Concurrent Blind Signatures without Random Oracles*

Aggelos Kiayias† Hong-Sheng Zhou†

Abstract

We present a blind signature scheme that is efficient and provably secure without random oracles under concurrent attacks utilizing only four moves of short communication. The scheme is based on elliptic curve groups for which a bilinear map exists and on extractable and equivocable commitments. The unforgeability of the employed signature scheme is guaranteed by the LRSW assumption while the blindness property of our scheme is guaranteed by the Decisional Linear Diffie-Hellman assumption.

We prove our construction secure under the above assumptions as well as Paillier's DCR assumption in the concurrent attack model of Juels, Luby and Ostrovsky from Crypto '97 using a common reference string. Our construction is the first efficient construction for blind signatures in such a concurrent model without random oracles. We present two variants of our basic protocol: first, a blind signature scheme where blindness still holds even if the public-key generation is maliciously controlled; second, a blind signature scheme that incorporates a "public-tagging" mechanism. This latter variant of our scheme gives rise to a partially blind signature with essentially the same efficiency and security properties as our basic scheme.

1 Introduction

Blind signatures were introduced by Chaum in [Cha82] and proved to be a most useful cryptographic scheme that has been the basis of many complex cryptographic constructions including e-cash systems and e-voting schemes. Informally, a blind signature is a signature scheme that incorporates a signing protocol that allows the signer to sign a document submitted by a user blindly, i.e., without obtaining any information about the document itself.

It was observed early on (at least as early as [Dam88], see also [PW91]) that blind signatures contain an instance of a secure function evaluation protocol in the following sense: the user possesses a private input m and a public-input pk which is the verification key of a digital signature algorithm, and the signer possesses a private input sk which is the signing-key of the digital signature algorithm; with this setup the user and the signer should execute a probabilistic secure function evaluation protocol that will allow the user to compute σ, a signature on m under pk, without revealing m to the signer and without the signer revealing sk to the user. Given the complexity of general secure function evaluation though, [Yao86, GMW87], in early work on blind signatures this paradigm was not very motivating. A more motivating paradigm was found in divertible zero-knowledge proofs [OO89, Oka92, CDP94] and many blind signatures were subsequently designed in this line of reasoning [PS96, PS97, Poi98, AO00, AO01, Abe01] as well as the first attempt to give provably secure constructions (in the random oracle model) was due to [PS96].

Regarding provably secure constructions, Pointcheval and Stern [PS96], presented secure blind signatures with three communication moves that were proven secure in the random oracle model under the

*An earlier version of this paper was titled "Two-round Concurrent Blind Signatures without Random Oracles" with each round meant to include two moves; this proved to be confusing with respect to the use of the term "round" in previous works and thus the "two-round" was removed from the title. The protocols presented in all versions of the present work have always been 4-move protocols.

†University of Connecticut, Computer Science and Engineering, Storrs, CT, USA, {aggelos,hszhou}@cse.uconn.edu.
Research partly supported by NSF CAREER Award CNS-0447808.

discrete-logarithm assumption assuming only logarithmically many messages were transmitted by the user. This result was later improved to polynomially many messages but five communication moves [Poi98] and the round complexity was finally decreased to three moves and polynomially many messages in [AO01, Abe01]. A two moves protocol was presented in [BNPS01] assuming the RSA inversion oracle assumption. We stress that all these results were proven secure in the random oracle model.

Concurrency in the context of blind signatures was put forth by Juels, Luby and Ostrovsky [JLO97] who presented the first security model for blind signatures that takes into account that the adversary may launch many concurrent sessions of the blind signing protocol (operating as either the user or the signer). Concurrency is particularly important since in implementations of blind signatures in e-voting and e-cash schemes, see e.g., [Cha82, FOO92, Kim04], the signer is a multi-threaded server that accepts many concurrent sessions of users that are executing the signing protocol. Thus, it is of crucial importance to consider the security of blind signatures, when *(1)* a malicious signer attempts to defeat the blindness of many concurrently joining users, and *(2)* a coalition of malicious users attempts to extract information about the signing key of the multi-threaded signer server. Still, the design of schemes that satisfied such stronger models proved elusive. In fact, Lindell [Lin03] showed that concurrent security for blind signatures is impossible in the bare model (i.e., without any setup assumption). On the other hand, in the CRS model, Canetti et al. [CLOS02] gave a generic construction for multi-party secure function evaluation that achieves an even stronger notion of security than concurrency (universal composition) and can be used to solve (generically) the blind signature problem using a CRS. Note that this construction is not efficient and some trusted setup assumption such as using a CRS is necessary for a blind signature given the result of Lindell [Lin03]. More recently, Camenisch et al. [CKW04] using a weaker model than that of [JLO97] that only allowed sequential attacks presented an eight-move blind signature scheme that is based on the Strong-RSA assumption leaving as open problem the possibility of achieving concurrent security in an efficient scheme.

Our Contribution. In this paper, we give the first efficient construction for blind signatures to achieve concurrent security in the sense of [JLO97] assuming a common reference string. The four-move interactions between the user and the signer in the signing protocol requires overall communication not exceeding 2 Kbytes (about 10.2 Kbits to be precise) for a full signature generation. Achieving this level of efficiency while simultaneously maintaining provability in a concurrency model required the careful composition of a number of cryptographic primitives. As our underlying digital signature scheme (i.e., the type of signature that is obtained by users) we use the elliptic curve based signature scheme of Camenisch and Lysyanskaya [CL04] (henceforth called a CL signature). We also employ a variant of Linear Encryption, an encryption scheme that was originally introduced in the context of group signatures by Boneh, Boyen and Shacham [BBS04]. Here we find a novel use of this primitive in the context of blind signatures. In addition to these primitives, our construction makes essential use of discrete-logarithm equivocal commitments based on Pedersen commitments [Ped91] and extractable commitments based on Paillier encryption [Pai99].

The central idea of our construction is to use a variant of Linear Encryption to produce a very efficient secure function evaluation protocol for CL signatures that proceeds roughly as follows: the user selects on the fly a key for the encryption scheme and encrypts her message with it. The signer upon receiving this encryption takes advantage of the homomorphic properties of the encryption to blindly transform the ciphertext into a randomized encryption of a CL signature and then transmits the resulting rerandomized ciphertext back to the user. We make an essential use of the homomorphic properties of the underlying encryption in the efficient generation of non-adversarial randomness between the mutually distrustful players.

In order to prove security under concurrent attacks a number of provisions have to be taken in the blind signature protocol design. Most importantly, in our signing protocol, both sides will be required to prove statements about their local computations. As a result, performing the whole protocol in four moves is one of the most delicate parts of our construction. The homomorphic encryption based interaction that is used for the secure signature computation needs to be paired with an extractable commitment. Moreover, an

equivocable commitment is used for ensuring that no information leakage occurs from the user to the signer or vice versa. Finally, the signer, proves to the user that he is following the protocol specifications and is applying his signing key to the user's ciphertext whereas the user has to prove that he is consistent across his commitments.

The construction is proven to satisfy the two properties of the [JLO97] model as follows: the blindness property is ensured under the Decisional Composite Residuosity assumption of [Pai99] and the Decision Linear Diffie-Hellman assumption of [BBS04]. The unforgeability property is proven under the LRSW assumption of [LRSW99]. Note that the resulting signature from the signing protocol is about half the size of an RSA based Chaum blind signature.

Stronger blindness property. We consider a stronger adversarial model for blindness where the public-key is adversarially controlled; we show how it is possible to modify our basic protocol in a straightforward way to achieve this stronger blindness property.

Public-tagging and partial blindness. We finally provide an extension of our scheme that allows the public-tagging of blindly signed messages, i.e., all messages that are obtained by the users also contain a publicly known tag that is decided prior to the signing protocol execution. This extension is essentially equivalent to a partially blind signature construction, a notion that was formalized in [AF96]. In a partially blind signature every message is tagged with a public-string that is produced jointly by the user and the signer. The blindness property is then restricted to hold only for blind signatures with same tag. Partial blindness is important as it allows the signer to reuse the same public-key for a variety of different blind signature functions.

2 Preliminaries

Bilinear Groups. Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p such that $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear map, i.e., for all $t, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, it holds that $e(t^a, v^b) = e(t, v)^{ab}$ and e is non-trivial, i.e., $e(g, g) \neq 1$. Note that $|\mathbb{G}_T| = p$.

Camenisch-Lysyanskaya Signature. Camenisch and Lysyanskaya [CL04] proposed a digital signature scheme (which we will call it CL-signature for short) that was adaptively chosen message secure in the standard model. Our blind signature will be based on this signature scheme and we describe it below:

- The key generation algorithm $\texttt{gen}^{CL}$: generate the bilinear group parameter $(p, \mathbb{G}, \mathbb{G}_T, g, e)$; then choose $x, y \xleftarrow{r} \mathbb{Z}_p^*$, and compute $X = g^x$ and $Y = g^y$; set secret key as $sk = (x, y)$ and public key as $pk = (p, \mathbb{G}, \mathbb{G}_T, g, e; X, Y)$.
- The signing algorithm $\texttt{sign}^{CL}$: on input message m, secret key $sk = (x, y)$, and public key $pk = (p, \mathbb{G}, \mathbb{G}_T, g, e; X, Y)$, choose a random $a \in \mathbb{G}$, and output the signature $\sigma = (a, a^y, a^{x+mxy})$.
- The verification algorithm $\texttt{verify}^{CL}$: on input public key $pk = (p, \mathbb{G}, \mathbb{G}_T, g, e; X, Y)$, message m, and signature $\sigma = (a, b, c)$, check whether the verification equations $e(a, Y) = e(g, b)$ and $e(X, a)e(X, b)^m = e(g, c)$ hold.

The underlying assumption of CL-signatures is called the LRSW assumption, which was introduced by Lysyanskaya et al. [LRSW99]. Note that in this paper it was also shown that this assumption holds for generic groups.

Assumption 2.1 (LRSW Assumption). Given the bilinear group parameters $(p, g, \mathbb{G}, \mathbb{G}_T, e)$. Let $X, Y \in \mathbb{G}, X = g^x, Y = g^y$ and define $O_{X,Y}()$ to be an oracle that, on input a value $m \in \mathbb{Z}_p$, it outputs a triple (a, b, c) such that $b = a^y$, and $c = a^{x+mxy}$ where $a \xleftarrow{r} \mathbb{G}$. Then, for all probabilistic polynomial time adversaries $\mathcal{A}$,

$$\Pr\left[\begin{array}{l} x, y \in \mathbb{Z}_p; X = g^x; Y = g^y; (m, a, b, c) \leftarrow \mathcal{A}^{O_{X,Y}} : \\ \quad m \notin \mathsf{Q} \wedge m \in \mathbb{Z}_p \wedge m \neq 0 \wedge a \in \mathbb{G} \wedge b = a^y \wedge c = a^{x+mxy} \end{array}\right] \leq \epsilon$$

where ϵ is a negligible function in security parameter λ, and Q is the set of queries that $\mathcal{A}$ made to $O_{X,Y}()$.

Linear Encryption. Boneh et al. [BBS04] proposed a variant of ElGamal encryption, called, Linear Encryption that is suitable for groups over which the DDH assumption fails. We call it LE for short.

- The key generation algorithm $\mathtt{gen}^{LE}$: the public key pk is a triple of generators $t, v, w \in \mathbb{G}$ and the secret key sk is the exponents $x, y \in \mathbb{Z}_p^*$ such that $t^x = v^y = w$.
- The encryption algorithm $\mathtt{enc}^{LE}$: to encrypt a message $m \in \mathbb{G}$, choose random values $a, b \in \mathbb{Z}_p$, and output the triple $(t^a, v^b, m \cdot w^{a+b})$.
- The decryption algorithm $\mathtt{dec}^{LE}$: given an encryption (T, V, W), we recover the plaintext m as follows $m = \mathtt{dec}_{sk}^{LE}(T, V, W) = \frac{W}{T^x \cdot V^y}$.

The Linear encryption is based on the Decision Linear Diffie-Hellman assumption, which was first introduced by Boneh et al. [BBS04]. With $g \in \mathbb{G}$ as above, along with arbitrary generators t,v, and w of G, consider the following problem:

Definition 2.2 (Decision Linear Diffie-Hellman Problem in $\mathbb{G}$). Given $t, v, w, t^\alpha, v^\beta, w^\gamma \in \mathbb{G}$ as input, output 1 if $\alpha + \beta = \gamma$ and 0 otherwise.

It is believed that DLDH is a hard problem even in bilinear groups where DDH is easy. Now we define the advantage of an algorithm $\mathcal{A}$ in deciding the DLDH problem in $\mathbb{G}$ as

$$\mathsf{Adv}_{\mathsf{DLDH}}^{\mathcal{A}} = \left| \begin{array}{l} \Pr[1 \leftarrow \mathcal{A}(t, v, w, t^\alpha, v^\beta, w^{\alpha+\beta}) : t, v, w \in \mathbb{G}, \alpha, \beta \in \mathbb{Z}_p] \\ \quad - \Pr[1 \leftarrow \mathcal{A}(t, v, w, t^\alpha, v^\beta, \chi) : t, v, w, \chi, \in \mathbb{G}, \alpha, \beta \in \mathbb{Z}_p] \end{array} \right|$$

Assumption 2.3 (Decision Linear Diffie-Hellman Assumption). We say that the Decision Linear Diffie-Hellman assumption holds in $\mathbb{G}$ if for all PPT algorithms $\mathcal{A}$ it holds that $\mathsf{Adv}_{\mathsf{DLDH}}^{\mathcal{A}}$ is negligible in the security parameter λ.

