Two-Round Concurrent Blind Signatures without Random Oracles

Aggelos Kiayias* Hong-Sheng Zhou*

November 29, 2005

Abstract

We present the first blind signature scheme that is efficient and provably secure without random oracles under concurrent attacks utilizing only two rounds 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 guarranteed by the LRSW assumption while the blindness property of our scheme is guarranteed by the Decisional Linear Diffie Hellman assumption.

We prove our construction secure under the above assumptions as well as the DCR and DLOG assumptions in the concurrent attack model of Juels, Luby and Ostrovsky from Crypto '97. Our construction is the first scheme that instantiates the security definition of Juels et al. with an efficient construction in the standard model. We consider various modifications to our basic protocol that inlude a blind signature scheme with revokable blindness as well as a blind signature that incorporates a "public-tagging" mechanism. The latter extension of our scheme gives rise to a partially blind signature with the same efficiency and security properties as our basic scheme.

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

Contents

List of Tables

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 (essentially 2-rounds since these protocols have the signer go first which typically is a server) that were proven secure in the random oracle model under the discrete-logarithm assumption assuming only logarithmically many messages were transmitted by the user. This result was later improved to polynomially many messages but 5 communication moves [Poi98] and the round complexity was finally decreased to 3 moves (essentially 2 rounds) and polynomially many messages in [AO01, Abe01]. A single round 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 in the context of blind signatures since in implementations of blind signatures in e-voting and e-cash schemes, see e.g., [Cha82, FOO92, Kim04], the signer is in fact 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. Juels et al. [JLO97] demonstrated the feasibility of concurrently secure blind signatures by providing a construction based on generic secure function evaluation. Naturally, this protocol was not efficient, but it was the first instantiation of a secure blind signature (and in fact in the concurrent setting) that did not require random oracles. Till today this (inefficient) construction remains the only blind signature known that is secure in the concurrent setting without random oracles. More recently, with respect to blind signatures without random oracles Camenisch et al. [CKW04] considered a weaker model than that of [JLO97] that only allowed sequential attacks and presented an 8-move blind signature scheme based on the Strong-RSA assumption leaving as open problem the possibility of achieving concurrent security.

We conclude this brief overview on the blind signature primitive by observing that to date no *efficient* blind signature construction is known that is provably secure in the model of Juels et al. without random oracles; providing an efficient scheme was discussed in [JLO97] and was left as an open problem. We settle this open question in this work.

1.1 Our Main Contribution

We present an efficient construction for a blind signature scheme utilizing only two rounds of communication and a full proof of security of this construction in the concurrent model of Juels et al. [JLO97]. The two round interaction between the user and the signer in the signing protocol requires overall communication not exceeding 2 Kbytes (about 10.5 Kbits to be precise) for a full signature generation. In the process of presenting our security proof we in fact reformulate and use the [JLO97] model in a stronger fashion since we allow the adversary to select the public-key of the underlying signature in the blindness attack (whereas in previous work such key was assumed part of the trusted setup). Achieving this level of efficiency while simultaneously maintaining provability in the strong model of [JLO97] 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] for the purpose of designing group signatures. 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 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 since it is based on those that we manage the efficient generation of non-adversarial randomness between the mutually distrustful players. Finally, in order to prove security under concurrent attacks the homomorphic encryption based interaction needs to be paired with an extractable commitment for the user's special Linear Encryption ciphertext. Further simulation requirements require an equivocable commitment to be used for ensuring that no information leakage occurs from the user to the signer.

Note that the resulting signature from the signing protocol is about half the size of an RSA based Chaum blind signature. Table 1 compares the round complexity of our construction in comparison to previous blind signature schemes. The construction is proven to satisfy the two properties of [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] and the discrete-logarithm assumption over prime order modular groups.

1.2 Extensions and Variants

We present a variant of our construction where the blindness property is strengthened and relies only on the Decision Linear Diffie-Hellman assumption. The requirement for the DCR assumption is transferred to the unforgeability property. In this stronger blindness variant of our scheme it is ensured that the message that is submitted by the user is hidden only with a public-key that is in the control of the user and is short-lived (i.e., used only for a single signing protocol as opposed to throughout the life-time of the common-reference string). Thus blindness, while still claimed in the computational sense, it is only based on user-selected secrets.

The second variant of our scheme we modify the blindness property but in the opposite direction. In this case, we include a trusted third party in the blind signature setup, that is capable of removing the blindness of a user given a signing protocol transcript. We call this capability "blindness revocation" and is a useful

paper	# of signatures	# of moves	properties	model	setting
[PS96]	Log(ploy)	3	plain	ROM	sequential
[PS97]	Log(poly)	3	plain	ROM	sequential
[AO00]	Log(poly)	3	partial	ROM	sequential
[Poi98]	ploy	5	plain	ROM	sequential
[Abe01]	poly	3	plain	ROM	sequential
[AO01]	poly	4(2-round)	fair	ROM	concurrent
[BNPS01]	poly	2(1-round)	plain	ROM	concurrent
[JLO97]	poly	poly	plain	plain	concurrent
[CKW04]	poly	8(4-round)	plain	plain	sequential
Our Scheme	poly	4(2-round)	plain/partial	plain	concurrent

Table 1: Comparison of present work to previous blind signatures

property in cases where for arbitration purposes the confidentiality of a certain signing transcript must be lifted. The trusted third party operation, provides a type of fairness mechanism for blind signatures, cf. [SPC95].

Finally we 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 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 functionalities.

2 Preliminaries

2.1 Bilinear Groups

Let $\mathbb{G} = \langle g \rangle$ is 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 $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, it holds that $e(u^a, v^b) = e(u, v)^{ab}$ and e is non-trivial, i.e., $e(g, g) \neq 1$.

