}(1^k) \text{ and gives both } params \text{ and } msk \text{ to } \mathcal{A}_{II}.$$

`Phase 1:` The adversary $\mathcal{A}_{II}$ can perform adaptively polynomial bounded number of queries. Since $\mathcal{A}_I$ can computes partial private key by using msk, it does not require the partial-private key extraction query . The queries `Extract private key` , `Request public key`, `Signcryption` and `Unsigncryption` as in game `IND-CCA2-I` are same in `IND-CCA2-II` . `Challenge :` When phase-I is completed, $\mathcal{A}_{II}$ chooses two plaintexts messages (m_0, m_1) of equal length, a sender's identity ID_s^*, and a receiver's identity ID_r^* on which he would like to challenge. We note that, $\mathcal{A}_{II}$ is not allowed to extract the secret value of ID_r^* and replace his public key. However, he quires the full private key for the receiver ID_r^* .

The challenger picks a random bit δ from $\{0, 1\}$, computes

$$\sigma^* \leftarrow \texttt{Signcrypt}(params, m_\delta, S_{ID_s^*}, ID_s^*, PK_{ID_s^*}, ID_r^*, PK_{ID_r^*}),$$

and returns σ^* to $\mathcal{A}_{II}$. `Phase 2:` The adversary $\mathcal{A}_{II}$ can ask a polynomial bounded number of queries adaptively again as in Phase-1. $\mathcal{A}_{II}$ cannot extract the private key for ID_r^*. In addition, $\mathcal{A}_{II}$ cannot make an unsigncryption query on σ^* under ID_s^* and ID_r^*, unless the public key $PK_{ID_s^*}$ or $PK_{ID_r^*}$ has been replaced after the challenge phase. `Guess:` $\mathcal{A}_{II}$ produces a bit δ' and wins the game if $\delta' = \delta$. The advantage of $\mathcal{A}_{II}$ is defined to be

$$\text{Adv}_{\texttt{CLSC}}^{\texttt{IND-CCA2-II}}(\mathcal{A}_{II}) = |2\Pr[\delta' = \delta] - 1|,$$

where $\Pr[\delta' = \delta]$ denotes the probability that $\delta' = \delta$.

Definition 5. *A CLSC scheme is said to be* `IND-CCA2-I` *secure (resp.*`IND-CCA2-II` *secure) if there is no probabilistic polynomial time (PPT) adversary $\mathcal{A}_I$ (resp. $\mathcal{A}_{II}$) which wins* `IND-CCA2-I` *(resp.* `IND-CCA2-II`*) with non-negligible advantage. A CLSC scheme is said to be* `IND-CCA2` *secure if it is both* `IND-CCA2-I` *secure and* `IND-CCA2-II` *secure.*

`Game for Unforgeability`

This is the strong existential unforgeability, we describes two games "`sUF-CMA-I`" and "`sUF-CMA-II`". Consider two adversaries of type-I and type-II as $\mathcal{F}_I$ and $\mathcal{F}_I I$. They interacts with their corresponding challengers. The challenges maintains a records contains the history of "query-answer". `sUF-CMA-I:` `Initial:` The challenger runs the `Setup` algorithm and obtains the system parameter and master secret key. The master secret key is to be keeping secret with him and send the system parameter to the attacker. It is given by

$$(params, msk) \leftarrow \texttt{Setup}(1^k) \text{ and gives } params \text{ to } \mathcal{F}_I.$$

The challenger keeps master secret key msk and sends $params$ to $\mathcal{F}_I$.

`Attack:` The adversary $\mathcal{F}_I$ performs a polynomial bounded number of queries in an adaptive manner. `Forgery:` $\mathcal{F}_I$ generates $(m^*, \sigma^*, ID_s^*, ID_r^*)$. We note that, ID_s^* was not queried to extract partial private key but $\mathcal{F}_I$ can query to extract full-private key of ID_r^*. Further, ID_s^* should not be equal to an identity for which both the public key has been replaced and the partial private key has been extracted. In the meantime σ^* was not returned by the signcryption oracle on the input (m^*, ID_s^*, ID_r^*) during Attack stage. $\mathcal{F}_I$ wins the game if the result of

$$\texttt{Unsigncrypt}(params, \sigma^*, ID_s^*, PK_{ID_s^*}, S_{ID_r^*}, ID_r^*, PK_{ID_r^*})$$

is not the the symbol $\perp$. Particularly σ^* is a valid ciphertext of m^* in ID_s^* and ID_r^* as challenge sender and receiver respectively. The advantage of wining of $\mathcal{F}_I$ is defined by the probability given by

$$Pr[succ] = Adv_{\texttt{CLSC}}^{\texttt{sUF-CMA-I}}(\mathcal{F}_I)$$

`sUF-CMA-II:`

This is the game in which $\mathcal{F}_{II}$ interacts with the "challenger": `Initial:` The challenger runs $(params, msk) \leftarrow$ `Setup`(1^k) and gives both $params$ and msk to $\mathcal{F}_{II}$. `Attack:` The adversary $\mathcal{F}_{II}$ performs a polynomial bounded number of queries just like in the `IND-CCA2-II` game. `Forgery:` $\mathcal{F}_{II}$ produces a quaternion $(m^*, \sigma^*, ID_s^*, ID_r^*)$. ID_s^* should not be queried to extract a private key. In addition, σ^* was not returned by the signcryption oracle on the input (m^*, ID_s^*, ID_r^*) during Attack stage. $\mathcal{F}_{II}$ wins the game if the result of

$$\texttt{Unsigncrypt}(params, \sigma^*, ID_s^*, PK_{ID_s^*}, S_{ID_r^*}, ID_r^*, PK_{ID_r^*})$$

is not the $\perp$ symbol. The advantage of $\mathcal{F}_{II}$ is defined as the probability that it wins.

Definition 6. *A CLSC scheme is said to be sUF-CMA-I secure (resp. sUF-CMA-II secure) if there is no PPT adversary $\mathcal{F}_I$ (resp. $\mathcal{F}_{II}$) which wins* `sUF-CMA-I` *(resp.* `sUF-CMA-II`*) with non-negligible advantage. A CLSC scheme is said to be sUF-CMA secure if it is both sUF-CMA-I secure and sUF-CMA-II secure.*

6 Proposed Certificateless Signcryption Scheme(CLSC)

6.1 Construction

The proposed CLSC is defined be the following seven PPT algorithms.

– Setup: given security parameter k, KGC chooses bilinear map groups $(\mathbb{G}_1, \mathbb{G}_2)$ of same prime order $p > 2^k$ and generators $P \in \mathbb{G}_1$, $g = e(P, P)$, where $g \in \mathbb{G}_2$. Again KGC chooses three collision resistant cryptographic hash functions H_1 and H_2 maps as:
  - $H_1 : \{0,1\}^* \to \mathbb{Z}_p^*$.
  - $H_2 : \{0,1\}^* \times \mathbb{G}_2 \to \mathbb{Z}_p^*$.
  - $H_3 : \mathbb{G}_2 \to \{0,1\}^n$.

  KGC picks randomly $s \xleftarrow{R} \mathbb{Z}_p^*$ as master key and computes its public key as $P_{pub} = sP \in \mathbb{G}_1$. Public system parameters are

$$\texttt{params} = \{\mathbb{G}_1, \mathbb{G}_2, P, g, P_{pub}, H_1, H_2, H_3\}$$

– Partial-Private-Key-Extract: Given an user's identity $\texttt{ID} \in \mathbb{Z}_p^*$, PKG computes the private key as $S_{ID} = \frac{1}{H_1(ID)+s}P \in \mathbb{G}_1$, and sends to the respective user through a secure channel. The user can verify through the equation $e(S_{ID}, P_{pub} + H_1(ID)P) = g$. Let for the sake of convenience, denotes $U = P_{pub} + H_1(ID)P$.

– Set-Secret-Value: The user with identity ID set his secret value v by picking $v \xleftarrow{R} \mathbb{Z}_p^*$ randomly.

– Set-Private-Key: The user with identity ID set his complete private key as $S : (S_{ID}, v)$.

– Set-Public-Key: The user with identity ID computes his public key as $PK_{ID} = v \cdot U$. Hence $PK_{ID_i} = v_i U_i$, for $i = s$ and r denotes as sender and receiver respectively.