Paillier-Encryption. In our scheme we will employ the public-key encryption introduced by Paillier [Pai99]:

- The key generation algorithm $\mathtt{gen}^{Pai}$: let p and q be random primes for which it holds $\mathsf{p} \neq \mathsf{q}$, $|\mathsf{p}| = |\mathsf{q}|$ and $\gcd(\mathsf{pq}, (\mathsf{p}-1)(\mathsf{q}-1)) = 1$; let $\mathsf{n} = \mathsf{pq}$, $\pi = \mathrm{lcm}(\mathsf{p}-1, \mathsf{q}-1)$, $K = \pi^{-1} \bmod \mathsf{n}$, and $\mathsf{g} = (1 + \mathsf{n})$; the public key is $pk = (\mathsf{n}, \mathsf{g})$ while the secret key is $sk = (\mathsf{p}, \mathsf{q})$.
- The encryption algorithm $\mathtt{enc}^{Pai}$: the plaintext set is $\mathbb{Z}_\mathsf{n}$; given a plaintext m, choose a random $\zeta \in \mathbb{Z}_\mathsf{n}^*$, and let the ciphertext be $E_m = \mathtt{enc}_{pk}^{Pai}(m, \zeta) = \mathsf{g}^m \zeta^\mathsf{n} \bmod \mathsf{n}^2$.
- The decryption algorithm $\mathtt{dec}^{Pai}$: given a ciphertext E_m, let $K = \pi^{-1} \bmod \mathsf{n}$ and now observe that $(E_m)^{\pi K} = \mathsf{g}^{m \cdot \pi K} \cdot \zeta^{\mathsf{n} \cdot \pi K} = \mathsf{g}^{m \cdot \pi K \bmod \mathsf{n}} \cdot \zeta^{\mathsf{n} \cdot \pi K \bmod \mathsf{n}\pi} = \mathsf{g}^{m \bmod \mathsf{n}} \cdot \zeta^{0 \bmod \mathsf{n}\pi} = \mathsf{g}^m = 1 + m\mathsf{n} \bmod \mathsf{n}^2$. Thus, it is possible to recover $m = \frac{((E_m)^{\pi K} \bmod \mathsf{n}^2) - 1}{\mathsf{n}} \bmod \mathsf{n}$.

The cryptosystem above has been proven semantically secure if and only if the Decisional Composite Residuosity (DCR) assumption [Pai99] is true. The advantage of an algorithm $\mathcal{A}$ in deciding the DCR problem is defined as follows:

$$\mathsf{Adv}_{\mathsf{DCR}}^{\mathcal{A}} = \left| \Pr[1 \leftarrow \mathcal{A}(z) : z \in \mathbb{Z}_{\mathsf{n}^2}^*] - \Pr[1 \leftarrow \mathcal{A}(z) : z \in HR_{\mathsf{n}^2}^\mathsf{n}] \right|$$

where $HR_{\mathsf{n}^2}^\mathsf{n}$ is the subgroup of n-th residues modulo n^2.

Assumption 2.4 (Decisional Composite Residuosity Assumption). We say that the DCR assumption holds in $\mathbb{G}$ if for all PPT algorithms $\mathcal{A}$ it holds that $\mathsf{Adv}_{\mathsf{DCR}}^{\mathcal{A}}$ is negligible in the security parameter λ.

Commitment Schemes. A commitment scheme is a protocol with two stages, the commit stage and the decommit stage, between two parties, the committer and the receiver. A commitment scheme consists of a key generation algorithm $\mathtt{gen}$ which can be used to produce a public key pk, a commitment algorithm $\mathtt{com}$ which is used by the committer to produce a commitment to the message m and the decommitment information ζ, i.e., $(c, \zeta) \leftarrow \mathtt{com}_{pk}(m)$, and a decommitment verification algorithm $\mathtt{dec}$ which can be used by the receiver to verify the decommitment information ζ and the message m with respect to the commitment c, i.e., $\mathtt{dec}(c, m, \zeta) \in \{0, 1\}$. Frequently the decommitment information ζ is the random coins used by the commitment algorithm and we will write $c \leftarrow \mathtt{com}_{pk}(m, \zeta)$.

A commitment scheme will satisfy two properties: *hiding*, the receiver can not obtain any information about m given $\mathtt{com}_{pk}(m, \zeta)$; and *binding*, the committer cannot change his mind about m later, i.e. he cannot change the decommitment verification information (m, ζ) into some (m', ζ') where $m \neq m'$, so that $c \leftarrow \mathtt{com}_{pk}(m, \zeta)$ and $\mathtt{dec}(c, m', \zeta') = 1$.

In an *extractable* commitment, there is a trapdoor information xk associated to each public key pk that allows the trapdoor owner to compute m from any $\mathtt{com}_{pk}(m, \zeta)$. In an *equivocable* commitment on the other hand, there is a trapdoor information ek associated to each public key pk that allows a committer who is a trapdoor owner to compute ζ' given any $m, \zeta, m', c \leftarrow \mathtt{com}_{pk}(m, \zeta)$ so that $\mathtt{dec}(c, m', \zeta') = 1$.

Common Reference String Model. In the common reference string (CRS) model, we assume that each player can access a common string that is guaranteed to come from a prescribed distribution. Furthermore, no players (including the adversaries) will know the trapdoor information related to the procedure of choosing the string. The trapdoor will be known to the simulator in the proof of security. In practice, a trusted third party can generate the CRS by running the CRS generator $\mathtt{K}$, i.e. $(\mathtt{crs}, \tau) \leftarrow \mathtt{K}(1^\lambda)$, and discarding the trapdoor τ. The string $\mathtt{crs}$ is published, and all parties receive it as additional input.

3 Formal Model for Blind Signatures

In this section, we revisit in detail the formal model for blind signatures as introduced in [JLO97] and we reformulate it to the common reference string (CRS) model. We stress again that some trusted setup assumption is necessary in the light of Lindell's negative result for blind signatures [Lin03] in the "bare" concurrent model.

3.1 Blind Signature Scheme

Definition 3.1 (Blind Signature Scheme). A blind digital signature scheme is a four-tuple, consisting of two interactive Turing machines (S, U) and two algorithms ($\mathtt{gen}$,$\mathtt{verify}$). Here S denotes the signer, and U the user.

- $\mathtt{gen}(1^\lambda)$ is a probabilistic polynomial time key-generation algorithm which takes as an input a security parameter 1^λ and outputs a pair (pk, sk) of public and secret keys.
- $\mathsf{S}(pk, sk)$ and $\mathsf{U}(pk, m)$ is a pair of polynomially time bounded probabilistic interactive Turing machines, where both machines have the following tapes: read-only input tape, write-only output tape, a read/write work tape, a read-only random tape, and two communication tapes, a read-only and a write-only tape. They are both given on their input tapes as a common input a pk produced by the key generation algorithm. Additionally S is given on his input tape the corresponding secret key sk and U is given on his input tape a message m, where the length of all inputs must be polynomial in the security parameter 1^λ. Both U and S engage in an interactive protocol for some polynomial in λ number of moves. At the end of this protocol S outputs either *completed* or *not-completed* and U outputs either σ or $\perp$.
- $\mathtt{verify}(m, \sigma, pk)$ is a deterministic polynomial time algorithm, which outputs 1 or 0.

The correctness requirement for the above is that for any message m, and for all random choices of the key generation algorithm, if both S and U follow the protocol then S always outputs *completed*, and if the output of the user is σ then $\texttt{verify}(m, \sigma, pk) = 1$.

Note that in the CRS model, both S, U receive as additional input the $\texttt{crs}$ string.

3.2 Blindness and Unforgeability

The security properties for blind signatures defined in [JLO97] are **blindness** and **unforgeability**. Below we revisit their modelling and we give detailed definitions for these properties in the CRS model.

Definition 3.2 (Blindness). Assume $(\texttt{crs}, \tau) \leftarrow \mathtt{K}(1^\lambda)$, $(pk, sk) \leftarrow \mathtt{gen}(1^\lambda)$. We define an oracle $\mathcal{I}^\phi$ with public input $(1^\lambda, \texttt{crs}, pk)$ which simulates two user instantiations U^L and U^R, where $\phi \in \{0, 1\}$. The adversary $\mathcal{A}$ will be communicating with this oracle trying to predict ϕ given input $(1^\lambda, \texttt{crs}, pk, sk)$. The oracle $\mathcal{I}^\phi$ operates as follows:

- Given $\langle \texttt{challenge}, m_0, m_1 \rangle$, the oracle $\mathcal{I}^\phi$ simulates two user instantiations U^L and U^R with input the public-key pk and the messages m_ϕ and $m_{1-\phi}$ respectively. The oracle $\mathcal{I}^\phi$ keeps a database with the state of each user instantiation; the state includes all coin tosses of the user instantiation and the contents of all tapes including the communication tape. The oracle uses st^L (resp. st^R) to record the state of U^L (resp. U^R).
- Given $\langle \texttt{advance}, \rho, msg \rangle$, where $\rho \in \{L, R\}$, the oracle $\mathcal{I}^\phi$ recovers the state of st^ρ, and simulates the user instantiation U^ρ with msg till U^ρ either terminates or returns a response to the signer. If U^ρ returns a response, then $\mathcal{I}^\phi$ returns this to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^\rho = st^\rho || st$. Note that this kind of query can be executed several times depending on the number of moves of the blind signature protocol.
- Given $\langle \texttt{terminate}, msg^L, msg^R \rangle$, the oracle $\mathcal{I}^\phi$ recovers the state st^L (resp. st^R), and simulates the user instantiation U^L (resp. U^R) with msg^L (resp. msg^R) till U^L (resp. U^R) terminates or fails. If both user instantiations terminate successfully and output two signatures, then the oracle returns these signatures to $\mathcal{A}$, otherwise returns $(\perp, \perp)$.

Given any probabilistic polynomial time $\mathcal{A}$, we define its advantage against blindness as:

$$\mathsf{Adv}^{\mathcal{A}}_{\mathsf{blind}}(\lambda) = \left| \Pr \left[\begin{array}{l} \phi \leftarrow \mathcal{A}^{\mathcal{I}^\phi(1^\lambda, \texttt{crs}, pk)}(1^\lambda, \texttt{crs}, pk, sk) \ : \\ \qquad \phi \xleftarrow{\mathtt{r}} \{0,1\}, (\texttt{crs}, \tau) \leftarrow \mathtt{K}(1^\lambda), (pk, sk) \leftarrow \mathtt{gen}(1^\lambda) \end{array} \right] - \frac{1}{2} \right|$$

and say that the blind signature scheme satisfies the blindness property if $\mathsf{Adv}^{\mathcal{A}}_{\mathsf{blind}}(\lambda)$ is negligible in λ.

Definition 3.3 (Unforgeability). We define an oracle $\mathcal{I}$ that is simulating concurrently an arbitrary number of signer instantiations. The oracle accepts two types of queries defined as follows:

- $\langle \texttt{start}, msg \rangle$. The oracle $\mathcal{I}$ selects a session identifier sid, and simulates the signer instantiation S with msg till S either terminates or returns a response. If the signer instance returns a response to the user, $\mathcal{I}$ returns this with the session identifier sid as an answer to the oracle query. The oracle $\mathcal{I}$ keeps a database with the state of S for the session identifier sid; the state includes all coin tosses of S, and the contents of all tapes including the communication tape.
- $\langle \texttt{advance}, sid, msg \rangle$. The oracle $\mathcal{I}$ looks up the table of sessions and recovers the state of S for the session with identifier sid (if session sid exists). Subsequently, $\mathcal{I}$ writes msg in the communication tape of S and simulates it till it either terminates or returns a response to the user. If it returns a message to the user, $\mathcal{I}$ returns this as an answer to the oracle query. If no session identifier exists the oracle returns "fail."

The oracle $\mathcal{I}$ maintains a counter ℓ that counts the number of times that the oracle has successfully terminated a signer session. Each time that $\mathcal{I}$ successfully terminates a signer session it increases the counter ℓ by 1. A "one-more forgery" adversary against the blind signature is a polynomial-time probabilistic machine $\mathcal{A}$ that is given as input $(1^\lambda, \mathtt{crs}, pk)$ where $(\mathtt{crs}, \tau) \leftarrow \mathtt{K}(1^\lambda)$ and $(pk, sk) \leftarrow \mathtt{gen}(1^\lambda)$. The adversary $\mathcal{A}$ interacts with $\mathcal{I}(\mathtt{crs}, pk, sk)$ and terminates by returning a sequence of $(m_1, \sigma_1), ..., (m_{\ell'}, \sigma_{\ell'})$ where $m_i \neq m_j$ for all $i, j : 1 \leq i \neq j \leq \ell'$. We define the advantage of $\mathcal{A}$ in the above attack by

$$\mathsf{Adv}^{\mathcal{A}}_{\mathsf{unforge}}(\lambda) = \Pr[\wedge_{i=1}^{\ell'}(1 \leftarrow \mathtt{verify}(pk, m_i, \sigma_i)) \wedge (\ell' > \ell)]$$

and say that the blind signature scheme is unforgeable if $\mathsf{Adv}^{\mathcal{A}}_{\mathsf{unforge}}(\lambda)$ is negligible in λ.

4 The Proposed Scheme

4.1 Setup and Generation of Keys

We start the description of our construction by describing the setup definition as well as the way that the involved parties, the user and the signer generate their keys.

Public Parameters. The public parameter $\mathtt{pub}$ contains general information about all protocol executions as well as a specific bilinear group parameter $(p, \mathbb{G}, \mathbb{G}_T, g, e)$ appropriately selected.

Common Reference String. Next we describe how the common reference string $\mathtt{crs}$ is selected. It includes two parts, $\mathtt{crs}_1$ and $\mathtt{crs}_2$. First, we generate parameters for a Pedersen-like [Ped91] commitment scheme over an elliptic curve group: let $\mathbf{G} = \langle \mathbf{g} \rangle$ be a cyclic elliptic curve group of prime order Q; select $r \xleftarrow{\mathtt{r}} \mathbb{Z}^*_Q$ and compute $\mathbf{h} = \mathbf{g}^r$; set $\mathtt{crs}_1 = \langle Q, \mathbf{g}, \mathbf{h}, \mathbf{G}, \mathcal{H} \rangle$, where $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{Z}_Q$ is a collision resistant hash function and set the trapdoor to be $\tau_1 = r$. Then we generate parameters for the Paillier encryption: let p and q be random primes for which it holds $\mathsf{p} \neq \mathsf{q}$, $|\mathsf{p}| = |\mathsf{q}|$ and $\gcd(\mathsf{pq}, (\mathsf{p}-1)(\mathsf{q}-1)) = 1$; let $\mathsf{n} = \mathsf{pq}$, and $\mathsf{g} = (1+\mathsf{n})$; set $\mathtt{crs}_2 = \langle \mathsf{n}, \mathsf{g} \rangle$ and the trapdoor $\tau_2 = \langle \mathsf{p}, \mathsf{q} \rangle$. Now we have $\mathtt{crs} = (\mathtt{crs}_1, \mathtt{crs}_2)$; the two trapdoors τ_1, τ_2 as well as any random coins used for the generation of $\mathtt{crs}$ are discarded.