2.2 Digital Signature

A signature scheme $\mathtt{DS} = (\mathtt{GEN}, \mathtt{SIGN}, \mathtt{VERIFY})$ is defined by the three following algorithms:

- The key generation algorithm $\mathtt{GEN}$. On input 1^λ, the algorithm $\mathtt{GEN}$ outputs a pair (pk, sk) of matching public key and secret key.
- The signing algorithm $\mathtt{SIGN}$. Given a message m and a pair of matching public key and secret key (pk, sk), $\mathtt{SIGN}$ generates the corresponding signature σ.
- The verification algorithm $\mathtt{VERIFY}$. Given a message-signature pair (m, σ) and a public key pk, $\mathtt{VERIFY}$ test if σ is a valid signature of m with respect to pk.

2.2.1 Camenisch-Lysyanskaya Signature

Camenisch and Lysyanskaya [CL04] constructed a plain signature (call it CL-signature for short) and proved it secure in the standard model.

- The key generation algorithm $\texttt{GEN}^{CL}$: generate the bilinear group parameter $(p, \mathbb{G}, \mathbb{G}_T, g, e)$; then choose $x, y \in \mathbb{Z}_p$, 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.

We will use the CL-signature to construct blind signatures. The underlying assumption of CL-signatures is called the LRSW assumption, which was introduced by Lysyanskaya et al. [LRSW99]. It was also shown, in the same paper, 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 $X, Y \in \mathbb{G}, X = g^x, Y = g^y$. Let $O_{X,Y}(.)$ be an oracle that, on input a value $m \in \mathbb{Z}_p$, outputs a triple (a, b, c) such that $b = a^y$, and $c = a^{x+mxy}$ where $a \xleftarrow{\mathtt{r}} \mathbb{G}$. Then for all probabilistic polynomial time adversaries $\mathcal{A}$,

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

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

2.3 Public-key Encryption

A public-key encryption scheme $\texttt{PKE} = (\texttt{GEN}, \texttt{ENCRYPT}, \texttt{DECRYPT})$ is defined by the three following algorithms:

- The key generation algorithm GEN. On input 1^λ, the algorithm GEN outputs a pair (pk, sk) of matching public key and secret key.

- The encryption algorithm ENCRYPT. Given a message m and a public key pk, ENCRYPT generates a ciphertext C of m.

- The decryption algorithm DECRYPT. Given a ciphertext C and the secret key sk, DECRYPT gives back the plaintext m.

2.3.1 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 $\texttt{GEN}^{LE}$: the public key pk is a triple of generators $u, v, w \in \mathbb{G}$ and the secret key sk is the exponents $x, y \in \mathbb{Z}_p$ such that $u^x = v^y = w$.

- The encryption algorithm $\texttt{ENCRYPT}^{LE}$: to encrypt a message $m \in \mathbb{G}$, choose random values $a, b \in \mathbb{Z}_p$, and output the triple $(u^a, v^b, m \cdot w^{a+b})$.

- The decryption algorithm $\texttt{DECRYPT}^{LE}$: given an encryption (U, V, W), we recover the plaintext $m = \texttt{DECRYPT}^{LE}_{sk}(U, V, W) = \frac{W}{U^x \cdot V^y}$

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

Definition 2.2 (Decision Linear Diffie-Hellman Problem in $\mathbb{G}$). Given $u, v, w, u^\alpha, v^\beta, w^\gamma \in \mathbb{G}$ 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

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

Assumption 2.3 (Decision Linear Diffie-Hellman Assumption). We say that the Decision Linear Diffie-Hellman assumption holds in $\mathbb{G}$ if no PPT algorithm $\mathcal{A}$ has non-negligible advantage $\mathsf{Adv}^{\mathcal{A}}_{\mathsf{DLDH}}$.

2.3.2 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 $\mathsf{p}, \mathsf{q} > 2$, $\mathsf{p} \neq \mathsf{q}$, $|\mathsf{p}| = |\mathsf{q}|$ 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 $\mathtt{ENCRYPT}^{Pai}$: 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 $M = \mathtt{ENCRYPT}^{Pai}_{pk}(m, \zeta) = \mathsf{g}^m \zeta^\mathsf{n} \bmod \mathsf{n}^2$.
- The decryption algorithm $\mathtt{DECRYPT}^{Pai}$: given a ciphertext M, observe that $M^{\pi K} = \mathsf{g}^{m \cdot \pi K} \cdot \zeta^{\mathsf{n} \cdot \pi K} = \mathsf{g}^{m \cdot \zeta K \bmod \mathsf{n}} \cdot \zeta^{\mathsf{n} \cdot \pi K \bmod \mathsf{n}\pi} = \mathsf{g}^{m \bmod \mathsf{n}} \cdot \zeta^{0 \bmod \mathsf{n}\pi} = \mathsf{g}^m = 1 + m\mathsf{n} \bmod \mathsf{n}^2$. Thus, it is possible to recover $m = \frac{(M^{\pi K} \bmod \mathsf{n}^2) - 1}{\mathsf{n}} \bmod \mathsf{n}$.

The cryptosystem above has been proven semantically secure if and only if the Decisional Composite Residuosity (DCR) assumption is true.

Assumption 2.4 (Decisional Composite Residuosity Assumption). There is no PPT distinguisher for n-th residues modulo n^2. In other words, there is no PPT adversary that can distinguish $\mathbb{Z}^\mathsf{n}_{\mathsf{n}^2}$ from $\mathbb{Z}^*_{\mathsf{n}^2}$, where $\mathbb{Z}^\mathsf{n}_{\mathsf{n}^2} \stackrel{\mathsf{def}}{=} \{z \in \mathbb{Z}^*_{\mathsf{n}^2} \mid \exists y \in \mathbb{Z}^*_{\mathsf{n}^2} : z = y^\mathsf{n} \bmod \mathsf{n}^2\}$.