– CL-Signcrypt: Given a message $m \in \{0,1\}^*$, sender's private key S_{ID_s}, recipient's identity ID_r. She performs the following steps to compute signcrypt.
  1. Select $\mu \xleftarrow{R} \mathbb{Z}_p^*$ randomly and computes $\lambda = g^{\mu}$.
  2. Set $h = H_2(m, PK_{ID_r})$.
  3. Computes $T = \mu \cdot U_r$.
  4. Computes $c = m \oplus H_3(\lambda)$.
  5. Computes $\sigma = \frac{1}{v_s+h} S_{ID_s}$.

  Signcrypt is $\tau = <c, \sigma, T>$

– CL-UnSigncrypt:Given the ciphertext c, σ, ID_s, ID_r, T and receiver partial private key S_{ID_r} computes the plaintext message as
  1. Computes $\lambda = e(S_{ID_r}, T)$
  2. Computes $m = c \oplus H_3(\lambda)$

– CL-Verify: Given params, m and σ, any user/sender with his identity ID_s verifies as
  1. Computes $h = H_2(m, PK_{ID_s})$.
  2. Accept the message m *iff* the following equation holds

$$g = e(\sigma, PK_{ID_s} + hU_s) \tag{1}$$

  and returns the message m and signature $(\sigma, h) \in \mathbb{G}_1 \times \mathbb{Z}_p^*$.

6.2 Analysis of the scheme

This section, we proof the consistency of the scheme and analyze the security and performance.

6.3 Consistency

$e(S_{ID}, T) = e(\frac{1}{H_1(ID)+s}P, \mu(P_{pub} + H_1(ID)P)$
$= e(\frac{1}{H_1(ID)+s}P, \mu(H_1(ID) + s)P)$
$= e(P, P)^{\mu} = g^{\mu} = \lambda$
Now we verify the consistency of equation 1.
$e(\sigma, PK_{ID_s} + hU_s) = e(\frac{1}{v_s+h}S_{ID_s}, v_sU_s + hU_s)$
$= e(\frac{1}{v_s+h}S_{ID_s}, (v_s + h)U_s)$
$= e(S_{ID_s}, U_s) = e(P, P)$
$= e(P, P) = g$

6.4 Security Analysis

In this section, we present the security proof for `Confidentiality` and `Unforgeability` of our proposed scheme, where hash functions are modeled as random oracle over the game against `Type-I` and `Type-II` adversary defined in Section 4.

Theorem 1. *Under the assumption of intractability of q-BDHIP and CDHP in $\mathbb{G}_1$, the proposed* `CLSC` *scheme is* `IND-iCCA-I` *and* `IND-iCAA-II` *secure in random oracle model respectively.*

Lemma 1. *Assume that there exist an PPT* `IND-iCCA-I` *attacker $\mathcal{A}$ of* `Type-I` *has an advantage ϵ against the proposed* `CLSC` *scheme in time t submitting queries q_{h_i} to the corresponding hash functions H_i, $i = 1, 2$ and 3 modeled as random oracle. Let q_k is query to the secret-value, L_{pk} is query to public key replacement, q_{se} and q_{us} denotes signcrypt and q_{us} unsigncrypt extraction query respectively, then there exist an (ϵ^*, t^*) algorithm $\mathcal{B}$ that can solve q-BDHI problem in $\mathbb{G}_1$ with probability*

$$\epsilon^* > \frac{\epsilon}{q_{h_1}(2q_{h_2}+q_{h_3})}\left(\frac{q_{se}(q_{se}+q_{h_2})}{2^k}\right)\left(\frac{q_{us}}{2^k}\right)$$

within a time

$$t^* < t + \mathcal{O}(q_{h_1}^2)t_{sm} + \mathcal{O}(q_{se} + q_{us})t_{pair} + \mathcal{O}(q_{us}q_{h_2})t_{exp}$$

Where t_{sm} denotes the running time for scalar multiplication on $\mathbb{G}_1$, t_{pair} running time for pairing computation and t_{exp} denotes the running time for exponent operation.

Proof. $\mathcal{A}_I$ runs $\mathcal{B}$ as a subroutine to solve q-BDHIP problem to break `IND-iCCA-I` security. Given instance $(P, \beta P, \beta^2 P \dots \beta^q P)$ q-BDHI problem to $\mathcal{B}$ and computes $e(P,P)^{\frac{1}{\beta}}$. It performs the following phases:

`Preparation Phase`: $\mathcal{B}$ picks randomly $l \xleftarrow{R} \{1, 2 \dots q_{h_1}\}$, elements $\theta_l \xleftarrow{R} \mathbb{Z}_p^*$ and $v_i \xleftarrow{R} \mathbb{Z}_p^*$ and computes $\theta_i = \theta_l - v_i$, for all $i \in \{1, 2 \dots p\} \backslash \{l\}$. We follow the technique of [22], it set up the generator $Q \in \mathbb{G}_1$ and another element $V = \beta Q \in \mathbb{G}_1$ by expanding the polynomial

$$\varphi(x) = \prod_{i=1, i\neq l}^{q}(x + v_i) = \sum_{j=0}^{q-1} c_j x^j$$

Generator Q and the element V of $\mathbb{G}_1$ can be computed as

$$Q = \sum_{j=0}^{q-1}(c_j(\beta^j P) = \varphi(\beta)P$$

and $V = \sum_{j=1}^{q} c_{j-1}(\beta^j P) = \beta\varphi(\beta)P = \beta Q$. As in [22],the pairs $(v_i, \frac{1}{\beta+v_i}Q)$, $\forall\ i = 1, 2 \dots l-1, l+1 \dots q$ are computed by expanding the following polynomial

$$\varphi_i(x) = \frac{\varphi(x)}{x+v_i} = \sum_{j=0}^{q-2} \tau_j x^i$$

and computes

$$\frac{1}{\beta+v_i}Q = \sum_{j=1}^{q-1} \tau_j(\beta^j P) = \frac{\varphi(\beta)}{\beta+v_i}P$$

It setup the public key of `PKG` as $Q_{pub} = -V - \theta_l Q = (-\beta - \theta_l)Q$ and master key as $s = (-\beta - \theta_l) \xleftarrow{R} \mathbb{Z}_p^*$.
Where $(\theta_l, -\frac{1}{(\beta+v_i)}Q)$, $i \xleftarrow{R} \{1, 2, \dots l-1, l+1 \dots q\}$. $\mathcal{B}$ setup the generator Q and system parameters as $Q_{pub} = (-\beta - \theta_l)Q$ and $g = e(Q, Q)$. Simulation of $\mathcal{B}$ is performed in the following two phases:

`Phase-I`: To simulate the hash oracles H_1, H_2 and H_3, $\mathcal{B}$ constructs three list L_1, L_2 and L_3 to store the output of the respective queries submitted to these hash functions. Also $\mathcal{B}$ maintains list L_k, L_{se} and L_{us} for user's public key and secret value tuple, Signcryption and Unsigncryption queries respectively.

At the beginning, $\mathcal{B}$ setup a counter $\pi = 1$ and takes (P, Q, Q_{pub}) as input. Assume that all H_1 queries are distinct, that at some instant, the identity ID_r which is the targeted one is submitted to H_1. Any query associated with ID should have to submit after H_1 query on ID.

- H_1**-Queries**: All the queries submitted to H_1 are indexed by π. On input ID_π, $\mathcal{B}$ returns θ_π and add the entry (ID_π, θ_π) to L_1 and increment π.
- **Extract Partial Private-Key Queries**: On input ID_π, if $\pi = l$, then $\mathcal{B}$ returns fail. Otherwise it knows that $H_1(ID_\pi) = \theta_\pi$ from L_1 and returns $J_\pi = \frac{1}{\theta_\pi + x}Q$ to $\mathcal{A}_I$.