Signer Parameters. The signer S uses the algorithm $\mathtt{gen}$ to generate his public and secret parameters based on $\mathtt{pub}$. The signer selects $x, y \xleftarrow{\mathtt{r}} \mathbb{Z}^*_p$ and computes $X = g^x$ and $Y = g^y$. Then it sets $PK_{\mathsf{S}} = \langle X, Y \rangle$ and $SK_{\mathsf{S}} = \langle x, y \rangle$; this is the key pair of S.

We note that the parameters selected above are assumed to be long-lived, i.e., they will be used for many executions of the signing protocol. On the other hand, the user has no long-lived parameters. Still, as part of each signing protocol the user will select some public and secret key that will have the lifetime of one signing protocol execution. We stress that this is not a necessity and each user may also keep his public-key parameters the same across signing protocol executions; in fact these parameters can be part of a PKI that all users are members of. This will make the protocol's time-complexity somewhat more efficient on the side of the user (but will have the cost of maintaining a user PKI).

User Parameters. Each user U generates his key pair on the fly: he selects $w \xleftarrow{\mathtt{r}} \mathbb{G} \backslash \{1\}$ and $\delta, \xi \xleftarrow{\mathtt{r}} \mathbb{Z}^*_p$, and set $t, v \in \mathbb{G}$ such that $t^\delta = v^\xi = w$. Set $PK_{\mathsf{U}} = \langle t, v, w \rangle$ as his public key and keep secretly $SK_{\mathsf{U}} = \langle \delta, \xi \rangle$ as his secret key.

Choice of Parameter Lengths. The length of each parameter p, n, Q is ν_p, ν_{n}, ν_Q respectively and should be selected so that the following are satisfied: (i) The DLDH assumption holds over the bilinear group parameter $(p, \mathbb{G}, \mathbb{G}_T, g, e)$, (ii) The LSRW assumption holds over the bilinear group parameter $(p, \mathbb{G}, \mathbb{G}_T, g, e)$, (iii) The discrete-logarithm (DLOG) assumption holds over the elliptic curve cyclic group $\mathbf{G}$, (iv) The DCR assumption holds over $\mathbb{Z}^*_{\mathsf{n}^2}$. Based on the present state of the art with respect to the solvability of the above problems, a possible choice of the parameters is for example $\nu_p = 171$ bits, $\nu_{\mathsf{n}} = 1024$ bits, $\nu_Q = 171$ bits.

4.2 Signing Protocol

We give a high-level description of our protocol before presenting in detail.
(1) First, both the user and the signer obtain the public inputs $\mathtt{pub}$, $\mathtt{crs}$, and PK_{S}, the signer gets the private input SK_{S}, and the user gets the private input message m.
(2) Then the user generates his key pair $(PK_{\mathsf{U}}, SK_{\mathsf{U}})$ for Linear Encryption, and keeps SK_{U} secret; the user generates a Paillier ciphertext for message m which is used as an extractable commitment; the user generates a special Linear Encryption ciphertext for m which will be signed by the signer.
(3) To guarantee that the Linear Encryption ciphertext and the Paillier ciphertext are consistent, the user interleaves within the protocol execution a 3-move Σ-protocol that shows the consistency of the commitment and the encryption. This protocol employs an equivocal Pedersen commitment scheme to allow zero-knowledge in the concurrent setting (cf. [Dam00]). When the signer successfully verifies the 3-move protocol which was initialized by the user, he will transform the Linear Encryption ciphertext by using his signing key SK_{S} and appropriately rerandomize it. This will result in the encryption of a CL-signature which will be recovered by the user using his secret key SK_{U}.
(4) To guarantee that the signer follows the protocol specifications, the signer is required to interleave a 3-move Σ-protocol as well in order to show that he is applying his secret-key appropriately on the Linear Encryption ciphertext that is provided by the user. Again we employ an equivocal Pedersen commitment to allow for concurrent zero-knowledge.
(5) When the user verifies successfully the final step of the signing protocol computation, he decrypts the CL-signature from the signer's ciphertext using his secret-key SK_{U} and obtains a CL-signature for the message m. Then he refreshes the randomness of the signature taking advantage of the randomness homomorphic property of CL-signatures.

Σ-protocols and Round-complexity. In our signing protocol we employ two Σ-protocols from both sides of the interaction. Both these protocols have the form $\langle commitment; challenge; response, decommitment\rangle$. A subtle difficulty in the design of our protocol is that if the two Σ-protocols are executed sequentially they will result in an overall round complexity of six moves. In order to maintain the four-move protocol complexity we want to "start" the Σ-protocol for the signer side before the user side Σ-protocol terminates. Nevertheless this will violate the security property of our scheme, so, in order to allow an early start of the signer side Σ-protocol we have the signer commit to the value he will prove a statement about and open the commitment *only in case* the user's side Σ-protocol verifies.

We outline the high-level description of our signing protocol in Figure 1. In the first step, the user U prepares two different encryptions of his private input m, called E_m and $\langle T, V, W\rangle$. Moreover, it computes the first move of a Σ-protocol that shows the consistency of the two encryptions and commits to it into $commitment_{\mathsf{U}}$. In the second step, the signer prepares an encryption ψ that can be decrypted by the user into a CL-signature but does not transmit yet this value to the user. Instead, it prepares the first move of a Σ-protocol that shows that he computed ψ correctly and commits to ψ as well as the first move into $commitment_{\mathsf{S}}$. In the third step, the user, given the challenge of the signer, completes the Σ-protocol that shows he computed the two encryptions E_m and $\langle T, V, W\rangle$ in a consistent way and transmits to the signer the decommitment information necessary to verify the consistency of the ciphertexts. In the fourth step, the signer verifies the Σ-protocol of the user and if it is accepted, the signer completes his Σ-protocol and transmits to the user the encryption ψ as well as the decommitment information necessary to verify the claim that ψ is correctly computed based on the signer's public-key. Finally the user verifies the Σ-protocol and if accepted it outputs the computed blind signature.

The detailed description of the protocol is shown in Figure 2. Note that $d_1 < p$, $d_2 < p$, i.e. $\lambda_1 < \nu_p$, $\lambda_2 < \nu_p$. For example $\lambda_0 = \lambda_1 = \lambda_2 = 80$ bits.

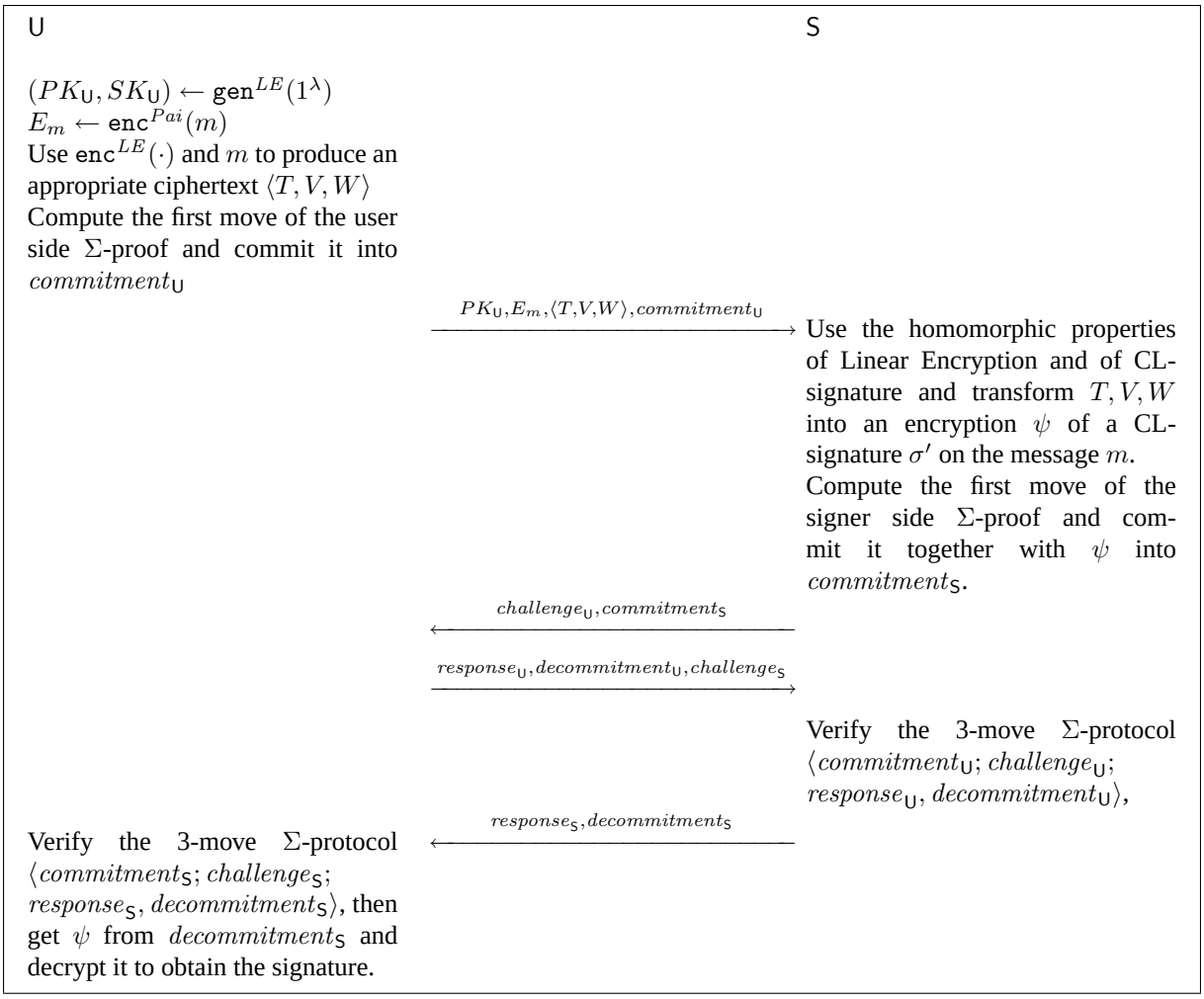

Figure 1: Overview of our blind signature generation protocol.

4.3 Signature Verification

Given a message-signature pair $(m; \sigma)$, where $\sigma = \langle a, b, c\rangle$, the verification algorithm is based on the two verification equations below: $e(a, Y) = e(g, b)$ and $e(X, a)e(X, b)^m = e(g, c)$.

4.4 Correctness and Security

The correctness and security of our scheme is captured by Theorem 4.1, Theorem 4.3, Theorem 4.5 as described here.

4.4.1 Correctness

Theorem 4.1 (Correctness). *If the signer and the user follow the signing protocol, the resulting signature satisfies the verification with provability 1.*

Proof. First, we check the correctness of the verification equations for the Σ-protocols.

$\mathtt{crs} = \langle Q, \mathbf{g}, \mathbf{h}, \mathbf{G}, \mathcal{H}; \mathsf{n}, \mathsf{g}\rangle; \mathtt{pub} = \langle p, g, \mathbb{G}, \mathbb{G}_T, e\rangle; PK_{\mathsf{S}} = \langle X, Y\rangle$

$\boxed{\mathsf{U}}$		$\boxed{\mathsf{S}}$
$MSG = \langle m\rangle, m \in [0, 2^{\nu_p}]$		$SK_{\mathsf{S}} = \langle x, y\rangle$
$(PK_{\mathsf{U}}, SK_{\mathsf{U}}) \leftarrow \mathtt{gen}^{LE}(1^\lambda)$ $PK_{\mathsf{U}} = \langle t, v, w\rangle, SK_{\mathsf{U}} = \langle \delta, \xi\rangle$		
$\widehat{m} \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}], A_m, B_m \xleftarrow{\mathtt{r}} \mathbb{Z}^*_{\mathsf{n}}$ $\alpha, k, l, \widehat{k}, \widehat{l} \xleftarrow{\mathtt{r}} \mathbb{Z}_p, \theta \xleftarrow{\mathtt{r}} \mathbb{G}\backslash\{1\}, \mu_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$ $E_m = \mathsf{g}^m (A_m)^{\mathsf{n}} \bmod \mathsf{n}^2$ $\widehat{E}_m = \mathsf{g}^{\widehat{m}} (B_m)^{\mathsf{n}} \bmod \mathsf{n}^2$ $T = t^k, V = v^l, W = \theta^m w^{k+l}$ $\widehat{T} = t^{\widehat{k}}, \widehat{V} = v^{\widehat{l}}, \widehat{W} = \theta^{\widehat{m}} w^{\widehat{k}+\widehat{l}}$		
$\omega_1 = \mathcal{H}(\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}), C_1 = \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$	$\xrightarrow{PK_{\mathsf{U}}, E_m, \langle\theta, T, V, W\rangle, C_1}$	$d_1 \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_1}$ $\alpha', k', l', \widehat{x}, \widehat{k}', \widehat{l}' \xleftarrow{\mathtt{r}} \mathbb{Z}_p, \mu_2 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$ $a' = \theta^{\alpha'}, b' = \theta^{y\alpha'}$ $T' = T^{xy\alpha'} t^{k'\alpha'}, V' = V^{xy\alpha'} v^{l'\alpha'}$ $W' = W^{xy\alpha'} \theta^{x\alpha'} w^{k'\alpha'+l'\alpha'}$ $L_T = e(T, b')^{\widehat{x}} e(t, a')^{\widehat{k}'}$ $L_V = e(V, b')^{\widehat{x}} e(v, a')^{\widehat{l}'}$ $L_W = (e(W, b')e(\theta, a'))^{\widehat{x}}\, e(w, a')^{\widehat{k}'+\widehat{l}'}$ $\omega_2 = \mathcal{H}(a', b', T', V', W', L_T, L_V, L_W)$
$d_2 \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_2}$	$\xleftarrow{d_1, C_2}$	$C_2 = \mathbf{g}^{\omega_2}\mathbf{h}^{\mu_2}$
$s_m = \widehat{m} - d_1 m \quad (\text{in } \mathbb{Z})$ $s_k = \widehat{k} - d_1 k, s_l = \widehat{l} - d_1 l$ $F_m = B_m (A_m)^{-d_1} \bmod \mathsf{n}$		
	$\xrightarrow{d_2, \langle s_m, s_k, s_l, F_m\rangle, \langle \widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\rangle}$	$E_m \in^? \mathbb{Z}^*_{\mathsf{n}^2}, s_m \in^? \pm[0, 2^{\lambda_0+\lambda_1+\nu_p+1}]$ $\omega_1 = \mathcal{H}(\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}), C_1 \in^? \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$ $\widehat{E}_m \in^? \mathsf{g}^{s_m} (F_m)^{\mathsf{n}} (E_m)^{d_1} \bmod \mathsf{n}^2$ $\widehat{T} \in^? t^{s_k} T^{d_1}, \widehat{V} \in^? v^{s_l} V^{d_1}$ $\widehat{W} \in^? \theta^{s_m} w^{s_k+s_l} W^{d_1}$ $s_x = \widehat{x} - d_2 x,$ $s_{k'} = \widehat{k}' - d_2 k', s_{l'} = \widehat{l}' - d_2 l'$
$\omega_2 = \mathcal{H}(a', b', T', V', W', L_T, L_V, L_W)$	$\xleftarrow[\langle a', b', T', V', W', L_T, L_V, L_W, \mu_2\rangle]{\langle s_x, s_{k'}, s_{l'}\rangle}$	
$C_2 \in^? \mathbf{g}^{\omega_2}\mathbf{h}^{\mu_2}$ $e(a', Y) \in^? e(b', g)$ $L_T \in^? e(T, b')^{s_x} e(t, a')^{s_{k'}} e(T', \theta)^{d_2}$ $L_V \in^? e(V, b')^{s_x} e(v, a')^{s_{l'}} e(V', \theta)^{d_2}$ $L_W \in^? (e(W, b')e(\theta, a'))^{s_x} \cdot e(w, a')^{s_{k'}+s_{l'}} e(W', \theta)^{d_2}$ $a = (a')^\alpha, b = (b')^\alpha, c = \left(\frac{W'}{T'^\delta V'^\xi}\right)^\alpha$ $\sigma = \langle a, b, c\rangle$		
output $(m; \sigma)$		