2.4 Commitment

A commitment scheme is a two-stage interactive protocol between two parties, the *committer* holding a message m and a random string r, and the *receiver*. In the first stage, called the commit-stage, the committer gives some information derived from m, r to the receiver such that (1) the receiver can not obtain any information about m, i.e. the commitment is *hiding* and (2) the committer cannot change his mind about m later, i.e. the commitment is *binding*. In the second stage, called the open-stage, the committer sends m, r to the receiver, who verifies that m, r match the communication of the first stage. In general, the committer will use an algorithm $commit_{pk}$ which is keyed by a public key pk to compute $c \leftarrow commit_{pk}(m, r)$, and send c to the receiver; to open the commitment, the committer just sends m, r to the receiver who checks if $c = commit_{pk}(m, r)$. The hiding property means given c the receiver does not learn m, and the binding property means the committer cannot change his mind by computing m', r' such that $c = commit_{pk}(m', r')$ and $m' \neq m$.

2.4.1 Extractable Commitment

In an *extractable* commitment, there is a trapdoor information ex_{pk} is associated to each public key pk which allows the trapdoor owner to compute m from any $commit_{pk}(m, r)$. In our bind signature scheme, the user sends $\widehat{m}$, the blinded form of his message m to the signer, and the signer manipulates $\widehat{m}$ into a scrambled signature $\widehat{\sigma}$. When the user gets $\widehat{\sigma}$, he will transform $\widehat{\sigma}$ into a blind signature σ for the message m. The user should also commit to m when sending $\widehat{m}$, by sending the signer a corresponding commitment $commit_{pk}(m, r)$ for m. Obviously, the user should be restricted to choose both $\widehat{m}$ and $commit_{pk}(m, r)$ consistently over same m. When we prove the unforgeability of the scheme, the adversary controls a multitude of users that run concurrent blinding sessions with the signer. In this case, we want to be able to simulate the adversary and attack the unforgeability of the underlying signature. While it is possible to use rewinding to extract m from the commitment and then simulate the remaining part with such m this solution is not suitable in the concurrent setting. Using an extractable commitment properly paired with the remaining components of our scheme we extract m without rewinding (by setting things up such that the simulator knows the trapdoor ex_{pk}).

2.4.2 Equivocal Commitment

In an *equivocable* commitment, there is a trapdoor information eq_{pk} that is associated to each public key pk which allows the committer to change his mind. As mentioned before, in our blind signature scheme, we need a sound proof to guarantee that the blinded form $\widehat{m}$ and the commitment $commit_{pk}(m, r)$ are corresponding to the same message m. In the blindness attack against the scheme the signer is controlled by the adversary and engages concurrent user sessions with the aim to extract information about the employed messages. It follows that all interactions of the user during the signing protocol should be zero-knowledge. We will employ Damgård's technique [Dam00] over a 3-move Σ-protocol to ensure zero-knowledge: we use the equivocable variant of a Pedersen multi-commitment [Ped91] to "wrap up" a general 3-move proof of knowledge that the blinded form and the commitment are over same message.

3 Formal Model for Blind Signatures

In this section, we revisit in detail the formal model for blind signatures as introduced in [JLO97].

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 $(\mathcal{S}, \mathcal{U})$ and two algorithms (GEN,VERIFY). Here $\mathcal{S}$ denotes the signer, and $\mathcal{U}$ the user.

- $\texttt{GEN}(1^\lambda)$ 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.
- $\mathcal{S}(pk, sk)$ and $\mathcal{U}(pk, m)$ are 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 a key generation algorithm. Additionally $\mathcal{S}$ is given on his input tape a corresponding secret key sk and $\mathcal{U}$ is given on his input tape a message m, where the length of all inputs must be polynomial in the security parameter 1^λ of the key generation algorithm. Both $\mathcal{U}$ and $\mathcal{S}$ engage in the interactive

protocol of some polynomial number of rounds. At the end of this protocol $\mathcal{S}$ outputs either *completed* or *not-completed* and $\mathcal{U}$ outputs either σ or $\perp$.

- $\texttt{VERIFY}(m, \sigma, pk)$ 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 $\mathcal{S}$ and $\mathcal{U}$ follow the protocol then $\mathcal{S}$ always outputs *completed*, and if the output of the user is σ then the $\texttt{VERIFY}(m, \sigma, pk) = 1$.

3.2 Blindness and Unforgeability

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

We stress that our formal model is stronger compared to that of [JLO97] as it does allow for adversarial selection of the public-key of the signing algorithm in a blindness attack; on the contrary, [JLO97] assumed a trusted selection for public and signing key pk, sk in their formulation of the blindness attack.

The two players of the signing protocol will have an additional input $\texttt{t}$ that will include two components, $ComInfo$ and CRS. $ComInfo$ will include some joint information that has been decided in advance, e.g., some modular group that the players wish to use, or other public-information. The string CRS will contain some public-elements that will be used in the scheme. The two components $ComInfo$ and CRS that will be jointly denoted by $\texttt{t}$ will be generated by a procedure $\texttt{K}$ that will produce the values $\texttt{t}, \tau$. The value τ will be contain possibly some trapdoor information and will only be available to the simulator of the protocol.

Definition 3.2 (Blindness). Let $\phi \xleftarrow{\texttt{r}} \{0,1\}$ (note: ϕ will be a random bit which is kept secret from the adversary). We define an oracle $\mathcal{I}^{\phi}$ which simulates two user instantiations $\mathcal{U}^L$ and $\mathcal{U}^R$ (note: an adversary $\mathcal{A}$ will be communicating with this oracle trying to predict ϕ).

- Given $\langle \texttt{challenge}, m_0, m_1, pk \rangle$, the oracle $\mathcal{I}^{\phi}$ simulates $\mathcal{U}^L$ (resp. $\mathcal{U}^R$) with public-key pk and message m_{ϕ} (resp. $m_{1-\phi}$). 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 $\mathcal{U}^L$ (resp. $\mathcal{U}^R$).