– **Secret-value Queries**: On input ID_π, $\mathcal{B}$ searches the entry $<ID_\pi, \mu_\pi, PK_{ID_\pi}>$ in the list L_k and returns μ_π to $\mathcal{A}_I$. Otherwise, $\mathcal{B}$ picks $\mu_\pi \xleftarrow{R} \mathbb{Z}_p^*$ as the secret-value and computes $PK_{ID_\pi} = \mu_\pi U$, where $U = P_{pub} + \theta_\pi$. In this queries, $\mathcal{B}$ has to searches ID_π and runs H_1-queries. $\mathcal{B}$ includes $<ID_\pi, \mu_\pi, PK_{ID_\pi}>$ to L_k.

– **Request Public-Key Queries**: On input ID_π, $\mathcal{B}$ searches the entry $<ID_\pi, \mu_\pi, PK_{ID_\pi}>$, if founds, then returns PK_{ID_π}, otherwise $\mathcal{B}$ picks $\mu_\pi \xleftarrow{R} \mathbb{Z}_p^*$ randomly and computes $PK_{ID_\pi} = \mu_\pi U$. $\mathcal{B}$ includes $<ID_\pi, \mu_\pi, PK_{ID_\pi}>$ to L_k and returns PK_{ID_π} to $\mathcal{A}_I$.

– **Replace Public-Key Queries**: $\mathcal{A}_I$ submits public key replacement queries on input (ID_π, PK_{ID_π}). If ID_π exists in L_k, $\mathcal{B}$ sets $PK_{ID_\pi} = PK'_{ID_\pi}$, includes the tuples $<ID_\pi, \mu'_\pi, PK_{ID_\pi}>$ to L_{pk}. Here we have to assume that, $\mathcal{B}$ can recovers μ'_π from $\mathcal{A}_I$ as a secret value to the corresponding replaced public key PK'_{ID_π}. Otherwise $\mathcal{B}$ runs the public key extraction query to generate the tuple $<ID_\pi, \mu_\pi, PK_{ID_\pi}>$, then sets $PK_{ID_\pi} = PK'_{ID_\pi}$ and includes $<ID_\pi, \mu'_\pi, PK_{ID_\pi}>$ to L_{pk}.

– H_2-**Queries**: L_2 contains the tuples of type $<m_j, ID_\pi, PK_{ID_\pi}, h_{2,j}>$. On input (ID_π, m_j), $\mathcal{B}$ runs H_2 query. She picks $h_{2,j} \xleftarrow{R} \mathbb{Z}_p^*$ randomly.Sets $h_{2,j} = H_2(m_j, PK_{ID_\pi})$ and includes $<m_j, ID_\pi, PK_{ID_\pi}, h_{2,j}>$ to L_2. Then sends $h_{2,j}$ to $\mathcal{A}_I$.

– H_3-**Queries**: $\mathcal{B}$ recovers the secret value μ_π from L_k and computes the value λ. On input λ, $\mathcal{B}$ runs the query H_3. She picks random value $h_{3,j} \xleftarrow{R} \{0,1\}^n$ and sets $h_{3,j} = H_3(\lambda)$ and add the entry $(\lambda, h_{3,j})$ to the list L_3.

– **Signcryption Queries**: At any time, $\mathcal{B}$ simulates Signcryption queries on message m and identities $(ID_s, ID_r) = (ID_\rho, ID_\pi)$, for $\rho, \pi \in \{1, 2 \dots q_{h_1}\}$. Note that, if $\rho \neq l$, $\mathcal{B}$ knows the private key of sender *i*.e $S_{ID_\rho} = J_\rho$, where $J_\rho = -\frac{1}{\theta_\rho + x} P$ and can answer the query based on the construction of `Signcryption`. So we assume $\rho = l$ and $\pi \neq l$ by the reflexivity assumption [22]. Note that $\mathcal{B}$ knows the private key of the recipient *i*.e $S_{ID_\pi} = J_\pi$. The computational infeasibility to find a random tuple $(\sigma, U, h) \in \mathbb{G}_1 \times \mathbb{G}_1 \times \mathbb{Z}_p^*$ for which the following equation holds:

$$e(U, S_{ID_\pi}) = e(\sigma, PK_{ID_l} + hU) \tag{2}$$

To perform so $\mathcal{B}$ picks $\xi, h \xleftarrow{R} \mathbb{Z}_p^*$ and computes the following:
  - $\sigma = \xi^{-1} S_{ID_\pi}$
  - $U(1 - \xi^{-1} S_{ID_\pi}) = \xi^{-1} PK_{ID_l}$

in order to get the required equality $e(S_{ID_\pi}, U) = e(\sigma, PK_{ID_l} + hU)$ before patching the hash value $H_2(m, PK_{ID_\pi})$ to h_2. $\mathcal{B}$ fails if H_2 is defined already but is only happen with probability $(q_s + q_{h_2}) \frac{1}{2^k}$. Returns signcrypt $\tau = (m \oplus H_3(\lambda), \sigma, U)$.

– **UnSigncryption Queries**: On input the signcrypt $\tau = <c, \sigma, U>$ for identities $(ID_s, ID_r) = (ID_\rho, ID_\pi)$, $\mathcal{B}$ submits this queries. Let us assume that, $\pi = l$ and by irreflexivity property $\rho \neq l$, because otherwise $\mathcal{B}$ knows the private key of the recipient $S_{ID_\pi} = -J_\pi$ can executes `UnSigncrypt` algorithm normally. Since $\rho \neq l$, $\mathcal{B}$ has the private key S_{ID_ρ} of the sender and knows that for all valid signcrypt $\log_{SD_\rho}(\sigma) = \log_{(PK_{ID_\pi} + hQ_{pub} + h\theta_\pi Q)}(Q_{pub} + \theta_\pi Q)$, where $h = H_2(m, PK_{ID_\pi})$ is the digest obtains in the `Signcrypt` algorithm and $U = Q_{pub} + \theta_\pi Q$. Hence we have the equation

$$e(U, S_{ID_\rho}) = e(\sigma, PK_{ID_\pi} + hU) \tag{3}$$

Hence, the query is managed by computing $\psi = e(\sigma, PK_{ID})$ and search the entries $((m_i, \lambda, h_{2,i}, c, \psi)$ indexed by $i = \{1, 2, \dots q_{h_2}\}$, if not found τ is rejected, otherwise continue the simulation for each indexes and $\mathcal{B}$ checks the following equation

$$\frac{e(U, S_{ID_\rho})}{e(\sigma, PK_{ID_\pi})} = e(\sigma, U)^{h_{2,i}} \tag{4}$$

All pairing computations are performed once and it needs maximum q_{h_2} number of exponentiations that satisfies equation- 3. If there exists a unique index $i \in \{1 \dots q_{h_2}\}$ that satisfies equation-4 then it returns the matching pairs $(m_i, <h_{2,i}, \sigma>)$, otherwise it rejects τ. In general, an unsuitable rejection occurs with probability $Pr[\tau = \text{“}\texttt{reject}''] \leq \frac{q_u}{2^k}$ during the entire game.

`Challenge`: During the challenge phase, $\mathcal{A}$ sets two plaintext messages (m_0, m_1) of equal length and the identities (ID_s, ID_r) on which it would like to be challenged. If $ID_r \neq ID_l$, $\mathcal{B}$ aborts, otherwise $\mathcal{B}$ picks $\zeta \xleftarrow{R} \mathbb{Z}_p^*, c \xleftarrow{R} \{0,1\}^n$ and $\sigma \xleftarrow{R} \mathbb{G}_1$ randomly and outputs $\tau^* = <c, \sigma, U>$ where $U = -\zeta Q \in \mathbb{G}_1$. Let set $\gamma = \zeta/\beta$. We have

$$U = -\zeta Q = -\gamma\beta Q = (s+\theta_l)\gamma Q = \theta_l\gamma Q + \gamma Q_{pub}.$$

Hence as long as $\mathcal{A}_I$ does not query H_2 or H_3 on input $e(Q,Q)^{\gamma}$, she cannot recognize that τ^* is an invalid ciphertext.