Figure 2: Blind signature generation protocol.

$$
\begin{aligned}
\widehat{E}_m &= \mathsf{g}^{\widehat{m}}(B_m)^{\mathsf{n}} \bmod \mathsf{n}^2 = \mathsf{g}^{s_m+d_1\cdot m}(F_m\cdot(A_m)^{d_1})^{\mathsf{n}} \bmod \mathsf{n}^2\\
&= (\mathsf{g}^{s_m}(F_m)^{\mathsf{n}})\cdot(\mathsf{g}^m(A_m)^{\mathsf{n}})^{d_1} \bmod \mathsf{n}^2 = \mathsf{g}^{s_m}(F_m)^{\mathsf{n}}(E_m)^{d_1} \bmod \mathsf{n}^2,\\
\widehat{W} &= \theta^{\widehat{m}}w^{\widehat{k}+\widehat{l}} = \theta^{s_m+d_1\cdot m}w^{(s_k+s_l)+d_1\cdot(k+l)} = (\theta^{s_m}w^{s_k+s_l})\cdot(\theta^m w^{k+l})^{d_1} = \theta^{s_m}w^{s_k+s_l}W^{d_1},\\
\widehat{T} &= t^{\widehat{k}} = t^{s_k+d_1\cdot k} = t^{s_k}\cdot(t^k)^{d_1} = t^{s_k}T^{d_1}, \qquad \widehat{V} = v^{\widehat{l}} = v^{s_l+d_1\cdot l} = v^{s_l}\cdot(v^l)^{d_1} = v^{s_l}V^{d_1};\\
L_T &= e(T,b')^{\widehat{x}}e(t,a')^{\widehat{k}'} = e(T,b')^{s_x+d_2x}e(t,a')^{s_{k'}+d_2k'} = e(T,b')^{s_x}e(t,a')^{s_{k'}}\left(e(T,\theta^{y\alpha'})^x e(t,\theta^{\alpha'})^{k'}\right)^{d_2}\\
&= e(T,b')^{s_x}e(t,a')^{s_{k'}}\left(e(T,b')^x e(t,a')^{k'}\right)^{d_2} = e(T,b')^{s_x}e(t,a')^{s_{k'}}\left(e(T^{xy\alpha'},\theta)e(t^{k'\alpha'},\theta)\right)^{d_2}\\
&= e(T,b')^{s_x}e(t,a')^{s_{k'}}e(T^{xy\alpha'}t^{k'\alpha'},\theta)^{d_2} = e(T,b')^{s_x}e(t,a')^{s_{k'}}e(T',\theta)^{d_2},\\
L_V &= e(V,b')^{\widehat{x}}e(v,a')^{\widehat{l}'} = e(V,b')^{s_x+d_2x}e(v,a')^{s_{l'}+d_2l'} = e(V,b')^{s_x}e(v,a')^{s_{l'}}\left(e(V,\theta^{y\alpha'})^x e(v,\theta^{\alpha'})^{l'}\right)^{d_2}\\
&= e(V,b')^{s_x}e(v,a')^{s_{l'}}\left(e(V,b')^x e(v,a')^{l'}\right)^{d_2} = e(V,b')^{s_x}e(v,a')^{s_{l'}}\left(e(V^{xy\alpha'},\theta)e(v^{l'\alpha'},\theta)\right)^{d_2}\\
&= e(V,b')^{s_x}e(v,a')^{s_{l'}}e(V^{xy\alpha'}v^{l'\alpha'},\theta)^{d_2} = e(V,b')^{s_x}e(v,a')^{s_{l'}}e(V',\theta)^{d_2},\\
L_W &= (e(W,b')e(\theta,a'))^{\widehat{x}}\,e(w,a')^{\widehat{k}'+\widehat{l}'} = (e(W,b')e(\theta,a'))^{s_x+d_2x}\,e(w,a')^{(s_{k'}+s_{l'})+d_2(k'+l')}\\
&= (e(W,b')e(\theta,a'))^{s_x}\,e(w,a')^{s_{k'}+s_{l'}}\left((e(W,b')e(\theta,a'))^x\,e(w,a')^{k'+l'}\right)^{d_2}\\
&= (e(W,b')e(\theta,a'))^{s_x}\,e(w,a')^{s_{k'}+s_{l'}}\left(\left(e(W,\theta^{y\alpha'})e(\theta,\theta^{\alpha'})\right)^x\,e(w,\theta^{\alpha'})^{k'+l'}\right)^{d_2}\\
&= (e(W,b')e(\theta,a'))^{s_x}\,e(w,a')^{s_{k'}+s_{l'}}e(W^{xy\alpha'}\theta^{x\alpha'}w^{(k'+l')\alpha'},\theta)^{d_2}\\
&= (e(W,b')e(\theta,a'))^{s_x}\,e(w,a')^{s_{k'}+s_{l'}}e(W',\theta)^{d_2}.
\end{aligned}
$$

Then we check the correctness of the CL-signature.

$$
\begin{aligned}
a &= (a')^{\alpha} = \theta^{\alpha\alpha'},\\
b &= (b')^{\alpha} = (\theta^y)^{\alpha\alpha'} = (\theta^{\alpha\alpha'})^y = a^y,\\
c &= (W'/(T'^{\delta}V'^{\xi}))^{\alpha} = ((W^{xy}\theta^x w^{k'+l'})/((T^{xy}t^{k'})^{\delta}(V^{xy}v^{l'})^{\xi}))^{\alpha\alpha'}\\
&= ((W/(T^{\delta}V^{\xi}))^{xy}\cdot\theta^x\cdot(w^{k'+l'}/(t^{\delta k'}v^{\xi l'})))^{\alpha\alpha'}\\
&= ((\theta^m)^{xy}\cdot\theta^x\cdot 1)^{\alpha\alpha'} = (\theta^{\alpha\alpha'})^{mxy+x} = a^{mxy+x}
\end{aligned}
$$

So, $e(a,Y) = e(g,b)$ and $e(X,a)e(X,b)^m = e(g,c)$. □

4.4.2 Unforgeability

In this subsection, we prove the unforgeability of our scheme. Before proving the unforgeability of our scheme, we first build a useful lemma which guarantees that the user will use the same plaintext in the Linear Encryption and in the Paillier encryption based on the three-move proof in the blind signature generation protocol. Based on the lemma, then we can simulate the signer successfully and reduce the unforgeability to the unforgeability of the CL-signature.

Lemma 4.2. *In the blind signature generation protocol, under the DLOG assumption, a PPT adversary can generate a valid proof with the signer such that*

$$\log_\theta\left(\mathtt{dec}^{LE}(T,V,W)\right) \neq \mathtt{dec}^{Pai}(E_m) \mod p$$

only with probability $2^{-\lambda_1}$.

Proof. Define $m = \mathtt{dec}^{Pai}(E_m)$. Paillier encryption is 1-1 over $\mathbb{Z}^*_{\mathsf{n}^2}$, so it is well-defined and $m \in \mathbb{Z}_{\mathsf{n}}$. Also $E_m \in \mathbb{Z}^*_{\mathsf{n}^2}$ can be written as $E_m = \mathsf{g}^m(A_m)^{\mathsf{n}} \bmod \mathsf{n}^2$ for some $A_m \in \mathbb{Z}^*_{\mathsf{n}}$. Similarly, define $m' = \log_\theta\left(\mathtt{dec}^{LE}(T,V,W)\right)$. Recall that $\theta \in \mathbb{G}\backslash\{1\}$ and the order of $\mathbb{G}$ is prime p. So θ is a generator of $\mathbb{G}$, and we can get $\theta^{m'} = \mathtt{dec}^{LE}(T,V,W)$ and $m' \in \mathbb{Z}_p$. Also $t, v \in \mathbb{G}$ are generators of $\mathbb{G}$, and $T, V \in \mathbb{G}$ can be

written as $T = t^k$, $V = v^l$ for some $k, l \in \mathbb{Z}_p$. Note that $\mathtt{dec}^{LE}(T, V, W) = \frac{W}{T^\delta \cdot V^\xi}$. So $W = \theta^{m'} T^\delta V^\xi = \theta^{m'} t^{k\delta} v^{l\xi} = \theta^{m'} w^{k+l}$.

Now we assume that there is a PPT adversary who can generate a valid proof with the signer such that $m \neq m' \bmod p$. Up to now we have equations:

$$m \neq m' \bmod p \qquad m \in \mathbb{Z}_{\mathsf{n}}, m' \in \mathbb{Z}_p \tag{1}$$
$$E_m = \mathsf{g}^m (A_m)^{\mathsf{n}} \bmod \mathsf{n}^2 \qquad A_m \in \mathbb{Z}_{\mathsf{n}}^* \tag{2}$$
$$W = \theta^{m'} w^{k+l} \qquad k, l \in \mathbb{Z}_p \tag{3}$$
$$T = t^k \tag{4}$$
$$V = v^l \tag{5}$$

We have assumed that the proof is valid. So all verification equations hold:

$$\widehat{E}_m = \mathsf{g}^{s_m} (F_m)^{\mathsf{n}} (E_m)^{d_1} \bmod \mathsf{n}^2 \tag{6}$$
$$\widehat{W} = \theta^{s_m} w^{s_k + s_l} W^{d_1} \tag{7}$$
$$\widehat{T} = t^{s_k} T^{d_1} \tag{8}$$
$$\widehat{V} = v^{s_l} V^{d_1} \tag{9}$$

From equations (2) and (6), we have

$$E_m = \mathsf{g}^{s_m} (F_m)^{\mathsf{n}} (E_m)^{d_1} \bmod \mathsf{n}^2 = \mathsf{g}^{s_m} (F_m)^{\mathsf{n}} (\mathsf{g}^m (A_m)^{\mathsf{n}})^{d_1} \bmod \mathsf{n}^2 = \mathsf{g}^{s_m + d_1 m} (F_m (A_m)^{d_1})^{\mathsf{n}} \bmod \mathsf{n}^2$$

By the similar way, we can get $\widehat{T} = t^{s_k + d_1 k}$, $\widehat{V} = v^{s_l + d_1 l}$, and $\widehat{W} = \theta^{s_m + d_1 m'} w^{(s_k + d_1 k) + (s_l + d_1 l)}$. Now we call

$$\widehat{m} \stackrel{\mathsf{def}}{=} s_m + d_1 m \bmod \mathsf{n} \tag{10}$$
$$B_m \stackrel{\mathsf{def}}{=} F_m (A_m)^{d_1} \bmod \mathsf{n} \tag{11}$$
$$\widehat{k} \stackrel{\mathsf{def}}{=} s_k + d_1 k \bmod p \tag{12}$$
$$\widehat{l} \stackrel{\mathsf{def}}{=} s_l + d_1 l \bmod p \tag{13}$$
$$\widehat{m}' \stackrel{\mathsf{def}}{=} s_m + d_1 m' \bmod p \tag{14}$$

Consider that $\gcd(\mathsf{n}, p) = 1$. From the equation (10), we can let $\widehat{m} = s_m + d_1 m + A\mathsf{n}$, where $A \in \mathbb{Z}$. So $\widehat{m} - s_m - d_1 m = A\mathsf{n}$. Recall that $s_m \in \pm[0, 2^{\lambda_0 + \lambda_1 + \nu_p + 1}]$, and $\widehat{m} \in \pm[0, 2^{\lambda_0 + \lambda_1 + \nu_p}]$, $d_1 \in \{0,1\}^{\lambda_1}$, and $m \in [0, 2^{\nu_p}]$. So $\widehat{m} - s_m - d_1 m \in \pm[0, 2^{\lambda_0 + \lambda_1 + \nu_p + 2}]$, and $A = 0$ because $\ell_{\mathsf{n}} \gg \nu_p + \lambda_0 + \lambda_1 + 3$. So $\widehat{m} = s_m + d_1 m$.

From the equation (14), we can let $\widehat{m}' = s_m + d_1 m' + Bp$ where $B \in \mathbb{Z}$. So $\widehat{m} - \widehat{m}' = d_1 (m - m') - Bp$. Recall that $p \nmid (m - m')$. We can find such B only in the case of $p \mid (\widehat{m} - \widehat{m}') - d_1 (m - m')$. Note that $\langle m, m', \widehat{m}, \widehat{m}' \rangle$ is determined before receiving the challenge d_1 from the signer because $\langle t, v, w, E_m, \theta, T, V, W; C_1 \rangle$ is sent before receiving d_1 and $\langle \widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W} \rangle$ is bound by the commitment C_1 under the DLOG assumption. So we have only probability $2^{-\lambda_1}$ to find B. Therefore, under the DLOG assumption, the adversary cannot develop a valid proof with $m \neq m' \bmod p$ except negligible probability $2^{-\lambda_1}$.

□

Theorem 4.3 (Unforgeability). *The proposed scheme is unforgeable under the LRSW assumption.*

Proof. In this part, we will show under LRSW assumption, no PPT adversary user $\mathcal{A}$ can achieve "one-more" forgery with non-negligible probability. Let $(p, g, \mathbb{G}, \mathbb{G}_T, e; X, Y)$ be the input instance of LRSW problem. If a PPT user $\mathcal{A}$ obtains $\ell + 1$ valid message-signature pairs after ℓ times successful executions with the signer, we can construct oracle $\mathcal{I}$ which will output a valid pair $(m^*, \langle a^*, b^*, c^* \rangle)$, where m^* is not queried to the oracle $O_{X,Y}$.