- 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 $\mathcal{U}^{\rho}$ with msg till $\mathcal{U}^{\rho}$ either terminates or returns a response to the signer. If $\mathcal{U}^{\rho}$ 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 queries can be executed several times depending on the number of rounds 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 $\mathcal{U}^L$ (resp. $\mathcal{U}^R$) with msg^L (resp. msg^R) till $\mathcal{U}^L$ (resp. $\mathcal{U}^R$) either terminates or returns an output. If both user instantiations return outputs, then the oracle *returns these outputs* to $\mathcal{A}$, otherwise returns $(\perp, \perp)$.

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

$$\mathsf{Adv}^{\mathcal{A}}_{\mathsf{blind}}(\lambda) = \left| \Pr\left[\mathcal{A}^{\mathcal{I}^{\phi}(\texttt{t},\tau)}(1^{\lambda}, \texttt{t}) = \phi \,:\, \phi \xleftarrow{\texttt{r}} \{0,1\}; (\texttt{t}, \tau) \leftarrow \texttt{K}(1^{\lambda}) \right] - \frac{1}{2} \right|$$

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 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 $\mathcal{S}$ with msg till $\mathcal{S}$ either terminates or returns a response. If the signer instance returns a response to the user, $\mathcal{I}$ returns this with the session identifier as an answer to the oracle query. The oracle $\mathcal{I}$ keeps a database with the state of $\mathcal{S}$ for the session identifier sid; the state includes all coin tosses of $\mathcal{S}$, 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 $\mathcal{S}$ for the session with identifier sid (if session sid exists). Subsequently, $\mathcal{I}$ writes msg in the communication tape of $\mathcal{S}$ 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 id exists the oracle returns "fail."

The oracle $\mathcal{I}$ has read/write access to a counter l that counts the number 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.

An one-more forgery adversary against the blind signature is a polynomial-time probabilistic machine $\mathcal{A}$ that is given as input $\langle 1^\lambda, pk, \mathsf{t} \rangle$ where $\langle \mathsf{t}, \tau \rangle \leftarrow \mathtt{K}(1^\lambda)$ and $pk, sk \leftarrow \mathtt{GEN}(1^\lambda)$. The adversary $\mathcal{A}$ interacts with $\mathcal{I}(\mathsf{t}, \tau, pk, sk)$ and terminates by returning a sequence of $(m_1, \sigma_1), ..., (m_{l'}, \sigma_{l'})$ where $m_i \neq m_j, 1 \leq i \neq j \leq l'$.
We define the advantage of $\mathcal{A}$ in the above attack by

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

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 assumptions as well as the way that the involved parties, the user and the signer generate their keys.

Common Information. This string $ComInfo$ contains general information about each protocol execution as well as a specific bilinear group parameter $(p, \mathbb{G}, \mathbb{G}_T, g, e)$ of size sufficiently large.

Common Reference String. Next we describe how the common reference string CRS is selected. It includes two parts, CRS_1 and CRS_2. Let p and q be random primes for which $\mathsf{p}, \mathsf{q} > 2$, $\mathsf{p} \neq \mathsf{q}$, $|\mathsf{p}| = |\mathsf{q}|$ and $\gcd(\mathsf{pq}, (\mathsf{p}-1)(\mathsf{q}-1)) = 1$. Let $\mathsf{n} = \mathsf{pq}$, and $\mathsf{g} = (1+\mathsf{n})$. The public key is $\langle \mathsf{n}, \mathsf{g} \rangle$ while the secret key is $\langle \mathsf{p}, \mathsf{q} \rangle$. Set $CRS_1 = \langle \mathsf{n}, \mathsf{g} \rangle$ and $trapdoor_1 = \langle \mathsf{p}, \mathsf{q} \rangle$. Select large primes P, Q such that $P = 2Q+1$, select $h_r \xleftarrow{\mathsf{r}} \mathbb{Z}_P$, $\tau_M, \tau_W, \tau_U, \tau_V \xleftarrow{\mathsf{r}} \mathbb{Z}_Q$, and compute $h_M = h_r^{\tau_M} \bmod P$, $h_W = h_r^{\tau_W} \bmod P$, $h_U = h_r^{\tau_U} \bmod P$, $h_V = h_r^{\tau_V} \bmod P$. Set $CRS_2 = \langle h_M, h_W, h_U, h_V, h_r, P, Q \rangle$, and $trapdoor_2 = \langle \tau_M, \tau_W, \tau_U, \tau_V \rangle$. Now we have $CRS = (CRS_1, CRS_2)$, and discard $trapdoor_1$, $trapdoor_2$. Two one-to-one maps $\psi_1 : \mathbb{Z}_\mathsf{n} \rightarrow \mathbb{Z}_Q$ and $\psi_2 : \mathbb{G} \rightarrow \mathbb{Z}_Q$ are defined. For simplicity these maps will be included to $ComInfo = \langle p, g, \mathbb{G}, \mathbb{G}_T, e; \psi_1, \psi_2 \rangle$.

Signer Parameters. The signer $\mathcal{S}$ uses the algorithm $\mathtt{GEN}$ to generate his public and secret parameters based on $ComInfo$. The signer selects $x, y \xleftarrow{\mathsf{r}} \mathbb{Z}_p^*$ and computes $X = g^x$ and $Y = g^y$. Then it is set that $PK_S = \langle X, Y \rangle$ and $SK_S = \langle x, y \rangle$ is the key pair for $\mathcal{S}$.

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. Nevertheless,

as part of our signing protocol he 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 settings; in fact these parameters can be part of a PKI that all users are members of. This will make the protocol time-complexity more efficient on the side of the user. We postpone further consideration of this issue for the full version of the paper.

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

Choice of parameter lengths. The length of parameters p, n, Q are ℓ_p, ℓ_n, ℓ_Q, respectively should be selected so that the following are satisfied: (i) The DLDH assumption holds over the bilinear group parameter, (ii) The LSRW assumption holds over the bilinear group parameter, (iii) The discrete-logarithm (DLOG) assumption holds over the group $(\mathbb{Z}_P^*)^2$, (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 $\ell_p = 171$ bits, $\ell_\mathsf{n} = 1024$ bits, $\ell_Q = 1024$ bits.