`Phase-II` :$\mathcal{A}_I$ can submit polynomially bounded number of queries in an adaptive manner against Phase-I with constraint that she cannot submit a key extraction query on ID_r and cannot make an unsigncryption query on τ^* to obtain the corresponding plaintext. $\mathcal{B}$ answers the queries of $\mathcal{B}$ as in Phase-I. During the guess stage, her view is simulated as before and the eventual output obtained is ignored. To produce a result, $\mathcal{B}$ fetches a random entry $(m, \lambda, h_2, c, \psi)$ or $(\lambda, .)$ from the list L_2 or L_3. Since L_3 contains maximum $(q_{h_2} + q_{h_3})$ number of records, the selected entry will contain the valid element $\lambda = e(Q,Q)^{\gamma} = e(P,P)^{\varphi(\beta)^2\zeta/\beta}$ with probability $\frac{1}{2q_{h_2}+q_{h_3}}$, where $\varphi(x) = \sum_{j=0}^{q-1} c_j x^j$ is the polynomial for which $Q = \varphi(\beta)P$. The solution of q-BDHIP can be computed by noting that, if $\psi^* = e(Q,Q)^{\frac{1}{\beta}}$, then

$$e(Q,Q)^{\frac{1}{\beta}} = \psi^{*c_0^2} e(\textstyle\sum_{i=0}^{q-2}(c_{j+1}(\beta^j P), c_0 P))e(Q, \sum_{j=0}^{q-2} c_{i+2}(\beta^j)P)$$

<u>`Probability Analysis`</u>

We can analyze the advantages of $\mathcal{B}$ and calculate the probability of successes. Note that due to the occurrence of one of the following independent events, it fails to provide a consistent simulation.

- E_1: $\mathcal{A}_I$ does not choose the identity ID_l of the receiver during the challenge phase.
- E_2: a key extraction query is made on ID_l.
- E_3: Because of collision property of H_2, $\mathcal{B}$ aborts in a Signcryption query.
- E_4: At a particular instant of the game, $\mathcal{B}$ aborts Unsigncryption query since rejecting a valid ciphertext.

So the probability of the above invents are

- $\Pr[\neg E_1] = \frac{1}{q_{h_1}}$
- $\neg E_1 \Rightarrow \neg E_2$
- $\Pr[\neg E_3] \geq (1 - \frac{q_{se}(q_{se}+q_{h_2})}{2^k})$
- $\Pr[\neg E_4] \geq (1 - \frac{q_{us}}{2^k})$

Hence the probability of $\mathcal{B}$ does not abort is

$$\Pr[\neg\texttt{abort}] = \Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4]$$

$\Rightarrow \Pr[\neg\texttt{abort}] \geq \frac{1}{q_{h_1}}(1 - \frac{q_{se}(q_{se}+q_{h_2})}{2^k})(1 - \frac{q_{us}}{2^k})$
Further $\mathcal{B}$ selects the valid element from L_2 or L_3 with probability $\frac{1}{q_{h_1}(2q_{h_2}+q_{h_3})}$ Hence

$$\epsilon^* \geq \frac{\epsilon}{q_{h_1}(2q_{h_2}+q_{h_3})}(1 - \frac{q_{se}(q_{se}+q_{h_2})}{2^k})(1 - \frac{q_{us}}{2^k})$$

<u>`Running time`</u>

Running time is computed by calculating the various operations that are to be performed by *mathcalB* during the above simulation of all oracles.

Let to compute the bound of $\mathcal{B}$'s computation as she requires $\mathcal{O}(q_{h_1}^2)$ scalar multiplications in $\mathbb{G}_1$ in preparing phase, $\mathcal{O}(q_{se}+q_{us})$ number of pairing operation and $\mathcal{O}(q_{us}q_{h_2})$ exponents in $\mathbb{G}_2$. Hence the $\mathcal{B}$'s running time is bounded by

$$t^* < t + \mathcal{O}(q_{h_1}^2)t_{sm} + \mathcal{O}(q_{se}+q_{us})t_{pair} + \mathcal{O}(q_{us}q_{h_2})t_{exp}$$

Lemma 2. *Assume that there exist an PPT* `IND-iCCA-II` *attacker $\mathcal{A}$ of* `Type-II` *has an advantage ϵ against the proposed* `CLSC` *scheme in time t submitting queries q_{h_i} to the corresponding hash functions H_i, $i = 1, 2$ and 3 modeled as random oracle. Let q_k is query to the secret-value, q_{se} and q_{us} denotes signcrypt and q_{us} unsigncrypt extraction query respectively, then there exist an (ϵ^*, t^*) algorithm $\mathcal{B}$ that can solve CDH problem in $\mathbb{G}_1$ with probability*

$$\epsilon^* > \frac{\epsilon}{q_w(2q_{h_2}+q_{h_3})}(1 - \frac{q_{se}(q_{se}+q_{h_2})}{2^k})(1 - \frac{q_{us}}{2^k})$$

within a time

$$t^* < t + \mathcal{O}(q_w^2)t_{sm} + \mathcal{O}(q_{se} + q_{us})t_{pair} + \mathcal{O}(q_{us}q_{h_2})t_{exp}$$

Proof. $\mathcal{A}_{II}$ run $\mathcal{B}$ as a subroutine to solve CDH problem to break `IND-iCCA-II` security. Given tuples $<U, aU, bU>$, she has to compute abU in $\mathbb{G}_1$. It performs the following phases:

`Preparation Phase`: $\mathcal{B}$ generates master secret key s and P_{pub}. Sends to $\mathcal{A}_{II}$. $\mathcal{B}$ chooses $l \overset{R}{\longleftarrow} \{1, 2, \dots q_w\}$.
`Phase-I`: $\mathcal{A}_{II}$ is given access to run a series of queries in an adaptive manner.

- H_1-**Queries**: All the queries submitted to H_1 are indexed by π. On input ID_π, $\mathcal{B}$ returns θ_π and add the entry (ID_π, θ_π) to L_1 and increment π.
- **Extract Partial Private-Key Queries**: In `Type-II` model, `KGC` is modeled an honest but curious, the adversary knows the master secret key s who can generate partial private key for himself. So it does not require Partial-Private key extraction query.
- **Request Public-Key Queries**: On input ID_π, $\mathcal{B}$ searches the entry $<ID_\pi, \mu_\pi, PK_\pi>$, if founds, then returns PK_{ID_π}, otherwise $\mathcal{B}$ picks $\mu_\pi \overset{R}{\longleftarrow} \mathbb{Z}_p^*$ randomly and computes $PK_{ID_\pi} = \mu_\pi U$. $\mathcal{B}$ includes $<ID_\pi, \mu_\pi, PK_\pi>$ to L_k and returns PK_{ID_π} to $\mathcal{A}_I$.
- **Secret-value Queries**: On input ID_π, if $\pi = l$ *i*.e, $\mathcal{B}$ searches the entry $<ID_\pi, \mu_\pi, PK_\pi>$ in the list L_k and returns μ_π to $\mathcal{A}_I$ and aborts the simulation. Otherwise, $\mathcal{B}$ picks $\mu_\pi \overset{R}{\longleftarrow} \mathbb{Z}_p^*$ as the secret-value and computes $PK_{ID_\pi} = \mu_\pi U$, where $U = P_{pub} + \theta_\pi$. In this queries, $\mathcal{B}$ has to searches ID_π and runs H_1-queries. $\mathcal{B}$ includes $<ID_\pi, \mu_\pi, PK_\pi>$ to L_k.
- **Replace Public-Key Queries**: On input ID_π, if $\pi = l$, $\mathcal{B}$ aborts the simulation, otherwise $\mathcal{A}_I$ submits public key replacement queries on input (ID_π, PK_{ID_π}). If ID_π exists in L_k, $\mathcal{B}$ replaces $<ID_\pi, \mu_\pi, PK_\pi>$ with $<ID_\pi, \mu_\pi, \perp PK>$. Simultaneously also $\mathcal{B}$ has to run H_1 queries and send the obtain public key PK to $\mathcal{A}_I$.
- H_2-**Queries**: L_2 contains the tuples of type $<m_j, ID_\pi, PK_{ID_\pi}, h_{2,j}>$. On input (ID_π, m_j),$\mathcal{B}$ runs H_2 query. She picks $h_{2,j} \overset{R}{\longleftarrow} \mathbb{Z}_p^*$ randomly.Sets $h_{2,j} = H_2(m_j, PK_{ID_\pi})$ and includes $<m_j, ID_\pi, PK_{ID_\pi}, h_{2,j}>$ to L_2. Then sends $h_{2,j}$ to $\mathcal{A}_I$.
- H_3-**Queries**: $\mathcal{B}$ recovers the secret value μ_π from L_k and computes the value λ. On input λ, $\mathcal{B}$ runs the query H_3. She picks random value $h_{3,j} \overset{R}{\longleftarrow} \{0,1\}^n$ and sets $h_{3,j} = H_3(\lambda)$ and add the entry $(\lambda, h_{3,j})$ to the list L_3.
- **Signcryption Queries**: Sender generates signcrypt on message $m \in \{0,1\}^*$ and sends to the receiver. Assume that both the sender and receiver's public keys and recever's secret values have been queried, if public key of receiver has been replaced, then $\mathcal{B}$ should provide the corresponding secret value and simulates as:

  - Case-I: $ID_s \neq ID_l$, ID_s's full private key $S : (S_{ID_s}, \mu_s)$ could be queried and $\mathcal{B}$ returns the signcryption (σ, U, h).
  - Case-II: $ID_s = ID_l$, $\mathcal{B}$ picks the value $\mu, s \overset{R}{\longleftarrow} \mathbb{Z}_p^*$ and computes $U = P_{pub} + \theta_s P$, $\lambda = g^\mu$, $c = m \oplus H_3(\lambda)$ and $\sigma = \frac{\mu}{\mu + h}(PK_{ID_s})^{-1}P$, where the hash value θ_s, h and k are obtained from the simulation of H_1, H_2 and H_3 oracle respectively. Then $<c, \sigma, U>$ is a valid signcryption since it can pass the verification equation 1.
  - **UnSigncryption Queries**: The receiver obtains $<c, \sigma, U>$ from the sender. We consider that the respective user's public keys and secret values have been queried. $\mathcal{B}$ simulates as
    - Case-I: $ID_r \neq ID_l$. It is similar to the proof of lemma-1.
    - Case-II: $ID_r = ID_l$, $\mathcal{B}$ computes $\lambda = e(S_{ID_r}, PK_{ID_r})$. $\mathcal{A}_{II}$ decrypts the message as $c \oplus H_3(\lambda)$. Then she verifies the equation $g = e(\sigma, PK_{ID_r} + hU)$. If successes, $\mathcal{A}_{II}$ accepts the message, otherwise interrupts the simulation.

  `Challenge`: $\mathcal{A}_{II}$ returns $<m_0, m_1, ID_s^*, ID_r^*>$ to be challenged. If $ID_r = ID_j$, then $\mathcal{B}$ aborts the simulation, otherwise she picks $\delta \in \{0,1\}$, $\mu^* \overset{R}{\longleftarrow} \mathbb{Z}_p^*$ and sets $\lambda^* = g^{\mu^*}$, $c^* = m_\delta \oplus k^*$, $U^* = sP + \theta_s^* P$ and h^*. Where $k^* = H_3(\lambda^*)$. It returns $<c^*, \sigma^*, U^*>$ to the adversary $\mathcal{A}_{II}$.

`Phase-II`: $\mathcal{A}_{II}$ is not allowed to submit Unsigncrypt query on τ^* under ID_s^* and ID_r^* as sender and receiver respectively. Hence the public key of ID_r^* can not be replaced.
`Guess`: $\mathcal{A}_{II}$ returns $\delta' \in \{0,1\}$ as her guess. $\mathcal{A}_{II}$ wins the game if $\delta = \delta'$. It needs to solve CDH problem. $\mathcal{B}$ searches the entry $<\lambda, ID_s, ID_r, \mu, h_3>$ in L_3, there exist the value $bPK_{ID_r^*} = abU$ with the chance of $\frac{1}{(2q_{h_2} + q_{h_3})}$.

`Probability Analysis`

We can analyze the advantages of $\mathcal{B}$ and calculate the probability of successes. Note that due to the occurrence of one of the following independent events, it fails to provide a consistent simulation.

- E_1: $\mathcal{A}_{II}$ does not ask the secret value identity ID_l during the phase-I.
- E_2: $\mathcal{A}_{II}$ does not challenged on the identity ID_l, if E_1 happens.
- E_3: Because of collision property of H_2, $\mathcal{B}$ aborts in a Signcryption query.
- E_4: At a particular instant of the game, $\mathcal{B}$ aborts Unsigncryption query since rejecting a valid ciphertext.

So the probability of the above invents are

- $\Pr[\neg E_1] = \frac{1}{q_w}$
- $\neg E_1 \Rightarrow \neg E_2$
- $\Pr[E_3] \geq (1 - \frac{q_{se}(q_{se}+q_{h_2})}{2^k})$
- $\Pr[E_4] \leq (1 - \frac{q_{us}}{2^k})$

Hence the probability of $\mathcal{B}$ does not abort is

$$\Pr[\neg\texttt{abort}] = \Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4]$$

$\Rightarrow \Pr[\neg\texttt{abort}] \geq \frac{1}{q_w}(1 - \frac{q_{se}(q_{se}+q_{h_2})}{2^k})(1 - \frac{q_{us}}{2^k})$
Further $\mathcal{B}$ selects the valid element from L_2 or L_3 with probability $\frac{1}{q_w}(2q_{h_2} + q_{h_3})$

$$\epsilon^* > \frac{\epsilon}{q_w(2q_{h_2}+q_{h_3})}(1 - \frac{q_{se}(q_{se}+q_{h_2})}{2^k})(1 - \frac{q_{us}}{2^k})$$

`Running time`
Let to compute the bound of $\mathcal{B}$'s computation as she requires $\mathcal{O}(q_w^2)$ scalar multiplications in $\mathbb{G}_1$ in preparing phase, $\mathcal{O}(q_{se} + q_{us})$ number of pairing operation and $\mathcal{O}(q_{us}q_{h_2})$ exponents in $\mathbb{G}_2$ during the simulation of all the oracles. Hence bound of $\mathcal{B}$'s running time is

$$t^* < t + \mathcal{O}(q_w^2)t_{sm} + \mathcal{O}(q_{se} + q_{us})t_{pair} + \mathcal{O}(q_{us}q_{h_2})t_{exp}$$

Theorem 2. *Under the assumption of intractability of k-CAA and Inv-CDH in $\mathbb{G}_1$, the proposed* `CLSC` *scheme is* `sUF-iCMA-I` *and* `sUF-iCMA-II` *secure in random oracle model respectively.*

The theorem follows from Lemmas 3 & 4.

Lemma 3. *Assume that there exist an PPT* `sUF-iCMA-I` *attacker $\mathcal{A}_I$ of* `Type-I` *has an advantage ϵ against the proposed* `CLSC` *scheme in time t submitting queries q_{h_i} to the corresponding hash functions H_i, $i = 1, 2, 3$ modeled as random oracle. Let q_{ppk}, q_{pk}, q_{qk}, $q_{q_{se}}$ and q_{uc} denotes query to the partial private-key extraction, private key extraction oracle, public key request, signcryption and unsigncryption oracles respectively, then there exist an (ϵ^*, t^*) algorithm $\mathcal{B}$ that can solve k-CAA problem in $\mathbb{G}_1$ with probability*

$$\epsilon^* \geq \frac{1}{q_1}(1 - \frac{q_{se}(q_{se}+q_{ppk}+q_{pk})}{2^k})(1 - \frac{q_{us}}{2^k})$$

$$t^* < t + \mathcal{O}(q_1^2 + q_{se})t_{sm} + \mathcal{O}(q_{se})t_{inv} + \mathcal{O}(q_{se})t_{exp}$$

Proof. Let the forger apply a reliable simulation algorithm $\mathcal{B}$ and solve k-CAA problem. $\mathcal{B}$ is given a random instance $P, U = sP \leftarrow \mathbb{G}_1$, $\{Q_1, Q_2 \ldots Q_q\} \leftarrow \mathbb{Z}_q^*$ and
$\{\frac{1}{s+Q_1}P, \frac{1}{s+Q_2}P \ldots \frac{1}{s+Q_q}P\} \in \mathbb{G}_1$, after interacting with `Type-I` adversary $\mathcal{A}_I$ the algorithm $\mathcal{B}$ has to compute a pair $(Q^*, \frac{1}{s+Q^*}P$ for some $Q^* \notin \{Q_1, Q_2 \ldots Q_q\}$. The game is played between $\mathcal{C}$ and $\mathcal{A}_I$. $\mathcal{B}$ picks $l \xleftarrow{R} \{1, 2 \ldots q_1\}$.