1. The oracle sets $\mathtt{pub} = \langle p, g, \mathbb{G}, \mathbb{G}_T, e\rangle$ and $PK_{\mathsf{S}} = \langle X, Y\rangle$. The oracle generates $\mathtt{crs}_1 = \langle Q, \mathbf{g}, \mathbf{h}, \mathbf{G}, \mathcal{H}\rangle$ and $\tau_1 = r$ for the equivocal Pedersen commitment scheme; generates $\mathtt{crs}_2 = \langle \mathsf{n}, \mathsf{g}\rangle$ and $\tau_2 = \langle \mathsf{p}, \mathsf{q}\rangle$ for the Paillier encryption; sets $\mathtt{crs} = (\mathtt{crs}_1, \mathtt{crs}_2)$. Now the oracle supplies the adversary with $\langle \mathtt{pub}, \mathtt{crs}, PK_{\mathsf{S}}\rangle$, keeps $\langle \tau_1, \tau_2\rangle$.

2. The oracle $\mathcal{I}$ will be queried by $\mathcal{A}$ which operates like that in one of the two cases below:

Case 1: $\mathcal{A}$ queries $\mathcal{I}$ with $\langle \mathtt{start}, msg\rangle$, where $msg = \{PK_{\mathsf{U}}, E_m, \langle \theta, T, V, W\rangle, C_1\}$. The oracle $\mathcal{I}$ will create a session identity sid and set the corresponding state $st = \perp$; the oracle $\mathcal{I}$ will simulates the signer S with msg till S either terminates or returns a response rsp to the user; the oracle $\mathcal{I}$ records the current state in st. If S returns rsp then $\mathcal{I}$ returns this with the session identity to $\mathcal{A}$, i.e. $\mathcal{I}$ returns $\{sid, d_1, C_2\}$ to $\mathcal{A}$, where $d_1 \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_1}$ and $C_2 = \mathbf{g}^{\gamma_2}$, $\gamma_2 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$.

Case 2: $\mathcal{A}$ queries $\mathcal{I}$ with $\langle \mathtt{advance}, sid, msg\rangle$, where $msg = \{d_2, \langle s_m, s_k, s_l, F_m\rangle, \langle \widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\rangle\}$. The oracle $\mathcal{I}$ will simulate the signer S with msg and previous state st. The S checks whether all equations hold: $C_1 =^? \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$ where $\omega_1 = \mathcal{H}(\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W})$, $\widehat{E}_m =^? \mathsf{g}^{s_m}(F_m)^{\mathsf{n}}(E_m)^{d_1} \bmod \mathsf{n}^2$, $\widehat{T} =^? t^{s_k}T^{d_1}$, $\widehat{V} =^? v^{s_l}V^{d_1}$, $\widehat{W} =^? \theta^{s_m}w^{s_k+s_l}W^{d_1}$. If not true, terminates. Otherwise, the oracle $\mathcal{I}$ generates an identically distributed response to $\mathcal{A}$.

Consider the Pedersen commitment scheme is involved. From Lemma 4.2 above, under the DLOG assumption, except negligible error probability $2^{-\lambda_1}$, the oracle $\mathcal{I}$ can obtain the m under $\{\theta, T, V, W\}$ by decrypting m from E_m, and then obtain $\langle a', b', T', V', W'\rangle$ based on this m: the oracle $\mathcal{I}$ simulates S to decrypt E_m into $m = \mathtt{dec}^{Pai}_{\tau_2}(E_m)$ by using the trapdoor information $\tau_2 = \langle \mathsf{p}, \mathsf{q}\rangle$; then the oracle $\mathcal{I}$ simulates $O_{X,Y}$ with input $m \bmod p$ which returns $\langle a, b, c\rangle$, and computes $a' = a$, $b' = b$, $W' = cw^{k''+l''}$, $T' = t^{k''}$, $V' = v^{l''}$, where $k'', l'' \xleftarrow{\mathtt{r}} \mathbb{Z}_p$. Note that here $\langle T', V', W'\rangle$ is in fact the ciphertext of c over the public key $\langle t, v, w\rangle$. The simulated $\{a', b', T', V', W'\}$ is indistinguishable from the protocol answer consider the error probability $2^{-\lambda_1}$ is negligible. In fact, without the error probability, the two distribution is identical, i.e. $\{a, b, cw^{k''+l''}, t^{k''}, v^{l''}\} \approx \{(\theta)^{\alpha'}, (\theta^y)^{\alpha'}, (W^{xy}\theta^x w^{k'+l'})^{\alpha'}, (T^{xy}t^{k'})^{\alpha'}, (V^{xy}v^{l'})^{\alpha'}\}$, for random $\langle k'', l''\rangle$ and $\langle \alpha', k', l'\rangle$. Note that $\langle a, b, c\rangle$ is the response from $O_{X,Y}$. So, a is a random element in $\mathbb{G}$, $b = a^y$, $c = a^{x+mxy}$. We know $W = \theta^m w^{k+l}$, $T = t^k$, $V = v^l$, for some $k, l \in \mathbb{Z}_p$. We can compute $(W^{xy}\theta^x w^{k'+l'})^{\alpha'} = ((\theta^m w^{k+l})^{xy}\theta^x w^{k'+l'})^{\alpha'} = ((\theta)^{\alpha'})^{x+mxy} w^{(kxy+k')\alpha'+(lxy+l')\alpha'}$, $(T^{xy}t^{k'})^{\alpha'} = ((t^k)^{xy}t^{k'})^{\alpha'} = t^{(kxy+k')\alpha'}$, $(V^{xy}v^{l'})^{\alpha'} = ((v^l)^{xy}v^{l'})^{\alpha'} = v^{(lxy+l')\alpha'}$. Replace $\theta^{\alpha'}, (kxy+k')\alpha', (lxy+l')\alpha'$ with a, k'', l'', we will know the two probability distributions are identical.

Next, the oracle $\mathcal{I}$ randomly selects $s_x, s_{k'}, s_{l'} \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, and let $L_T = e(T, b')^{s_x}e(t, a')^{s_{k'}}e(T', \theta)^{d_2}$, $L_V = e(V, b')^{s_x}e(v, a')^{s_{l'}}e(V', \theta)^{d_2}$, $L_W = (e(W, b')e(\theta, a'))^{s_x}\, e(w, a')^{s_{k'}+s_{l'}}e(W', \theta)^{d_2}$; computes $\omega_2 = \mathcal{H}(a', b', T', V', W', L_T, L_V, L_W)$; uses the trapdoor $\tau_1 = r$ to compute μ_2 such that $C_2 = \mathbf{g}^{\omega_2}\mathbf{h}^{\mu_2}$, i.e. $\mu_2 = \frac{\gamma_2 - \omega_2}{r}$. Consider the 3-move proof is zero-knowledge [Dam00], the simulated distribution $\{a', b', T', V', W', L_T, L_V, L_W, \mu_2; s_x, s_{k'}, s_{l'}\}$ is indistinguishable from that in the protocol answer.

3. $\mathcal{A}$ outputs message-signature pairs.

Now assume that $\mathcal{A}$ can break the scheme, which means $\mathcal{A}$ can generate ℓ' message-signature pairs $(m_1^*; \sigma_1^*), (m_2^*; \sigma_2^*), \ldots, (m_{\ell'}^*; \sigma_{\ell'}^*)$ with $m_i \neq m_j$ and $\ell' > \ell$. Since $\ell' - \ell \geq 1$, at least one message, say m_O^*, is not queried to oracle $O_{X,Y}$, though $(m_O^*; \sigma_O^*)$ is a valid pair. In other word, we can construct a valid pair $(m_O^*; \sigma_O^*)$, where m_O^* is not in query history. This breaks the LRSW assumption.

□

4.4.3 Blindness

In this subsection, we show the blindness of our scheme. Before going to the proof of the blindness of our scheme, we first build a useful lemma which guarantee that the signer will use the correct ciphertext $\langle \theta, T, V, W \rangle$ and his secret key $\langle x, y \rangle$ to generate $\langle a', b', T', V', W' \rangle$ based on the three-move proof.

Lemma 4.4. *In the blind signature generation protocol, under the DLOG assumption, a PPT adversary can generate a valid proof with the user such that*

$$\log_g Y \neq \log_{a'} b' \mod p$$

or

$$\log_g X + \log_g X \cdot \log_g Y \cdot \log_\theta \left(\mathtt{dec}^{LE}(T, V, W)\right) \neq \log_{a'} \left(\mathtt{dec}^{LE}(T', V', W')\right) \mod p$$

only with probability $2^{-\lambda_2}$.

Proof. Based on the verification equation $e(a', Y) = e(b', y)$ it is very easy to prove the first part of the lemma. Next we focus on the second part. Now we have $Y = g^y$, $X = g^x$, $b' = (a')^y$. Define $m = \log_\theta \left(\mathtt{dec}^{LE}(T, V, W)\right)$, and we have $T = t^k$, $V = v^l$, $W = \theta^m w^{k+l}$ for some $k, l \in \mathbb{Z}_p$ by using the same argument in the proof of Lemma 4.2. Note that $\mathbb{G}_T$ is also order prime p. There exist $\widehat{x}, \widehat{k'}, \widehat{l'}, \widehat{\eta}, k', l', \eta \in \mathbb{Z}_p$ such that,

$$L_T = e(T, b')^{\widehat{x}} e(t, a')^{\widehat{k'}} \tag{15}$$
$$L_V = e(V, b')^{\widehat{x}} e(v, a')^{\widehat{l'}} \tag{16}$$
$$L_W = (e(W, b')e(\theta, a'))^{\widehat{x}}\, e(w, a')^{\widehat{\eta}} \tag{17}$$
$$e(T', \theta) = e(T, b')^x e(t, a')^{k'} \tag{18}$$
$$e(V', \theta) = e(V, b')^x e(v, a')^{l'} \tag{19}$$
$$e(W', \theta) = (e(W, b')e(\theta, a'))^x\, e(w, a')^{\eta} \tag{20}$$

Assume there is a PPT can generate valid proof such that $\log_g X + \log_g X \cdot \log_g Y \cdot \log_\theta \left(\mathtt{dec}^{LE}(T, V, W)\right) \neq \log_{a'} \left(\mathtt{dec}^{LE}(T', V', W')\right) \mod p$; the verification equations are

$$L_T = e(T, b')^{s_x} e(t, a')^{s_{k'}} e(T', \theta)^{d_2} \tag{21}$$
$$L_V = e(V, b')^{s_x} e(v, a')^{s_{l'}} e(V', \theta)^{d_2} \tag{22}$$
$$L_W = (e(W, b')e(\theta, a'))^{s_x}\, e(w, a')^{s_{k'} + s_{l'}} e(W', \theta)^{d_2} \tag{23}$$

From equations (15,16,18,19,21,22), we can obtain

$$s_x = \widehat{x} + d_2 x \mod p \tag{24}$$
$$s_{k'} = \widehat{k'} + d_2 k' \mod p \tag{25}$$
$$s_{l'} = \widehat{l'} + d_2 l' \mod p \tag{26}$$

From equations (17, 20, 23, 24), we can obtain

$$s_{k'} + s_{l'} = \widehat{\eta} + d_2 \eta \mod p \tag{27}$$

From equations (25-27), we can obtain

$$\widehat{k'} + \widehat{l'} - \widehat{\eta} = -d_2(k' + l' - \eta) \mod p \tag{28}$$

Note that $\langle a', b', T', V', W'; L_T, L_V, L_W \rangle$ is bound by commitment C_2 which is sent before the challenge d_2; and $\langle k', l', \eta, \widehat{k'}, \widehat{l'}, \widehat{\eta} \rangle$ is determined before receiving d_2 from the user. So, except probability $2^{-\lambda_2}$, the signer cannot get d_2 before receiving it from the user. Now the equation $\eta = k' + l' \bmod p$ holds; otherwise the signer can compute such $d_2 = -(\widehat{k'} + \widehat{l'} - \widehat{\eta})/(k' + l' - \eta)$ before he receives the value. Put the equation $\eta = k' + l' \bmod p$ into equation (28), we can also get $\widehat{\eta} = \widehat{k'} + \widehat{l'} \bmod p$. Assume $a' = \theta^{\alpha'}$ and recall that $b' = (a')^y$, we can obtain $T' = T^{xy\alpha'} t^{k'\alpha'}$ from equation (18); similarly we can obtain $V' = V^{xy\alpha'} v^{l'\alpha'}$ and $W' = W^{xy\alpha'} \theta^{x\alpha'} w^{k'\alpha' + l'\alpha'}$. Define $c' = \mathtt{dec}^{LE}(T', V', W')$. Then $c' = \frac{W'}{(T')^\delta (V')^\xi} = \theta^{(x + xym)\alpha'} = (a')^{x + xym}$. And $\log_{a'} \left(\mathtt{dec}^{LE}(T', V', W')\right) = \log_{a'} c' = x + xym =$

$\log_g X + \log_g X \cdot \log_g Y \cdot \log_\theta \left(\mathtt{dec}^{LE}(T,V,W)\right) \mod p$ which contradicts the assumption. So, based on a secure commitment scheme, except the probability $2^{-\lambda_2}$, no PPT adversary can develop a valid proof such that $\log_g X + \log_g X \cdot \log_g Y \cdot \log_\theta \left(\mathtt{dec}^{LE}(T,V,W)\right) \neq \log_{a'} \left(\mathtt{dec}^{LE}(T',V',W')\right) \mod p$. This completes the proof.