4.2 Signing Protocol

We give a high-level description before going to the details:

- First, both the user and the signer obtain the public inputs $ComInfo$, CRS, and PK_S, the signer gets the private input SK_S, and the user gets the private input message m.
- Then the user generates his key pair (PK_U, SK_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 Linear Encryption ciphertext for m which will be signed by the signer.
- To guarantee that the ciphertext and commitment are consistent, the user interleaves within the protocol execution a 3-move Σ-protocol with the signer that shows the consistency. This protocol employs an equivocal commitment scheme to allow for concurrent zero-knowledge argument (cf. [Dam00]).
- When the signer verifies the 3-move protocol successfully, he will transform the Linear Encryption ciphertext using his signing key SK_S and appropriately rerandomize it. This will result in the encryption of an essential component of a CL-signature.
- Finally, the user transforms the CL-signature from the signer into a blind signature for message m. This takes advantage of the homomorphic property of the CL-signature, in particular, the fact that the scheme is malleable and a signature holder can refresh the randomness of the signature.

We outline the high-level blind signature generation protocol in Figure 1. A detailed description is shown in Figure 2. Note that $d < p$, i.e. $\lambda_0 < \ell_p$. For example $\lambda_0 = 160$ bits, $\lambda_1 = 160$ bits.

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 4.4.

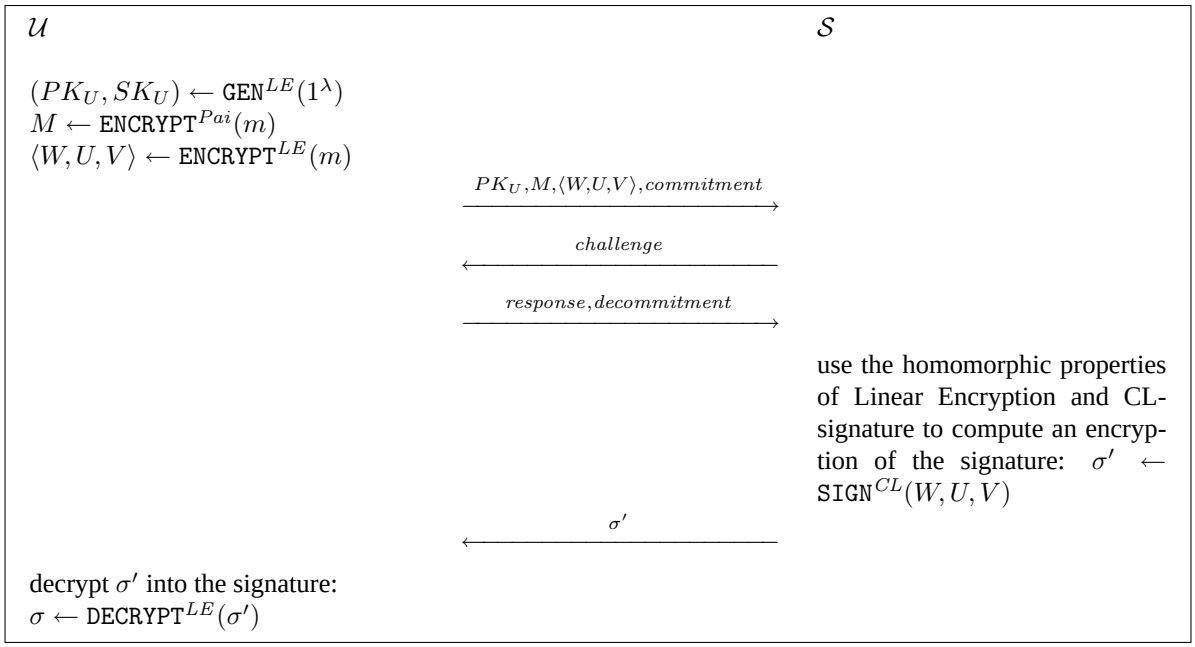

Figure 1: Overview of our blind signature generation protocol.

4.4.1 Correctness

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

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

$$\begin{aligned}
T_M &= \mathsf{g}^{k_0} k_1^{\mathsf{n}} \bmod \mathsf{n}^2 = \mathsf{g}^{s_0+d\cdot m}(s_1 \cdot l_1^d)^{\mathsf{n}} \bmod \mathsf{n}^2 \\
&= (\mathsf{g}^{s_0} s_1^{\mathsf{n}}) \cdot (\mathsf{g}^m s_1^{\mathsf{n}})^d \bmod \mathsf{n}^2 = \mathsf{g}^{s_0} s_1^{\mathsf{n}} M^d \bmod \mathsf{n}^2, \\
T_W &= \theta^{k_0} w^{k_2+k_3} = \theta^{s_0+d\cdot m} w^{(s_2+s_3)+d\cdot(l_2+l_3)} \\
&= (\theta^{s_0} w^{s_2+s_3}) \cdot (\theta^m w^{l_2+l_3})^d = \theta^{s_0} w^{s_2+s_3} W^d, \\
T_U &= u^{k_2} = u^{s_2+d\cdot l_2} = u^{s_2} \cdot (u^{l_2})^d = u^{s_2} U^d, \\
T_V &= v^{k_3} = v^{s_3+d\cdot l_3} = v^{s_3} \cdot (v^{l_2})^d = v^{s_3} V^d.
\end{aligned}$$

Then we check the correctness of the CL-signature.

$$\begin{aligned}
a &= (a')^t = \theta^{tt'}, \\
b &= (b')^t = (\theta^y)^{tt'} = (\theta^{tt'})^y = a^y, \\
c &= (W'/(U'^{\delta} V'^{\xi}))^t = ((W^{xy}\theta^x w^{l'_2+l'_3})/((U^{xy} u^{l'_2})^{\delta}(V^{xy} v^{l'_3})^{\xi}))^{tt'} \\
&= ((W/(U^{\delta} V^{\xi}))^{xy} \cdot \theta^x \cdot (w^{l'_2+l'_3}/(u^{\delta l'_2} v^{\xi l'_3})))^{tt'} \\
&= ((\theta^m)^{xy} \cdot \theta^x \cdot 1)^{tt'} = (\theta^{tt'})^{mxy+x} = a^{mxy+x}
\end{aligned}$$

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. We first build a useful lemma which guarantee that the user will use the same plaintext in the Linear Encryption and in the Paillier encryption based on the three-move proof. Consider a commitment scheme will fix the plaintexts. The lemma will still hold when we use the Pedersen multi-commitment to wrap up the three-move proof assuming the DLOG assumption.