- **Setup**: To set $g = e(P, P)$ and $P_{pub} = sP$, $\mathcal{B}$ initializes these parameters by running `Setup` algorithm. The master secret key s is unknown to $\mathcal{B}$. $\mathcal{B}$ chooses a challenged $ID \xleftarrow{R} \mathbb{Z}_p^*$ and returns the public parameters $\{P, g, P_{pub}, H_1, H_2\}$ and sends to $\mathcal{F}_I$. Let us we assume that, before submitting any queries to H_2, partial private-key extraction and private-key extraction taking ID_i as input to the algorithm then $\mathcal{A}_I$ submits queries to H_1.
  `Phase-I`: $\mathcal{F}_I$ runs the algorithm $\mathcal{B}$ as subroutine and access to series of oracles adaptively.

- H_1-**Queries**: On input ID_i, $\mathcal{B}$ returns $\theta_i = H_1(ID_i) = Q_i$ and includes the entry $<i, ID_i, \theta_i, J_i>$, where $J_i = \frac{1}{s+Q_i}P$ to L_1. If $i = l$, $\theta_l = H_1(ID_l) = Q^*$ which is unknown to $\mathcal{B}$.
- **Extract Partial Private-Key Queries**: When $\mathcal{F}_I$ asks for ID_l, $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ retrieves J_i from L_1.
- **Signcryption Queries**: When the sender likes to deliver the message $m = \{0,1\}^*$ to the receiver, it undergoes the following two cases.
  - `Case-I`: If $ID_s \neq ID_l$, $\mathcal{B}$ can extract the public key PK_{ID_s} and full private key $S_s : (\mu_s, J_s)$ by running Public Key, Partial-Private key and Secret-value extraction queries. Then she executes `CL-Signcryption` algorithm.
  - `Case-II`:If $ID_s = ID_l$, $\mathcal{B}$ picks $\mu, h \xleftarrow{R} \mathbb{Z}_p^*$ and computes $U = P_{pub} + \theta_s P$, where $\theta_s = H_1(ID_s)$, $\sigma = \frac{1}{(\mu_s+h)}J_s, \lambda = g^{\mu}, c = m \oplus H_3(\lambda)$. Eventually $\mathcal{B}$ returns $\tau = <c, \sigma, U>$ provides to $\mathcal{F}_I$. All the simulation of hash oracles for H_1, H_2 and H_3 are same that have been described in the previous lemma.
- **UnSigncryption Queries**: In Unsigncryption queries, it undergoes the following two cases:
  - `Case-I`: If $ID_r \neq ID_l$, $\mathcal{B}$ executes `CL-UnSigncrpt` by the following algorithm
  $$m \leftarrow \mathtt{CL-UnSigncrypt}(params, ID_s, PK_{ID_s}, ID_r, PK_{ID_r}, S_r, \tau).$$
  Then $\mathcal{B}$ can verifies through the verification equation. If it cannot pass, it returns $\perp$, otherwise sends m to $\mathcal{F}_I$.
  - `Case-II`: If $ID_r = ID_l$, $\mathcal{F}_I$ cannot compute the value $\lambda = e(P,P)^{\mu}$. $\mathcal{B}$ recovers L_3 with tuple $<*, h_3>$. If there exists corresponding tuple matches with it, the take λ form the list and computes $m = c \oplus h_3$, otherwise $\mathcal{B}$ selects a random value $\lambda \in \mathbb{G}_1$ and generates decryption key h_3. Then $\mathcal{F}_I$ run the `CL-UnSigncrypt` algorithm using the key and perform the verification process. If the verification equation holds, $\mathcal{F}_I$ accepts m, otherwise returns `''failure''`.

`Forgery phase`: At the end of the simulation, $\mathcal{F}_I$ returns $\tau^* = <c^*, \sigma^*, U^*>$ as a valid signcryption with respect to the targeted identities ID_s^*, ID_r^*. If $ID_s \neq ID_l$, $\mathcal{B}$ aborts the simulation and repeats by choosing the same random values, but different value of μ. It computes $\lambda_1^* = g^{\mu_1^*}$ and $\lambda_2^* = g^{\mu_2^*}$. $c_1^* = m \oplus H_3(\lambda_1^*)$ and $c_2^* = m \oplus H_3(\lambda_2^*)$. Similarly also computes h_1^* and h_2^*. Then computes two different σ_1^* and σ_2^* as

$$\begin{aligned}
\sigma_1^* = \tfrac{1}{\mu_1^*+h_1^*}J_{ID}^*,\ \sigma_2^* &= \tfrac{1}{\mu_2^*+h_2^*}J_{ID}^* \\
\sigma_1^* - \sigma_2^* &= \{\tfrac{1}{\mu_1^*+h_1^*} - \tfrac{1}{\mu_2^*+h_2^*}\}J_{ID}^* \\
J_{ID}^* &= (\sigma_1^* - \sigma_2^*)\{\tfrac{1}{\mu_1^*+h_1^*} - \tfrac{1}{\mu_2^*+h_2^*}\}^{-1} \\
\Rightarrow \tfrac{1}{s+Q^*}P &= (\sigma_1^* - \sigma_2^*)\{\tfrac{1}{\mu_1^*+h_1^*} - \tfrac{1}{\mu_2^*+h_2^*}\}^{-1}
\end{aligned}$$

Hence $\mathcal{B}$ can succeeds computing the group element $\frac{1}{s+Q^*}P$ and returns the pair $(Q^*, \frac{1}{s+Q^*}P)$ as a solution to the challenge of $\mathcal{F}_I$ for $Q^* \notin \{Q_1, Q_2 \dots Q_q\}$.

`Probability Analysis`

We can analyze the advantages of $\mathcal{B}$ and calculate the probability of successes. Note that due to the occurrence of one of the following independent events, it fails to provide a consistent simulation.

- E_1: $\mathcal{F}_I$ does not ask the partial private key.
- E_2: $\mathcal{F}_I$ does not ask sender's identity as ID_l during the forgery phase.
- E_3: Because of collision property of H_2 and H_3, $\mathcal{B}$ aborts in a Signcryption query.
- E_4: At a particular instant of the game, $\mathcal{B}$ aborts Unsigncryption query that generates a valid ciphertext.

So the probability of the above invents are

- $\Pr[\neg E_1] = \frac{1}{q_1}$
- $\neg E_1 \Rightarrow \neg E_2$
- $\Pr[E_3] \geq 1 - \frac{q_{se}(q_{se}+q_{ppk}+q_{pk})}{2^k}$
- $\Pr[E_4] \geq (1 - \frac{q_{us}}{2^k})$

Hence the probability of $\mathcal{B}$ does not abort is

$$\begin{aligned}
\Pr[\neg\mathtt{abort}] &= \Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4] \\
\Rightarrow \Pr[\neg\mathtt{abort}] &\geq \tfrac{1}{q_1}(1 - \tfrac{q_{se}(q_{se}+q_{ppk}+q_{pk})}{2^k})(1 - \tfrac{q_{us}}{2^k})
\end{aligned}$$

Further $\mathcal{B}$ selects the valid element from L_2 or L_3 with probability $\frac{1}{q_{h_1}(2q_{h_2}+q_{h_3})}$

$$\epsilon^* \geq \frac{1}{q_1}(1-\frac{q_{se}(q_{se}+q_{ppk}+q_{pk})}{2^k})(1-\frac{q_{us}}{2^k})$$

<u>`Running time`</u>
Let to compute the bound of $\mathcal{B}$'s computation as she requires $\mathcal{O}(q_1^2+q_{se})$ number of scalar multiplications in preparation phase and signcryption oracle query and one inversion and exponent operation in signcryption oracle query in $\mathbb{G}_2$. Hence the bound of $\mathcal{B}$'s running time is

$$t^* < t + \mathcal{O}(q_1^2+q_{se})t_{sm} + \mathcal{O}(q_{se})t_{inv} + \mathcal{O}(q_{se})t_{exp}$$