□

Theorem 4.5 (Blindness). *The proposed scheme is blind under the DLDH assumption and the DCR assumption.*

We start from the blindness model, and define it as Game 0; we slightly change Game 0 by simulating the left user instantiation by Damgård's trick in Game 1; and then we slightly change Game 1 again and do the similar simulation for the right user instantiation in Game 2. The statistical distance of the probability distribution of Game 0 and Game 1, and of Game 1 and Game 2 are negligible. Now we slightly change Game 2 into Game 3 when two user instantiations verify the verification equations successfully: instead of generating σ based on $\langle a', b', T', V', W' \rangle$ in Game 2, generate σ by using the signing key (x, y) on m. Based on Lemma 4.4, we show the statistical distance between Game 2 and Game 3 is negligible. Next we slightly change Game 3 by simulating the left user instantiation with inputting a random message (not one of the messages selected by the adversary) to the Paillier encryption in Game 4; then do the similar simulation for the right user instantiation in Game 5. Both distances between Game 3 and Game 4, and Game 4 and Game 5 are negligible under the DCR assumption. Similarly, we slightly change Game 5 into Game 6 by simulating the left user instantiation with inputting a random message to the linear encryption; then change Game 6 into Game 7 by similar way for the right user instantiation. Again the distances between Game 5 and Game 6, and Game 6 and Game 7 are negligible under the DLDH assumption. Therefore, the probability distribution in Game 0 is indistinguishable from that in Game 7. Consider in Game 7, the two messages (m_0, m_1) have never been involved in the communications between the user instantiations and the adversary signer, which means the adversary has no advantage to win the game (with just probability $\frac{1}{2}$ to predict ϕ). So, in Game 0, the adversary has at most negligible advantage to win the game under the assumptions.

Proof. We use the sequential games technique to prove this part, and define games $\mathsf{G}_j^{\mathcal{A}}$ between the adversary $\mathcal{A}$ and the oracle $\mathcal{I}_j^{\phi}$ which simulates two user instantiation: the left one U^L and the right one U^R, where $j = 0, 1, \ldots, 7$. Also we define $\mathbf{E}_j$ to be the event that $\phi = \phi'$ in $\mathsf{G}_j^{\mathcal{A}}$.

Game 0:

Follow the blindness model, we can define Game 0 as below:

$\mathsf{G}_0^{\mathcal{A}}(1^\lambda)$

1. $\phi \xleftarrow{\mathtt{r}} \{0,1\}$;
2. $(\mathtt{pub}, \mathtt{crs}, PK_{\mathsf{S}}, SK_{\mathsf{S}}) \leftarrow \mathtt{gen}(1^\lambda)$;
3. $\phi' \leftarrow \mathcal{A}^{\mathcal{I}_0^{\phi}(1^\lambda, \mathtt{pub}, \mathtt{crs}, PK_{\mathsf{S}})}(1^\lambda, \mathtt{pub}, \mathtt{crs}, PK_{\mathsf{S}}, SK_{\mathsf{S}})$;
4. if $\phi = \phi'$ then 1;

Here $\mathcal{I}_0^{\phi}$ is defined as:

- Given $\langle \mathtt{challenge}, m_0, m_1 \rangle$, the oracle $\mathcal{I}_0^{\phi}$ simulates U^L (resp. U^R) with m_ϕ (resp. $m_{1-\phi}$). The oracle $\mathcal{I}_0^{\phi}$ keeps a database with the state of each user instantiation; the state includes all coin tosses of the user instantiation and the contents of all tapes including the communication tape. Here the oracle uses st^L (resp. st^R) to record the state of U^L (resp. U^R).

- Given $\langle \mathtt{advance}, \rho, msg \rangle$, where $\rho \in \{L, R\}$:

– If $msg = \perp$, then $\mathcal{I}_0^\phi$ recovers the state of st^ρ, and simulates the user instantiation U^ρ till U^ρ either terminates or returns a response to the signer. If U^ρ returns a response rsp, then $\mathcal{I}_0^\phi$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^\rho = st^\rho||st$. Let m be the simulated message for U^ρ, i.e. $m = m_\phi$ for $\rho = L$ and $m = m_{1-\phi}$ for $\rho = R$, we have,

(a) $(PK_{\mathsf{U}}^\rho, SK_{\mathsf{U}}^\rho) \leftarrow \mathtt{gen}^{LE}(1^\lambda)$

(b) $\widehat{m} \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}]$, $A_m, B_m \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathtt{n}}^*$, $\alpha, k, l, \widehat{k}, \widehat{l} \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}\backslash\{1\}$, $\mu_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$.

(c) $E_m \leftarrow \mathtt{enc}_{\mathtt{crs}_2}^{Pai}(m, A_m)$

(d) $\langle T, V, W\rangle \leftarrow \mathtt{enc}_{\mathtt{pub}, PK_{\mathsf{U}}^\rho}^{LE}(m, \theta, \widehat{k}, \widehat{l})$

(e) $\widehat{E}_m \leftarrow \mathtt{enc}_{\mathtt{crs}_2}^{Pai}(\widehat{m}, B_m)$

(f) $\langle \widehat{T}, \widehat{V}, \widehat{W}\rangle \leftarrow \mathtt{enc}_{\mathtt{pub}, PK_{\mathsf{U}}^\rho}^{LE}(\widehat{m}, \theta, \widehat{k}, \widehat{l})$

(g) $\omega_1 = \mathcal{H}(\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W})$, $C_1 = \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$

(h) $rsp = \{PK_{\mathsf{U}}, E_m, \langle\theta, T, V, W\rangle, C_1\}$

– If $msg = \{d_1, C_2\}$, then $\mathcal{I}_0^\phi$ recovers the state of st^ρ, and simulates the user instantiation U^ρ with msg till U^ρ either terminates or returns a response rsp to the signer. If U^ρ returns a response rsp, then $\mathcal{I}_0^\phi$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^\rho = st^\rho||st$.

Here rsp is in the form of $\{d_2, \langle s_m, s_k, s_l, F_m\rangle, \langle\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\rangle\}$, where $\langle\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\rangle$ is recovered from the previous state of st^ρ, and $\langle s_m, s_k, s_l, F_m\rangle$ is generated as: $s_m = \widehat{m} - d_1 \cdot m$ in $\mathbb{Z}$, $s_k = \widehat{k} - d_1 \cdot k \bmod p$, $s_l = \widehat{l} - d_1 \cdot l \bmod p$, $F_m = B_m(A_m)^{-d_1} \bmod \mathtt{n}$, $d_2 \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_2}$.

- Given $\langle\mathtt{terminate}, msg^L, msg^R\rangle$, the oracle $\mathcal{I}_0^\phi$ recovers the state st^L (resp. st^R), and simulates the user instantiation U^L (resp. U^R) with msg^L (resp. msg^R) till U^L (resp. U^R) either terminates or returns an output, where msg^ρ is in form of $\{s_x, s_{k'}, s_{l'}; a', b', T', V', W', L_T, L_V, L_W, \mu_2\rangle\}$. Each U^ρ will verify all equations:

$C_2 = \mathbf{g}^{\omega_2}\mathbf{h}^{\mu_2}$ where $\omega_2 = \mathcal{H}(a', b', T', V', W', L_T, L_V, L_W)$,
$e(a', Y) = e(b', g)$,
$L_T = e(T, b')^{s_x} e(t, a')^{s_{k'}} e(T', \theta)^{d_2}$,
$L_V = e(V, b')^{s_x} e(v, a')^{s_{l'}} e(V', \theta)^{d_2}$,
$L_W = (e(W, b')e(\theta, a'))^{s_x}\, e(w, a')^{s_{k'}+s_{l'}} e(W', \theta)^{d_2}$

If the two user instantiations verify the verification equations successfully, each of them generates $\sigma = (a, b, c)$ by $a = (a')^\alpha$, $b = (b')^\alpha$, $c = (W'/(T'^\delta V'^\xi))^\alpha$. Let the generated signatures from the two user instantiations be σ_0, σ_1 for message m_0, m_1 respectively. The oracle set $rsp = (\sigma_0, \sigma_1)$. Otherwise set $rsp = (\perp, \perp)$. The oracle returns rsp to $\mathcal{A}$.

Game 1:

We modify $\mathsf{G}_0^\mathcal{A}$ into $\mathsf{G}_1^\mathcal{A}$ by changing step 2 into:

2. $(\mathtt{pub}, \mathtt{crs}_2, PK_{\mathsf{S}}, SK_{\mathsf{S}}) \leftarrow \mathtt{gen}(1^\lambda)$; generates $\mathtt{crs}_1 = \langle Q, \mathbf{g}, \mathbf{h}, \mathbf{G}, \mathcal{H}\rangle$ and $\tau_1 = r$ for the equivocal Pedersen commitment scheme; set $\mathtt{crs} = (\mathtt{crs}_1, \mathtt{crs}_2)$.

and changing $\mathcal{I}_0^\phi$ into $\mathcal{I}_1^\phi$. Note that $\mathcal{I}_1^\phi$ is same as $\mathcal{I}_0^\phi$ except that

- Given $\langle\mathtt{advance}, \rho, msg\rangle$, where $\rho \in \{L, R\}$. If $\rho = R$, $\mathcal{I}_1^\phi$ operates identically as $\mathcal{I}_0^\phi$; but if $\rho = L$, $\mathcal{I}_1^\phi$ works as follows:

– If $msg = \perp$, then $\mathcal{I}_1^\phi$ recovers the state of st^L, and simulates the user instantiation U^L till U^L either terminates or returns a response to the signer. If U^L returns a response rsp, then $\mathcal{I}_1^\phi$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$. Let $m = m_\phi$, we have,

(a) $(PK_{\mathsf{U}}^L, SK_{\mathsf{U}}^L) \leftarrow \mathtt{gen}^{LE}(1^\lambda)$

(b) $A_m \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$, $\alpha, k, l \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}\backslash\{1\}$.

(c) $E_m \leftarrow \mathtt{enc}_{\mathtt{crs}_2}^{Pai}(m, A_m)$

(d) $\langle T, V, W\rangle \leftarrow \mathtt{enc}_{\mathtt{pub}, PK_{\mathsf{U}}^L}^{LE}(m, \theta, k, l)$

(e) $\gamma_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$, $C_1 = \mathbf{g}^{\gamma_1}$

(f) $rsp = \{PK_{\mathsf{U}}^L, E_m, \langle\theta, T, V, W\rangle, C_1\}$

– If $msg = \{d_1, C_2\}$, then $\mathcal{I}_1^\phi$ recovers the state of st^L, and simulates the user instantiation U^L with msg till U^L either terminates or returns a response rsp to the signer. If U^L returns a response rsp, then $\mathcal{I}_1^\phi$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$.

(a) $s_m \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}]$, $F_m \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$, $s_k, s_l \xleftarrow{\mathtt{r}} \mathbb{Z}_p$

(b) $\widehat{E}_m = \mathbf{g}^{s_m}(F_m)^{\mathsf{n}}(E_m)^d \mod \mathsf{n}^2$

(c) $\widehat{W} = \theta^{s_m} w^{s_k+s_l} W^{d_1}$, $\widehat{T} = t^{s_k} T^{d_1}$, $\widehat{V} = v^{s_l} V^{d_1}$

(d) use $\tau_1 = r$ to compute μ_1 such that $C_1 = \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$ where $\omega_1 = \mathcal{H}(\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W})$, i.e. $\mu_1 = \frac{\gamma_1-\omega_1}{r} \mod Q$

(e) $rsp = \{d_2, \langle s_m, s_k, s_l, F_m\rangle, \langle\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\rangle\}$

Game 2:

We modify $\mathsf{G}_1^{\mathcal{A}}$ into $\mathsf{G}_2^{\mathcal{A}}$ by changing $\mathcal{I}_1^\phi$ into $\mathcal{I}_2^\phi$. Note that $\mathcal{I}_2^\phi$ is same as $\mathcal{I}_1^\phi$ except that :

- Given $\langle\mathtt{advance}, \rho, msg\rangle$, where $\rho \in \{L, R\}$. If $\rho = L$, $\mathcal{I}_2^\phi$ operates identically as $\mathcal{I}_1^\phi$; but if $\rho = R$, $\mathcal{I}_2^\phi$ operates similarly as the case $\rho = L$ with $m = m_{1-\phi}$, i.e. runs the same operations for the right user instantiation U^R.

Game 3:

We modify $\mathsf{G}_2^{\mathcal{A}}$ into $\mathsf{G}_3^{\mathcal{A}}$ by changing $\mathcal{I}_2^\phi$ into $\mathcal{I}_3^\phi$. Note that $\mathcal{I}_3^\phi$ is same as $\mathcal{I}_2^\phi$ except that

- Given $\langle\mathtt{terminate}, msg^L, msg^R\rangle$, the oracle $\mathcal{I}_3^\phi$ recovers the state st^L (resp. st^R), and simulates the user instantiation U^L (resp. U^R) with msg^L (resp. msg^R) till U^L (resp. U^R) either terminates or returns an output.

  If the two user instantiations verify the verification equations successfully, now the oracle generates two signatures σ_0, σ_1 for m_0, m_1 by using the signing key: $\sigma = (a, a^y, a^{x+xym})$ where $a \xleftarrow{\mathtt{r}} \mathbb{G}$. The oracle set $rsp = (\sigma_0, \sigma_1)$. Otherwise set $rsp = (\perp, \perp)$. The oracle returns rsp to $\mathcal{A}$.

Game 4:

We modify $\mathsf{G}_3^{\mathcal{A}}$ into $\mathsf{G}_4^{\mathcal{A}}$ by changing $\mathcal{I}_3^\phi$ into $\mathcal{I}_4^\phi$. Note that $\mathcal{I}_4^\phi$ is same as $\mathcal{I}_3^\phi$ except that

- Given $\langle\mathtt{challenge}, m_0, m_1\rangle$, the oracle $\mathcal{I}_4^\phi$ randomly selects $\widetilde{m}_0$, $\widetilde{m}_1$ from the message space and simulates U^L (resp. U^R) with m_ϕ or $\widetilde{m}_0$ (resp. $m_{1-\phi}$ or $\widetilde{m}_1$).

- Given $\langle\mathtt{advance}, \rho, msg\rangle$, where $\rho \in \{L, R\}$. If $\rho = R$, $\mathcal{I}_4^\phi$ operates identically as $\mathcal{I}_3^\phi$; but if $\rho = L$, $\mathcal{I}_4^\phi$ works as follows:

– If $msg = \perp$, then $\mathcal{I}_4^\phi$ recovers the state of st^L, and simulates the user instantiation U^L till U^L either terminates or returns a response to the signer. If U^L returns a response rsp, then $\mathcal{I}_4^\phi$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$. Let $\widetilde{m} = \widetilde{m}_0$, $m = m_\phi$, we have,

(a) $(PK_{\mathsf{U}}^L, SK_{\mathsf{U}}^L) \leftarrow \mathtt{gen}^{LE}(1^\lambda)$
(b) $A_m \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$, $\alpha, k, l \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}\backslash\{1\}$.
(c) $E_{\widetilde{m}} \leftarrow \mathtt{enc}_{\mathtt{crs}_2}^{Pai}(\widetilde{m}, A_m)$
(d) $\langle T, V, W\rangle \leftarrow \mathtt{enc}_{\mathtt{pub}, PK_{\mathsf{U}}^L}^{LE}(m, \theta, k, l)$
(e) $\gamma_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$, $C_1 = \mathbf{g}^{\gamma_1}$
(f) $rsp = \{PK_{\mathsf{U}}^L, E_{\widetilde{m}}, \langle\theta, T, V, W\rangle, C_1\}$

– If $msg = \{d_1, C_2\}$, then $\mathcal{I}_4^\phi$ recovers the state of st^L, and simulates the user instantiation U^L with msg till U^L either terminates or returns a response rsp to the signer. If U^L returns a response rsp, then $\mathcal{I}_4^\phi$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$.