$\boxed{\mathcal{U}}$		$\boxed{\mathcal{S}}$
$ComInfo = \langle p, g, \mathbb{G}, \mathbb{G}_T, e; \psi_1, \psi_2 \rangle$		$ComInfo = \langle p, g, \mathbb{G}, \mathbb{G}_T, e; \psi_1, \psi_2 \rangle$
$CRS = \langle \mathsf{n}, \mathsf{g}; h_M, h_W, h_U, h_V, h_r, P, Q \rangle$		$CRS = \langle \mathsf{n}, \mathsf{g}; h_M, h_W, h_U, h_V, h_r, P, Q \rangle$
$PK_S = \langle X, Y \rangle$		$PK_S = \langle X, Y \rangle$, $SK_S = \langle x, y \rangle$
$MSG = \langle m \rangle$, $m \in [0, 2^{\ell_p}]$		
$(PK_U, SK_U) \leftarrow \mathtt{GEN}^{LE}(1^\lambda)$		
$PK_U = \langle u, v, w \rangle$, $SK_U = \langle \delta, \xi \rangle$		
$k_0 \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$, $l_1, k_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$		
$t, l_2, l_3, k_2, k_3 \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}$, $r \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$		
$M = \mathsf{g}^m l_1^{\mathsf{n}} \bmod \mathsf{n}^2$		
$W = \theta^m w^{l_2+l_3}$, $U = u^{l_2}$, $V = v^{l_3}$		
$T_M = \mathsf{g}^{k_0} k_1^{\mathsf{n}} \bmod \mathsf{n}^2$		
$T_W = \theta^{k_0} w^{k_2+k_3}$, $T_U = u^{k_2}$, $T_V = v^{k_3}$		
$t_M = \psi_1(T_M)$, $t_W = \psi_2(T_W)$		
$t_U = \psi_2(T_U)$, $t_V = \psi_2(T_V)$		
$\mathsf{com} = h_M^{t_M} h_W^{t_W} h_U^{t_U} h_V^{t_V} h_r^r \bmod P$		
	$\xrightarrow{PK_U, M, \theta, \langle W, U, V \rangle, \mathsf{com}}$	
		$d \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_0}$
	$\xleftarrow{d}$	
$s_0 = k_0 - d \cdot m$(in $\mathbb{Z}$)		
$s_1 = k_1 \cdot l_1^{-d} \bmod \mathsf{n}$		
$s_2 = k_2 - d \cdot l_2 \bmod p$		
$s_3 = k_3 - d \cdot l_3 \bmod p$		
	$\xrightarrow{\langle s_0, s_1, s_2, s_3 \rangle, \langle T_M, T_W, T_U, T_V, r \rangle}$	
		$M \in^? \mathbb{Z}_{\mathsf{n}^2}^*$
		$s_0 \in^? \pm[0, 2^{\lambda_0+\lambda_1+\ell_p+1}]$
		$t_M = \psi_1(T_M)$, $t_W = \psi_2(T_W)$
		$t_U = \psi_2(T_U)$, $t_V = \psi_2(T_V)$
		$\mathsf{com} =^? h_M^{t_M} h_W^{t_W} h_U^{t_U} h_V^{t_V} h_r^r \bmod P$
		$T_M =^? \mathsf{g}^{s_0} s_1^{\mathsf{n}} M^d \bmod \mathsf{n}^2$
		$T_W =^? \theta^{s_0} w^{s_2+s_3} W^d$
		$T_U =^? u^{s_2} U^d$, $T_V =^? v^{s_3} V^d$
		$t', l_2', l_3' \xleftarrow{\mathtt{r}} \mathbb{Z}_p$
		$a' = (\theta)^{t'}$, $b' = (\theta^y)^{t'}$
		$W' = (W^{xy} \theta^x w^{l_2'+l_3'})^{t'}$
		$U' = (U^{xy} u^{l_2'})^{t'}$, $V' = (V^{xy} v^{l_3'})^{t'}$
	$\xleftarrow{a', b', \langle W', U', V' \rangle}$	
$a = (a')^t$, $b = (b')^t$, $c = (W'/(U'^{\delta} V'^{\xi}))^t$		
$\sigma = \langle a, b, c \rangle$		
$\mathtt{VERIFY}(m, \sigma) =^? 1$		
output $(m; \sigma)$		

Figure 2: Blind signature generation protocol.

Based on the lemma, we can simulate the signer sucessfully and reduce the unforgeability to the unforgeability of the CL-signature, which is based on the LRSW assumption. Therefore, our unforgeability is based on both the LRSW and the DLOG assumption.

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

$$\log_\theta \texttt{DECRYPT}^{LE}(W,U,V) \neq \texttt{DECRYPT}^{Pai}(M) \bmod p.$$

only with probability $2^{-\lambda_0}$.

Proof. Define $m = \texttt{DECRYPT}^{Pai}(M)$. Paillier encryption is 1-1 over $\mathbb{Z}^*_{\mathsf{n}^2}$, so it is well-defined and $m \in \mathbb{Z}_{\mathsf{n}}$. Also $M \in \mathbb{Z}^*_{\mathsf{n}^2}$ can be written as $M = g^m l_1^{\mathsf{n}} \bmod \mathsf{n}^2$ for some $l_1 \in \mathbb{Z}^*_{\mathsf{n}}$.