Lemma 4. *Assume that there exist an PPT* `sUF-iCMA-II` *attacker $\mathcal{A}_{II}$ of* `Type-II` *has an advantage ϵ against the proposed* `CLSC` *scheme in time t submitting queries q_{h_i} to the corresponding hash functions H_i, $i = 1, 2, 3$ modeled as random oracle. Let q_{ppk}, q_{se} and q_{uc} denotes query to the partial private-key extraction, query to signcryption and query to unsigncryption, then there exist an (ϵ^*, t^*) algorithm $\mathcal{B}$ that can solve Inv-CDH problem in $\mathbb{G}_1$ with probability*

$$t^* < t + \mathcal{O}(q_1^2+q_{se})t_{sm} + \mathcal{O}(q_{se})t_{inv}$$

Proof. To proof the lemma, the challenger constructs an algorithm $\mathcal{B}$ as subroutine to solve Inv-CDH problem over $\mathbb{G}_1$. It takes the Inv-CDH instance $(\alpha, P, \alpha P)$ as input to computes $\frac{1}{\alpha}P$. So goal of $\mathcal{B}$ is to computes $\frac{1}{\mu+h}P$ over the group $\mathbb{G}_1$. Let the `Type-II` adversary is denoted by $\mathcal{F}_{II}$. $\mathcal{B}$ chooses $l = \{1, 2 \ldots q_1\}$ answers the oracle queries as follows:

– **Setup**: To set $g = e(P,P)$ and $P_{pub} = sP.$, $\mathcal{B}$ initializes these parameters by running `Setup` algorithm.
  $\mathcal{F}_{II}$ submits a series of polynomially bounded number of queries in an adaptive manner.
– H_1-**Queries**: All the queries submitted to H_1 are indexed by π. On input ID_π, $\mathcal{B}$ returns θ_π and add the entry (ID_π, θ_π) to L_1 and increment π.
– H_2-**Queries**: On input (m, λ), $\mathcal{B}$ returns the previously defined value if it exist, otherwise it returns a random $h_2 \overset{R}{\longleftarrow} \mathbb{Z}_p^*$. To look forward probable subsequent `Unsigncrypt/Verify` requests moreover $\mathcal{B}$ has to simulates H_3 random oracle to get $h_3 = H_3(\lambda) \in \{0,1\}^n$. Includes the entry $(m, \lambda, h_2, c, \psi)$ to L_2, where $\psi = e(\sigma, PK_{ID})$.
– H_3-**Queries**: It calls the input λ, $\mathcal{B}$ returns the preceding value, if it exists, otherwise select random value $h_3 \overset{R}{\longleftarrow} \{0,1\}^n$ and add the entry (λ, h_3) to the list L_3.
– **Partial-Private-Key query**: $\mathcal{F}_{II}$ can use the master secret key and computes the partial-private-key. Hence it does not require to run this query.
– **Secret-value Query**: $\mathcal{B}$ terminates the simulation if $\mathcal{F}_{II}$ asks the queries on secret value chosen by ID_l.
– **Request Public-Key Queries**: In this query, On input ID_l, $mathcalB$ set the public key as $PK_{ID_l} = \mu_l U$ send to $\mathcal{F}_{II}$.
– **Replace-Public-Key Queries**: $\mathcal{B}$ terminates the simulation if $\mathcal{F}_{II}$ asks the queries on input ID_l.
– **Signcryption Queries**: Sender with identity ID_s sends transmits the message $m \in \{0,1\}^*$. In this query, on input $<ID_l, \tilde{m}>$, $\mathcal{B}$ performs as
  - If $ID_s \neq ID_l$, $\mathcal{F}_{II}$ can recovers the public key PK_{ID_s} and full private key $S = (S_{ID_s}, \mu_s)$ from the above defined oracles and $\mathcal{B}$ runs the `CL-Signcrypt` algorithm.
  - If $ID_s = ID_l$, $\mathcal{B}$ retrieves the corresponding entry $<\tilde{m}, ID_l, Q_l, PK_{ID_l}, h_{2,l}>$ from L_2 list and computes $\tilde{\sigma} = \frac{1}{(\mu_l+h_{2,l})(s+Q_l)}P$. Simultaneously it runs H_3 query oracle and retrieve λ_i and computes $\tilde{c} = \tilde{m} \oplus H_3(\lambda)$. $\mathcal{B}$ returns the signcrypt τ and sends to $\mathcal{F}_{II}$.
– **UnSigncryption Queries**: When the recipient receives $\tau = <c, \sigma, U>$, $\mathcal{B}$ needs to consider following two conditions.
  - If $ID_r \neq ID_l$, on input ID_r, $\mathcal{F}_{II}$ runs secret value queries. The $\mathcal{B}$ computes the message m by performing `CL-Unsigncrypt` algorithm as
    $$m \leftarrow \texttt{CL-Unsigncrypt}(params, ID_s, PK_{ID_s}, ID_r, PK_{ID_r}, S_r, \tau)$$
    and verifies. If the verification equation holds, returns $\perp$, otherwise transmits m to $\mathcal{F}_{II}$.
  - If $ID_r = ID_l$, λ can be computed $\mathcal{B}$ recovers L_3 with tuple $<*, h_3>$. If there exists corresponding tuple matches with it, the take λ form the list and computes $m = c \oplus h_3$, otherwise $\mathcal{B}$ selects a random value $\lambda \in \mathbb{G}_1$ and generates decryption key h_3. Then $\mathcal{F}_{II}$ runs the `CL-UnSigncrypt` algorithm using the key and perform the verification process. If the verification equation holds, $\mathcal{F}_{II}$ accepts m, otherwise returns `‘‘failure’’`.

`Forgery Phase`: Then for identity ID_i, $\tilde{\sigma}$ is the sincrypt. Eventually $\mathcal{F}_{II}$ returns a signcrypt τ^* on message m^* by the `CL-Signcrypt` algorithm taking the public key PK^*_{ID} for the identity ID^* which passes the verification equation. If $ID^* \neq ID_l$, $\mathcal{B}$ returns ``` ``failure '' ``` and terminates the simulation. If $ID^* = ID_l$, $\mathcal{B}$ retrieves the corresponding tuple $<m^*, ID^*, Q^* h^*_2>$ from the list L_2. Now consider the verification equation $e(P,P) = e(\sigma^*, PK_{ID^*} + h^*U)$.
$e(\sigma^*, PK_{ID^*} + h^*U) = (\sigma^*, PK^*_{ID} + h^*(P_{pub} + Q^*P))$
$= e(\sigma^*, \mu(P_{pub} + Q^*P) + h^*(P_{pub} + Q^*P))$
$= e(\sigma^*, (\mu + h^*)(P_{pub} + Q^*P)$
$= e(\sigma^*, (\mu + h^*)(s + Q^*)P)$
$= e(P, \sigma^*(\mu + h^*)(s + Q^*)) = e(P,P)$
$\Rightarrow \sigma^*(\mu + h^*)(s + Q^*) = P$
$\Rightarrow \frac{1}{(\mu+h^*)}P = (s + Q^*)\sigma^*$
Hence $\mathcal{B}$ can succeeds to compute the group element $\frac{1}{(\mu+h^*)}P = (s + Q^*)\sigma^*$. In the other ward, she can solve Inv-CDH problem in $\mathbb{G}_1$.

`Probability Analysis`
We can analyze the advantages of $\mathcal{B}$ and calculate the probability of successes. Note that due to the occurrence of one of the following independent events, it fails to provide a consistent simulation.

- E_1: $\mathcal{A}_I$ does not choose the identity ID_l of the receiver during the challenge phase.
- E_2: a key extraction query is made on ID_l.
- E_3: Because of collision property of H_2, $\mathcal{B}$ aborts in a Signcryption query.
- E_4: At a particular instant of the game, $\mathcal{B}$ aborts Unsigncryption query since rejecting a valid ciphertext.