(a) $s_m \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}]$, $F_m \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$, $s_k, s_l \xleftarrow{\mathtt{r}} \mathbb{Z}_p$
(b) $\widehat{E}_{\widetilde{m}} = \mathbf{g}^{s_m}(F_m)^{\mathsf{n}}(E_{\widetilde{m}})^{d_1} \mod \mathsf{n}^2$
(c) $\widehat{W} = \theta^{s_m} w^{s_k+s_l} W^{d_1}$, $\widehat{T} = t^{s_k} T^{d_1}$, $\widehat{V} = v^{s_l} V^{d_1}$
(d) use $\tau_1 = r$ to compute μ_1 such that $C_1 = \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$ where $\omega_1 = \mathcal{H}(\widehat{E}_{\widetilde{m}}, \widehat{T}, \widehat{V}, \widehat{W})$, i.e. $\mu_1 = \frac{\gamma_1-\omega_1}{r} \mod Q$
(e) $rsp = \{d_2, \langle s_m, s_k, s_l, F_m\rangle, \langle\widehat{E}_{\widetilde{m}}, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\rangle\}$

Game 5:

We modify $\mathsf{G}_4^{\mathcal{A}}$ into $\mathsf{G}_5^{\mathcal{A}}$ by changing $\mathcal{I}_4^\phi$ into $\mathcal{I}_5^\phi$. Note that $\mathcal{I}_5^\phi$ is same as $\mathcal{I}_4^\phi$ except that

- Given $\langle\mathtt{advance}, \rho, msg\rangle$, where $\rho \in \{L, R\}$. If $\rho = L$, $\mathcal{I}_5^\phi$ operates identically as $\mathcal{I}_4^\phi$; but if $\rho = R$, $\mathcal{I}_5^\phi$ operates similarly as the case $\rho = L$ with $\widetilde{m} = \widetilde{m}_1$, $m = m_{1-\phi}$, i.e. runs the same operations for the right user instantiation U^R.

Game 6:

We modify $\mathsf{G}_5^{\mathcal{A}}$ into $\mathsf{G}_6^{\mathcal{A}}$ by changing $\mathcal{I}_5^\phi$ into $\mathcal{I}_6^\phi$. Note that $\mathcal{I}_6^\phi$ is same as $\mathcal{I}_5^\phi$ except that

- Given $\langle\mathtt{advance}, \rho, msg\rangle$, where $\rho \in \{L, R\}$. If $\rho = R$, $\mathcal{I}_6^\phi$ operates identically as $\mathcal{I}_5^\phi$; but if $\rho = L$, $\mathcal{I}_6^\phi$ works as follows:

– If $msg = \perp$, then $\mathcal{I}_6^\phi$ recovers the state of st^ρ, and simulates the user instantiation U^L till U^L either terminates or returns a response to the signer. If U^L returns a response rsp, then $\mathcal{I}_6^\phi$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$. Let $\widetilde{m} = \widetilde{m}_0$, we have,

(a) $(PK_{\mathsf{U}}^L, SK_{\mathsf{U}}^L) \leftarrow \mathtt{gen}^{LE}(1^\lambda)$
(b) $A_m \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$, $\alpha, k, l \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}\backslash\{1\}$.
(c) $E_{\widetilde{m}} \leftarrow \mathtt{enc}_{\mathtt{crs}_2}^{Pai}(\widetilde{m}, A_m)$
(d) $\langle\widetilde{T}, \widetilde{V}, \widetilde{W}\rangle \leftarrow \mathtt{enc}_{\mathtt{pub}, PK_{\mathsf{U}}^L}^{LE}(\widetilde{m}, \theta, k, l)$
(e) $\gamma_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$, $C_1 = \mathbf{g}^{\gamma_1}$
(f) $rsp = \{PK_{\mathsf{U}}^L, E_{\widetilde{m}}, \langle\theta, \widetilde{T}, \widetilde{V}, \widetilde{W}\rangle, C_1\}$

– If $msg = \{d_1, C_2\}$, then $\mathcal{I}_6^{\phi}$ recovers the state of st^L, and simulates the user instantiation U^L with msg till U^L either terminates or returns a response rsp to the signer. If U^L returns a response rsp, then $\mathcal{I}_6^{\phi}$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$.

(a) $s_m \xleftarrow{\mathsf{r}} \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}]$, $F_m \xleftarrow{\mathsf{r}} \mathbb{Z}_{\mathsf{n}}^*$, $s_k, s_l \xleftarrow{\mathsf{r}} \mathbb{Z}_p$

(b) $\widehat{E}_{\widetilde{m}} = \mathsf{g}^{s_m}(F_m)^{\mathsf{n}}(E_{\widetilde{m}})^{d_1} \mod \mathsf{n}^2$

(c) $\widehat{\widetilde{W}} = \theta^{s_m} w^{s_k+s_l} \widetilde{W}^{d_1}$, $\widehat{\widetilde{T}} = t^{s_k}\widetilde{T}^{d_1}$, $\widehat{\widetilde{V}} = v^{s_l}\widetilde{V}^{d_1}$

(d) use $\tau_1 = r$ to compute μ_1 such that $C_1 = \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$ where $\omega_1 = \mathcal{H}(\widehat{E}_{\widetilde{m}}, \widehat{\widetilde{T}}, \widehat{\widetilde{V}}, \widehat{\widetilde{W}})$, i.e. $\mu_1 = \frac{\gamma_1-\omega_1}{r} \mod Q$

(e) $rsp = \{d_2, \langle s_m, s_k, s_l, F_m\rangle, \langle \widehat{E}_{\widetilde{m}}, \widehat{\widetilde{T}}, \widehat{\widetilde{V}}, \widehat{\widetilde{W}}, \mu_1\rangle\}$

Game 7:

We modify $\mathsf{G}_6^{\mathcal{A}}$ into $\mathsf{G}_7^{\mathcal{A}}$ by changing $\mathcal{I}_6^{\phi}$ into $\mathcal{I}_7^{\phi}$. Note that $\mathcal{I}_7^{\phi}$ is same as $\mathcal{I}_6^{\phi}$ except that

- Given $\langle \mathtt{advance}, \rho, msg\rangle$, where $\rho \in \{L, R\}$. If $\rho = L$, $\mathcal{I}_7^{\phi}$ operates identically as $\mathcal{I}_6^{\phi}$; but if $\rho = R$, $\mathcal{I}_7^{\phi}$ operates similarly as the case $\rho = L$ with $\widetilde{m} = \widetilde{m}_1$, i.e. runs the same operations for the right user instantiation U^R.

Compute the Statistical Distance:

We prove in Game 0 and Game 1, $|\Pr[\mathbf{E}_0] - \Pr[\mathbf{E}_1]|$ is negligible. Observe that, for the probability distributions of the right user instantiations $[\mathsf{U}^R]_0$, $[\mathsf{U}^R]_1$ are identical. We still need to show for the left user instantiations $[\mathsf{U}^L]_0$, $[\mathsf{U}^L]_1$, the statistical distance of the probability distributions is negligible. First, we prove the statistical distance of $[s_m]_0$ and $[s_m]_1$ are negligible. Observe that in both games $m \in [0, 2^{\nu_p}]$, $\widehat{m} \in \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}]$, $d_1 \xleftarrow{\mathsf{r}} \{0,1\}^{\lambda_1}$. We can obtain that the statistical distance of the random variables $[s_m]_0 = \widehat{m} - d_1 \cdot m$ and $[s_m]_1 \xleftarrow{\mathsf{r}} \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}]$ is less than $2^{-\lambda_0-1}$. Then we can observe that $[F_m]_0$ and $[F_m]_1$, $[s_k]_0$ and $[s_k]_1$, $[s_l]_0$ and $[s_l]_1$ are identically distributed. So the statistical distance of $[s_m, s_k, s_l, F_m]_0$ and $[s_m, s_k, s_l, F_m]_1$ is $2^{-\lambda_0-1}$. From the equivocal property of the Pedersen commitment scheme, we know the distribution of $\{\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\}$ in Game 1 is identical to that in Game 0. So the statistical distance of the two games is $2^{-\lambda_0-1}$, i.e. $|\Pr[\mathbf{E}_0] - \Pr[\mathbf{E}_1]| \leq 2^{-\lambda_0-1}$. Use the similar argument, we can show in Game 1 and Game 2, $|\Pr[\mathbf{E}_1] - \Pr[\mathbf{E}_2]| \leq 2^{-\lambda_0-1}$.

Now we prove in Game 2 and Game 3, under the DLOG assumption, $|\Pr[\mathbf{E}_2] - \Pr[\mathbf{E}_3]|$ is negligible. From Lemma 4.4, in Game 2, if the user instantiation can verify the verification equations successfully, then the generated signature is $\sigma = (a, b = a^y, c = a^{x+mxy})$ except probability $2^{-\lambda_2}$. And in Game 3, signature σ is generated as above without any error probability. Consider there are two user instantiations. So, $|\Pr[\mathbf{E}_2] - \Pr[\mathbf{E}_3]| \leq 2^{-\lambda_2+1}$.

We prove in Game 3 and Game 4, under the DCR assumption, $|\Pr[\mathbf{E}_3] - \Pr[\mathbf{E}_4]|$ is negligible. Observe that, the probability distributions of the right user instantiations $[\mathsf{U}^R]_3$, $[\mathsf{U}^R]_4$ are identical. For the left user instantiations $[\mathsf{U}^L]_3$, $[\mathsf{U}^L]_4$, under the DCR assumption, $[E_m]_3$ and $[E_{\widetilde{m}}]_4$ are indistinguishable. So, $|\Pr[\mathbf{E}_3] - \Pr[\mathbf{E}_4]| \leq \mathsf{Adv}_{\mathsf{DCR}}$. By the similar argument, we can obtain $|\Pr[\mathbf{E}_4] - \Pr[\mathbf{E}_5]| \leq \mathsf{Adv}_{\mathsf{DCR}}$.

Next we prove in Game 5 and Game 6, under the DLDH assumption, $|\Pr[\mathbf{E}_5] - \Pr[\mathbf{E}_6]|$ is negligible. Observe that, for the probability distributions of the right user instantiations $[\mathsf{U}^R]_5$, $[\mathsf{U}^R]_6$ are identical, and for the left user instantiations $[\mathsf{U}^L]_5$, $[\mathsf{U}^L]_6$, under the DLDH assumption, $[T, V, W]_5$ and $[\widetilde{T}, \widetilde{V}, \widetilde{W}]_6$ are indistinguishable. So, $|\Pr[\mathbf{E}_5] - \Pr[\mathbf{E}_6]| \leq \mathsf{Adv}_{\mathsf{DLDH}}$. By the similar argument, we can get $|\Pr[\mathbf{E}_6] - \Pr[\mathbf{E}_7]| \leq \mathsf{Adv}_{\mathsf{DLDH}}$.

In Game 7, ϕ is not used, so the adversary $\mathcal{A}$ has only probability $\frac{1}{2}$ to win the game, i.e. $\Pr[\mathbf{E}_7] = \frac{1}{2}$. Based on the argument above, we can get

$$\begin{aligned}\left|\Pr[\mathbf{E}_0] - \tfrac{1}{2}\right| &= |\Pr[\mathbf{E}_0] - \Pr[\mathbf{E}_7]| = |\sum_{j=0}^{6} \Pr[\mathbf{E}_j] - \Pr[\mathbf{E}_{j+1}]| \leq \sum_{j=0}^{6} |\Pr[\mathbf{E}_j] - \Pr[\mathbf{E}_{j+1}]| \\ &= 2^{-\lambda_0 - 1} + 2^{-\lambda_0 - 1} + 2^{-\lambda_2 + 1} + \mathsf{Adv}_{\mathsf{DCR}} + \mathsf{Adv}_{\mathsf{DCR}} + \mathsf{Adv}_{\mathsf{DLDH}} + \mathsf{Adv}_{\mathsf{DLDH}} \\ &= 2^{-\lambda_0} + 2^{-\lambda_2 + 1} + 2\mathsf{Adv}_{\mathsf{DCR}} + 2\mathsf{Adv}_{\mathsf{DLDH}}\end{aligned}$$

which is negligible. This completes the proof of blindness.

□

Remark 4.6. Both unforgeability and blindness depend on the DLOG assumption as well. In Theorem 4.3 and Theorem 4.5, we do not include the DLOG assumption, because the DLOG assumption can be implied from the LRSW assumption or the DLDH assumption. Note that in our scheme, the size of elliptic curve groups $\mathbf{G}$ and $\mathbb{G}$ is same.

5 Extensions and Variants

Stronger Blindness Property. The formal model of Section 3 can be strengthened with respect to the blindness property by allowing to the malicious signer to select the public/secret-key pair $PK_{\mathsf{S}}, SK_{\mathsf{S}}$ instead of selecting these values honestly as in [JLO97]. It is simple to modify Definition 3.2 to include such stronger adversaries; this strengthening of the [JLO97] model has been observed recently in [Oka06, ANN06] as well. Our scheme can be easily modified to achieve such stronger blindness as follows: we have the signer make an extractable commitment on the signing key $SK_{\mathsf{S}} = \langle x, y \rangle$ (using the same public-parameters employed for the users' extractable commitment) and prove that such commitment is consistent with the computation of the encryption ψ. In the blindness proof, the oracle $\mathcal{I}^{\phi}$ can extract the signing key and the security proof remains essentially unchanged; note though that unforgeability as argued in Theorem 4.3 will also rely on the DCR assumption.