Similarly, define $m' = \log_\theta \texttt{DECRYPT}^{LE}(W,U,V)$. Recall $\theta \in \mathbb{G}$ and the order of $\mathbb{G}$ is prime p. So θ is a generator of $\mathbb{G}$, and we can get $\theta^{m'} = \texttt{DECRYPT}^{LE}(W,U,V)$ and $m' \in \mathbb{Z}_p$. Also $u, v \in \mathbb{G}$ are generators of $\mathbb{G}$, and $U, V \in \mathbb{G}$ can be written as $U = u^{l_2}$, $V = v^{l_3}$ for some $l_2, l_3 \in \mathbb{Z}_p$. Note that $\texttt{DECRYPT}^{LE}(W,U,V) = \frac{W}{U^\delta \cdot V^\xi}$. So $W = \theta^{m'} U^\delta V^\xi = \theta^{m'} u^{l_2\delta} v^{l_3\xi} = \theta^{m'} w^{l_2+l_3}$.

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:

$$\begin{aligned}
&m \neq m' \bmod p && m \in \mathbb{Z}_{\mathsf{n}}, m' \in \mathbb{Z}_p && (1)\\
&M = g^m l_1^{\mathsf{n}} \bmod \mathsf{n}^2 && l_1 \in \mathbb{Z}^*_{\mathsf{n}} && (2)\\
&W = \theta^{m'} w^{l_2+l_3} && l_2, l_3 \in \mathbb{Z}_p && (3)\\
&U = u^{l_2} && && (4)\\
&V = v^{l_3} && && (5)
\end{aligned}$$

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

$$\begin{aligned}
&T_M = g^{s_0} s_1^{\mathsf{n}} M^d \bmod \mathsf{n}^2 && (6)\\
&T_W = \theta^{s_0} w^{s_2+s_3} W^d && (7)\\
&T_U = u^{s_2} U^d && (8)\\
&T_V = v^{s_3} V^d && (9)
\end{aligned}$$

From equations (2) and (6) we have

$$T_M = g^{s_0} s_1^{\mathsf{n}} M^d \bmod \mathsf{n}^2 = g^{s_0} s_1^{\mathsf{n}} (g^m l_1^{\mathsf{n}})^d \bmod \mathsf{n}^2 = g^{s_0+dm} (s_1 l_1^d)^{\mathsf{n}} \bmod \mathsf{n}^2$$

By the similar way, we can get $T_U = u^{s_2+dl_2}$, $T_V = v^{s_3+dl_3}$, and $T_W = \theta^{s_0+dm'} w^{(s_2+dl_2)+(s_3+dl_3)}$. Now we call

$$\begin{aligned}
&k_0 \stackrel{\mathsf{def}}{=} s_0 + dm \bmod \mathsf{n} && (10)\\
&k_1 \stackrel{\mathsf{def}}{=} s_1 l_1^d \bmod \mathsf{n} && (11)\\
&k_2 \stackrel{\mathsf{def}}{=} s_2 + dl_2 \bmod p && (12)\\
&k_3 \stackrel{\mathsf{def}}{=} s_3 + dl_3 \bmod p && (13)\\
&k_0' \stackrel{\mathsf{def}}{=} s_0 + dm' \bmod p && (14)
\end{aligned}$$

Consider $\gcd(\mathsf{n}, p) = 1$. From the equations (10), we can let $k_0 = s_0 + dm + An$, where $A \in \mathbb{Z}$. So $k_0 - s_0 - dm = An$. Recall $s_0 \in \pm[0, 2^{\lambda_0+\lambda_1+\ell_p+1}]$, and $k_0 \in \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}$, $d \in \{0,1\}^{\lambda_0}$, and $m \in [0, 2^{\ell_p}]$. So, $k_0 - s_0 - dm \in \pm[0, 2^{\lambda_0+\lambda_1+\ell_p+2}]$, and $A = 0$ because $\ell_{\mathsf{n}} \gg \ell_p + \lambda_0 + \lambda_1 + 3$. So, $k_0 = s_0 + dm$.

From the equation (11), we can let $k_0' = s_0 + dm' + Bp$ where $B \in \mathbb{Z}$. So, $k_0 - k_0' = d(m - m') - Bp$. Recall $p \nmid (m - m')$. We can find such B only in the case of $p \mid (k_0 - k_0') - d(m - m')$, which is with negligible probability $2^{-\lambda_0}$. In other words, the adversary can only find satisfied k_0, k_0' to develop the proof with probability $2^{-\lambda_0}$.

Based on the argument above, we know that, except negligible probability $2^{-\lambda_0}$, the adversary cannot develop a valid proof with $m \neq m' \bmod p$.

We can prove the lemma with a general commitment scheme because a commitment scheme will fix the parameters $k_0, k_1, k_2, k_3, k_0', m, m', l_1, l_2, l_3$ in equations (10-14).We can similarly argue that the adversary has only negligible probability $2^{-\lambda_0}$ to develop a valid proof. Now the lemma will hold based on the assumption that the commitment uses. □

Theorem 4.3 (Unforgeability). *The proposed scheme is $(l, l+1)$-unforgeable if both the LRSW and the DLOG assumptions hold.*

Proof. In this part, we will show under LRSW assumption, no PPT adversary user $\mathcal{A}$ can achieve "one-more" forgery. 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 $l+1$ valid message-signature pairs after l times successful executions with the signer, we can construct a simulator 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 simulator defines two 1-1 maps ψ_1, ψ_2 as in the key-generation algorithm of the proposed scheme, and sets $ComInfo = \langle p, g, \mathbb{G}, \mathbb{G}_T, e, \psi_1, \psi_2 \rangle$. The simulator sets $PK_S = \langle X, Y \rangle$. The simulator generates $CRS_2 = \langle h_M, h_W, h_U, h_V, h_r, P, Q \rangle$ as in the key-generation algorithm, and discards the corresponding $trapdoor_2 = \langle \tau_M, \tau_W, \tau_U, \tau_V \rangle$; the simulator generates $CRS_1 = \langle \mathsf{n}, \mathsf{g} \rangle$ as in the key-generation algorithm, and keeps the corresponding $trapdoor_1 = \langle \mathsf{p}, \mathsf{q} \rangle$; the simulator sets $CRS = (CRS_1, CRS_2)$. The simulator supplies the adversary with $ComInfo, CRS, PK_S$.