So the probability of the above invents are

- $\Pr[\neg E_1] = \frac{1}{q_{h_1}}$
- $\neg E_1 \Rightarrow \neg E_2$
- $\Pr[E_3] \leq \frac{q_{se}(q_{se}+q_{h_2})}{2^k}$
- $\Pr[E_4] \leq \frac{q_{us}}{2^k}$

Hence the probability of $\mathcal{B}$ does not abort is

$$\Pr[\neg \texttt{abort}] = \Pr[\neg E_1 \wedge \neg E_2 \wedge \neg E_3 \wedge \neg E_4]$$
$$\Rightarrow \Pr[\neg \texttt{abort}] \geq \frac{1}{q_{h_1}}\left(\frac{q_{se}(q_{se}+q_{h_2})}{2^k}\right)\left(\frac{q_{us}}{2^k}\right)$$

Further $\mathcal{B}$ selects the valid element from L_2 or L_3 with probability $\frac{1}{q_{h_1}(2q_{h_2}+q_{h_3})}$

$$\epsilon^* \geq \frac{\epsilon}{q_{h_1}(2q_{h_2}+q_{h_3})}\left(\frac{q_{se}(q_{se}+q_{h_2})}{2^k}\right)\left(\frac{q_{us}}{2^k}\right)$$

`Running time`
$\mathcal{B}$'s requires $\mathcal{O}(q^2_{h_1})$ scalar multiplications in $\mathbb{G}_1$ during the preparing phase, one scalar multiplication and inversion operation are required in signcryption oracle query. Hence the bound of $\mathcal{B}$'s running time is given by

$$t^* < t + \mathcal{O}(q_1^2 + q_{se})t_{sm} + \mathcal{O}(q_{se})t_{inv}$$

7 Conclusion

This article proposes a generic construction of Certificateless Signcryption Scheme(`CLSC`) which is provably secure in random oracle model. The scheme is proven to be satisfied confidentiality and unforgeability against chosen ciphertext and message attack of `Type-I` and `Type-II` in an adaptive manner respectively. Our scheme is more efficient in term of computational cost and secure than the schemes proposed by et al.. It is suited to implement on low power and processor devices such as PDA, smart phone, WSNs and smart card etc.

References

1. Y. Zheng, Digital signcryption or how to achieve cost (signature & encryption) $\ll$ cost (signature) + cost(encryption), In: Advances in Cryptology-CRYPTO'97, Lecture Notes in Computer Science, vol. 1294, Springer-Verlag, 1997, pp. 165–179.
2. A. Shamir, Identity-based cryptosystems and signature schemes, In: Advances in Cryptology-CRYPTO'84, Lecture Notes in Computer Science, vol. 196, Springer-Verlag, 1984, pp. 47–53.
3. A. Fiat, A. Shamir, How to prove yourself: practical solutions to identification and signature problems, In: Advances in Cryptology-CRYPTO'86, Lecture Notes in Computer Science, vol. 263, Springer-Verlag, 1986, pp. 186–194.
4. L. Guillou, J.J. Quisquater, A "Paradoxical" Identity-based signature scheme resulting from zero-knowledge, In: Advances in Cryptology-CRYPTO'88, Lecture Notes in Computer Science, vol. 403, Springer-Verlag, 1988, pp. 216–231.
5. A.W. Dent, A survey of certificateless encryption schemes and security models, International Journal of Information Security 7 (2008) 349–377.
6. P.S.L.M. Barreto, B. Libert, N. McCullagh, J.J. Quisquater, Efficient and provably-secure identity-based signatures and signcryption from bilinear maps, In: Advances in Cryptology-ASIACRYPT 2005, Lecture Notes in Computer Science, vol. 3788, Springer-Verlag, 2005, pp. 515–532.
7. S.S.M. Chow, C. Boyd, J.M.G. Nieto, Security-mediated certificateless cryptography, In: Public Key Cryptography-PKC 2006, Lecture Notes in Computer Science, vol. 3958, Springer-Verlag, 2006, pp. 508–524.
8. X. Boyen, Multipurpose identity-based signcryption: a swiss army knife for identity-based cryptography, In: Advances in Cryptology-CRYPTO 2003, Lecture Notes in Computer Science, vol. 2729, Springer-Verlag, 2003, pp. 383–399.
9. L. Chen, J. Malone-Lee, Improved identity-based signcryption, In: Public Key Cryptography-PKC 2005, Lecture Notes in Computer Science, vol. 3386, Springer-Verlag, 2005, pp. 362–379.
10. S.S.M. Chow, S.M. Yiu, L.C.K. Hui, K.P. Chow, Efficient forward and provably secure ID-based signcryption scheme with public verifiability and public ciphertext authenticity, In: Information Security and Cryptology-ICISC 2003, Lecture Notes in Computer Science, vol. 2971, Springer-Verlag, 2003, pp. 352–369.
11. B. Libert, J.J. Quisquater, A new identity based signcryption schemes from pairings, In: 2003 IEEE Information Theory Workshop, Paris, France, 2003, pp. 155–158.
12. J. Malone-Lee, Identity based signcryption, Cryptology ePrint Archive, Report 2002/098.
13. S.S. Al-Riyami, K.G. Paterson, Certificateless public key cryptography, In: Advances in Cryptology-ASIACRYPT 2003, Lecture Notes in Computer Science, vol. 2894, Springer-Verlag, 2003, pp. 452–474.
14. S.S.D. Selvi, S.S. Vivek, C.P. Rangan Certificateless KEM and hybrid signcryption schemes revisited, In: Information Security Practice and Experience-ISPEC 2010, Lecture Notes in Computer Science, vol. 6047, Springer-Verlag, 2010, pp. 294–307.
15. S.S.D. Selvi, S.S. Vivek, C.P. Rangan Cryptanalysis of Certificateless Signcryption Schemes and an Efficient Construction Without Pairing, Cryptology ePrint Archive: Report 2009/298, Available from: http://eprint.iacr.org/2009/298.pdf.
16. W. Xie and Z. Zhang Certificateless Signcryption without Pairing," Cryptology ePrint Archive: Report 2010/187, Available from: http://eprint.iacr.org/2010/187.pdf.
17. J. Baek, R. Safavi-Naini, and W. Susilo Certificateless public key encryption without pairing," Ptoc. Information Security, Springer-Verlag, LNCS, vol. 3650, 2005, pp. 134-148.
18. J. Malone-Lee, W. Mao, Two birds one stone: signcryption using RSA, In: Topics in Cryptology-CT-RSA 2003, Lecture Notes in Computer Science, vol. 2612, Springer-Verlag, 2003, pp. 211–226.
19. B. Libert, J.J. Quisquater, On constructing certificateless cryptosystems from identity based encryption, In: Public Key Cryptography-PKC 2006, Lecture Notes in Computer Science, vol. 3958, Springer-Verlag, 2006, pp. 474–490.
20. Shsfi Goldwasser, Silivio Micali and Ronald Rivest A digital signature scheme secure against adaptive chosen-message attacks SIAM Journal on Computing - Special issue on cryptography, 1988, Vol-17, issue-02, pp. 281–308
21. M. Barbosa, P. Farshim, Certificateless signcryption, In: ACM Symposium on Information, Computer and Communications Security-ASIACCS 2008, Tokyo, Japan, 2008, pp. 369–372.
22. Efficient and Provably secure identity-based signature and signcryption from bilinear maps. In Advanced in cryptology-ASIACRYPT 2005, LNCS Vol.3788, 2005, pp .515–532.
23. D. Boneh, M. Franklin, Identity-based encryption from the weil pairing, In: Advances in Cryptology-CRYPTO 2001, Lecture Notes in Computer Science, vol. 2139, Springer-Verlag, 2001, pp. 213–229.
24. Zhenhua Liu, Yupu Hu, Xiangsong Zhang, Hua Ma, "Certificateless signcryption scheme in the standard model," Information Sciences 180 (2010) 452-464.
25. Wenjian Xie and Zhang Zhang "Efficient and Provably Secure Certificateless Signcryption from Bilinear Maps", Available from eprint.iacr.org/2009/578.

26. Fagen Li, Masaaki Shirase, and Tsuyoshi Takagi, "Certificateless hybrid signcryption," In ISPEC, LNCS 5451, pages 112-123. Springer, 2009.
27. J.H. An, Y. Dodis, T. Rabin, On the security of joint signature and encryption, In: Advances in Cryptology-EUROCRYPT 2002, Lecture Notes in Computer Science, vol. 2332, Springer-Verlag, 2002, pp. 83–107.