Public-Tagging and Partial Blindness. We construct an extension of our blind signature that allows the "public-tagging" of a message that is blindly signed. Public-tagging of blindly signed messages gives rise to what is called a partially blind signature [AF96]: the signer knows a portion of the message that he is about to sign. Public-tagging is useful as it allows the signer to keep the same public-key and issue blind signatures for different purposes (e.g., a bank may issue e-coins that are publicly-tagged blind signatures, and the tagging will correspond to the denomination, i.e., there will be a different tag for each coin denomination). It should be stressed that in a blind signature with public tagging the blindness property is only enforced within blind signatures with the same public-tag. The unforgeability property on the other hand remains identical. We develop a public-tagging mechanism for our basic scheme. The key idea is the following: we replace the underlying digital signature of [CL04] with the two message-block extended version (Scheme C for two messages in [CL04]). In the modified blind signature the messages will be of the form $\langle m, \mathsf{tag} \rangle$. The public information tag is included into pub. Here $\mathsf{tag} \in [0, 2^{\nu_p}]$. Note that the exact choice for the value of tag is negotiated by the signer and the user outside of the signing protocol.

In the modified signature that we use, the public and secret-key of the signer are modified and the values $PK_{\mathsf{S}} = \langle X, Y \rangle$ and $SK_{\mathsf{S}} = \langle x, y \rangle$ they are substituted with $PK_{\mathsf{S}} = \langle X, Y, Z \rangle$, $SK_{\mathsf{S}} = \langle x, y, z \rangle$, where $X = g^x$, $Y = g^y$, $Z = g^z$. Signing a message $\langle m, \mathsf{tag} \rangle$ corresponds to the following operation: select a random $a \in \mathbb{G}$ and output the signature $\sigma = \langle a, a^z, a^y, a^{yz}, a^{x+xym+xyz \cdot \mathsf{tag}} \rangle$. The modified signature has the following verification process: Given a message-signature pair $(m, \mathsf{tag}; \sigma)$, where $\sigma = \langle a, A, b, B, c \rangle$, we can verify it by the verification equations: $e(a, Z) = e(g, A)$; $e(a, Y) = e(g, b)$ and $e(A, Y) = e(g, B)$ and $e(X, a)e(X, b)^m e(X, B)^{\mathsf{tag}} = e(g, c)$.

The detailed partially signing protocol is similar to our basic signing protocol (i.e., it retains the four-move structure with short communication) and is shown in detail in Figure 3. We can obtain the security theorem below:

Theorem 5.1. *Under the LRSW assumption the proposed partially blind signature scheme is unforgeable even if the public-tag is adversarially selected for each signature; Under the DLDH assumption and the DCR assumption, the proposed scheme is blind for signatures with the same public-tag.*

To prove the unforgeability property in the theorem above, we can use the similar proof idea in Theorem 4.3. Consider that the scheme above is based on Camenisch-Lysyanskaya two message-block signature [CL04], we reduce the unfogerability to the security of the Camenisch-Lysyanskaya two message-block signature which is also based on the LRSW assumption. Consider tag is fixed across protocol executions, we can also use the similar proof idea in Theorem 4.5 to show the blindness of the above scheme is based on the DLDH and the DCR assumptions.

Acknowledgement

We thank one of the reviewers of Eurocrypt 2006 for pointing out a flaw in the design of a previous version of our blind signature protocol.

$\mathtt{crs} = \langle Q, \mathbf{g}, \mathbf{h}, \mathbf{G}, \mathcal{H}; \mathsf{n}, \mathsf{g}\rangle$; $\mathtt{pub} = \langle p, g, \mathbb{G}, \mathbb{G}_T, e; \mathsf{tag}\rangle$; $PK_{\mathsf{S}} = \langle X, Y, Z\rangle$

$\boxed{\mathsf{U}}$		$\boxed{\mathsf{S}}$
$MSG = \langle m\rangle$, $m \in [0, 2^{\nu_p}]$		$SK_{\mathsf{S}} = \langle x, y, z\rangle$
$(PK_{\mathsf{U}}, SK_{\mathsf{U}}) \leftarrow \mathtt{gen}^{LE}(1^\lambda)$ $PK_{\mathsf{U}} = \langle t, v, w\rangle$, $SK_{\mathsf{U}} = \langle \delta, \xi\rangle$ $\widehat{m} \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\nu_p}]$, $A_m, B_m \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$ $\alpha, k, l, \widehat{k}, \widehat{l} \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}\backslash\{1\}$, $\mu_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$ $E_m = \mathsf{g}^m (A_m)^{\mathsf{n}} \bmod \mathsf{n}^2$ $\widehat{E}_m = \mathsf{g}^{\widehat{m}} (B_m)^{\mathsf{n}} \bmod \mathsf{n}^2$ $T = t^k$, $V = v^l$, $W = \theta^m w^{k+l}$ $\widehat{T} = t^{\widehat{k}}$, $\widehat{V} = v^{\widehat{l}}$, $\widehat{W} = \theta^{\widehat{m}} w^{\widehat{k}+\widehat{l}}$		
$\omega_1 = \mathcal{H}(\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W})$, $C_1 = \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$	$\xrightarrow{PK_{\mathsf{U}}, E_m, \langle\theta, T, V, W\rangle, C_1}$	$d_1 \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_1}$ $\alpha', k', l', \widehat{x}, \widehat{k}', \widehat{l}' \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\mu_2 \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$ $a' = \theta^{\alpha'}$, $A' = \theta^{z\alpha'}$, $b' = \theta^{y\alpha'}$, $B' = \theta^{yz\alpha'}$ $T' = T^{xy\alpha'} t^{k'\alpha'}$, $V' = V^{xy\alpha'} v^{l'\alpha'}$ $W' = W^{xy\alpha'} \theta^{x\alpha' + xyz\alpha' \cdot \mathsf{tag}} w^{k'\alpha' + l'\alpha'}$ $L_T = e(T, b')^{\widehat{x}} e(t, a')^{\widehat{k}'}$ $L_V = e(V, b')^{\widehat{x}} e(v, a')^{\widehat{l}'}$ $L_W = (e(W, b') e(\theta, a'(B')^{\mathsf{tag}}))^{\widehat{x}} \cdot e(w, a')^{\widehat{k}'+\widehat{l}'}$ $\omega_2 = \mathcal{H}(a', A', b', B', T', V', W', L_T, L_V, L_W)$
$d_2 \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_2}$, $s_m = \widehat{m} - d_1 m$ (in $\mathbb{Z}$) $s_k = \widehat{k} - d_1 k$, $s_l = \widehat{l} - d_1 l$ $F_m = B_m (A_m)^{-d_1} \bmod \mathsf{n}$	$\xleftarrow{d_1, C_2}$	$C_2 = \mathbf{g}^{\omega_2}\mathbf{h}^{\mu_2}$
	$\xrightarrow{d_2, \langle s_m, s_k, s_l, F_m\rangle, \langle \widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W}, \mu_1\rangle}$	$E_m \in^? \mathbb{Z}_{\mathsf{n}^2}^*$, $s_m \in^? \pm[0, 2^{\lambda_0+\lambda_1+\nu_p+1}]$ $\omega_1 = \mathcal{H}(\widehat{E}_m, \widehat{T}, \widehat{V}, \widehat{W})$, $C_1 =^? \mathbf{g}^{\omega_1}\mathbf{h}^{\mu_1}$ $\widehat{E}_m =^? \mathsf{g}^{s_m} (F_m)^{\mathsf{n}} (E_m)^{d_1} \bmod \mathsf{n}^2$ $\widehat{T} =^? t^{s_k} T^{d_1}$, $\widehat{V} =^? v^{s_l} V^{d_1}$ $\widehat{W} =^? \theta^{s_m} w^{s_k+s_l} W^{d_1}$ $s_x = \widehat{x} - d_2 x$,
	$\xleftarrow[\langle a', A', b', B', T', V', W', L_T, L_V, L_W, \mu_2\rangle]{\langle s_x, s_{k'}, s_{l'}\rangle}$	$s_{k'} = \widehat{k}' - d_2 k'$, $s_{l'} = \widehat{l}' - d_2 l'$
$\omega_2 = \mathcal{H}(a', A', b', B', T', V', W', L_T, L_V, L_W)$ $C_2 =^? \mathbf{g}^{\omega_2}\mathbf{h}^{\mu_2}$, $e(a', Z) =^? e(A', g)$ $e(a', Y) =^? e(b', g)$, $e(A', Y) =^? e(B', g)$ $L_T =^? e(T, b')^{s_x} e(t, a')^{s_{k'}} e(T', \theta)^{d_2}$ $L_V =^? e(V, b')^{s_x} e(v, a')^{s_{l'}} e(V', \theta)^{d_2}$ $L_W =^? (e(W, b') e(\theta, a'(B')^{\mathsf{tag}}))^{s_x} \cdot e(w, a')^{s_{k'}+s_{l'}} e(W', \theta)^{d_2}$ $a = (a')^\alpha$, $b = (b')^\alpha$, $c = \left(\frac{W'}{T'^\delta V'^\xi}\right)^\alpha$ $A = (A')^\alpha$, $B = (B')^\alpha$ $\sigma = \langle a, A, b, B, c\rangle$		
output $(m, \mathsf{tag}; \sigma)$		

Figure 3: Partially blind signature generation protocol.

References

[Abe01] Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In Birgit Pfitzmann, editor, *EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 136–151. Springer, 2001.

[AF96] Masayuki Abe and Eiichiro Fujisaki. How to date blind signatures. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT 1996*, volume 1163 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1996.

[ANN06] Michel Abdalla, Chanathip Namprempre, and Gregory Neven. On the (im)possibility of blind message authentication codes. In David Pointcheval, editor, *CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2006.

[AO00] Masayuki Abe and Tatsuaki Okamoto. Provably secure partially blind signatures. In Mihir Bellare, editor, *CRYPTO 2000*, volume 1880 of *Lecture Notes in Computer Science*, pages 271–286. Springer, 2000.

[AO01] Masayuki Abe and Miyako Ohkubo. Provably secure fair blind signatures with tight revocation. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *Lecture Notes in Computer Science*, pages 583–602. Springer, 2001.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew K. Franklin, editor, *CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.

[BNPS01] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. In Paul F. Syverson, editor, *Financial Cryptography 2001*, volume 2339 of *Lecture Notes in Computer Science*, pages 319–338. Springer, 2001.

[CDP94] Lidong Chen, Ivan Damgård, and Torben P. Pedersen. Parallel divertibility of proofs of knowledge (extended abstract). In Alfredo De Santis, editor, *EUROCRYPT 1994*, volume 950 of *Lecture Notes in Computer Science*, pages 140–155. Springer, 1994.

[Cha82] David Chaum. Blind signatures for untraceable payments. In David Chaum, Ronald L. Rivest, and Alan T. Sherman, editors, *CRYPTO 1982*, pages 199–203. Plemum Press, 1982.

[CKW04] Jan Camenisch, Maciej Koprowski, and Bogdan Warinschi. Efficient blind signatures without random oracles. In Carlo Blundo and Stelvio Cimato, editors, *SCN 2004*, volume 3352 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 2004.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Matthew K. Franklin, editor, *CRYPTO 2004*, volume 3152 of *Lecture Notes in Computer Science*, pages 56–72. Springer, 2004.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *STOC 2002*, pages 494–503, 2002. Full version at `http://www.cs.biu.ac.il/~lindell/PAPERS/uc-comp.ps`.

[Dam88] Ivan Damgård. Payment systems and credential mechanisms with provable security against abuse by individuals. In Shafi Goldwasser, editor, *CRYPTO 1988*, volume 403 of *Lecture Notes in Computer Science*, pages 328–335. Springer, 1988.

[Dam00] Ivan Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In *EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 418–430. Springer, 2000.

[FOO92] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In Jennifer Seberry and Yuliang Zheng, editors, *ASIACRYPT 1992*, volume 718 of *Lecture Notes in Computer Science*, pages 244–251. Springer, 1992.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, 1987.

[JLO97] Ari Juels, Michael Luby, and Rafail Ostrovsky. Security of blind digital signatures (extended abstract). In Burton S. Kaliski Jr., editor, *CRYPTO 1997*, volume 1294 of *Lecture Notes in Computer Science*, pages 150–164. Springer, 1997.

[Kim04] Kwangjo Kim. Lessons from Internet voting during 2002 FIFA WorldCup Korea/Japan(TM). In *DIMACS Workshop on Electronic Voting – Theory and Practice*, 2004.

[Lin03] Yehuda Lindell. Bounded-concurrent secure two-party computation without setup assumptions. In *STOC 2003*, pages 683–692. ACM, 2003. Full version at `http://www.cs.biu.ac.il/~lindell/PAPERS/conc2party-upper.ps`.

[LRSW99] Anna Lysyanskaya, Ronald L. Rivest, Amit Sahai, and Stefan Wolf. Pseudonym systems. In Howard M. Heys and Carlisle M. Adams, editors, *Selected Areas in Cryptography 1999*, volume 1758 of *Lecture Notes in Computer Science*, pages 184–199. Springer, 1999.

[Oka92] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest F. Brickell, editor, *CRYPTO 1992*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53. Springer, 1992.

[Oka06] Tatsuaki Okamoto. Efficient blind and partially blind signatures without random oracles. In Shai Halevi and Tal Rabin, editors, *TCC 2006*, volume 3876 of *Lecture Notes in Computer Science*, pages 80–99. Springer, 2006. An extended version at `http://eprint.iacr.org/2006/102/`.

[OO89] Tatsuaki Okamoto and Kazuo Ohta. Divertible zero knowledge interactive proofs and commutative random self-reducibility. In Jean-Jacques Quisquater and Joos Vandewalle, editors, *EUROCRYPT 1989*, volume 434 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 1989.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 1999.

[Ped91] Torben P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In Joan Feigenbaum, editor, *CRYPTO 1991*, volume 576 of *Lecture Notes in Computer Science*, pages 129–140. Springer, 1991.

[Poi98] David Pointcheval. Strengthened security for blind signatures. In Kaisa Nyberg, editor, *EUROCRYPT 1998*, volume 1403 of *Lecture Notes in Computer Science*, pages 391–405. Springer, 1998.

[PS96] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT 1996*, volume 1163 of *Lecture Notes in Computer Science*, pages 252–265. Springer, 1996.

[PS97] David Pointcheval and Jacques Stern. New blind signatures equivalent to factorization (extended abstract). In *ACM Conference on Computer and Communications Security*, pages 92–99, 1997.

[PW91] Birgit Pfitzmann and Michael Waidner. How to break and repair a "provably secure" untraceable payment system. In Joan Feigenbaum, editor, *CRYPTO 1991*, volume 576 of *Lecture Notes in Computer Science*, pages 338–350. Springer, 1991.

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 162–167, Toronto, Ontario, Canada, 1986. IEEE.