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 $\langle \texttt{start}, msg \rangle$, where $msg = \{PK_U, M, \theta, \langle W, U, V \rangle, \mathsf{com}\}$. 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 $\mathcal{S}$ with msg till $\mathcal{S}$ either terminates or returns a response rsp to the user; the oracle $\mathcal{I}$ records the current state in st. If $\mathcal{S}$ returns rsp then $\mathcal{I}$ returns this with the session identity to $\mathcal{A}$, i.e. $\mathcal{I}$ return $\{sid, d\}$ to $\mathcal{A}$.

Case 2: $\mathcal{A}$ queries $\mathcal{I}$ with $\langle \texttt{advance}, sid, msg \rangle$, where $msg = \{\langle s_0, s_1, s_2, s_3 \rangle, \langle T_M, T_W, T_U, T_V, r \rangle\}$. The oracle $\mathcal{I}$ will simulate the signer $\mathcal{S}$ with msg and previous state st. The $\mathcal{S}$ checks if all equations hold: $\mathsf{com} = h_M^{t_M} h_W^{t_W} h_U^{t_U} h_V^{t_V} h_r^r \mod P$, $T_M = \mathsf{g}^{s_0} s_1^{\mathsf{n}} M^d \bmod \mathsf{n}^2$, $T_W = \theta^{s_0} w^{s_2+s_3} W^d$, $T_U = u^{s_2} U^d$, $T_V = v^{s_3} V^d$, where $t_M = \psi_1(T_M)$, $t_W = \psi_2(T_W)$, $t_U = \psi_2(T_U)$, $t_V = \psi_2(T_V)$. If not true, terminates. Otherwise, because we use Pedersen [Ped91] multi-commitment which is based on the DLOG assumption, from the lemma above, we can obtain the m under $\{\theta, W, U, V\}$ by decrypting m from M, and the oracle $\mathcal{I}$ can generate an identically distributed response $\{a', b', W', U', V'\}$ to $\mathcal{A}$ by simulating O_{XY} with m: $\mathcal{S}$ uses the trapdoor information $trapdoor_1 = \langle \mathsf{p}, \mathsf{q} \rangle$ to decrypt M into $m = \texttt{DECRYPT}^{Pai}_{trapdoor_1}(M)$, and returns m to the oracle $\mathcal{I}$. The oracle $\mathcal{I}$ simulates $O_{X,Y}$ with input $m \bmod p$ which returns $\langle a, b, c \rangle$. The oracle $\mathcal{I}$ computes $a' = a$, $b' = b$, $W' = c w^{l_2''+l_3''}$, $U' = u^{l_2''}$, $V' = v^{l_3''}$, where $l_2'', l_3'' \xleftarrow{\mathrm{r}} \mathbb{Z}_p$, and sends them to $\mathcal{A}$. Here $\langle W', U', V' \rangle$ is in fact the ciphertext of c over $\mathcal{A}$'s public key $\langle u, v, w \rangle$. We claim $\{a', b', W', U', V'\}$ is identically distributed to the protocol answer, i.e. $\{a, b, c w^{l_2''+l_3''}, u^{l_2''}, v^{l_3''}\} \approx \{(\theta)^{t'}, (\theta^y)^{t'}, (W^{xy} \theta^x w^{l_2'+l_3'})^{t'}, (U^{xy} u^{l_2'})^{t'}, (V^{xy} v^{l_3'})^{t'}\}$. Note that $\langle a, b, c \rangle$ is the response from $O_{X,Y}$. So, a is a random element in $\mathbb{G}$, $b = a^y$, $c = a^{x+mxy}$. Based on the lemma, and equation (1), we know $W = \theta^m w^{l_2+l_3}$, $U = u^{l_2}$, $V = v^{l_3}$. We can compute $(W^{xy} \theta^x w^{l_2'+l_3'})^{t'} = ((\theta^m w^{l_2+l_3})^{xy} \theta^x w^{l_2'+l_3'})^{t'} = ((\theta)^{t'})^{x+mxy} w^{(l_2xy+l_2')t'+(l_3xy+l_3')t'}$, $(U^{xy} u^{l_2'})^{t'} = ((u^{l_2})^{xy} u^{l_2'})^{t'} = u^{(l_2xy+l_2')t'}$, $(V^{xy} v^{l_3'})^{t'} = ((v^{l_3})^{xy} v^{l_3'})^{t'} = v^{(l_3xy+l_3')t'}$. Recall t', l_2', l_3' are randomly selected. So we can replace $\theta^{t'}, (l_2xy + l_2')t', (l_3xy + l_3')t'$ with a, l_2'', l_3'', which means that the two probability distributions are identical.

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 $(m_1^*;\sigma_1^*),(m_2^*;\sigma_2^*),\ldots,(m_{l'}^*;\sigma_{l'}^*)$ with $m_i \neq m_j$ and $l' > l$. Since $l'-l \geq 1$, 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. Start from the blindness model, we define 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 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 3; then do the similar simulation for the right user instantiation in Game 4. Both distances between Game 2 and Game 3, and Game 3 and Game 4 are $\mathsf{Adv}_{\mathsf{DCR}}$ which is negligible under the DCR assumption.

Similarly, we slightly change Game 4 into Game 5 by simulating the left user instantiation with inputting a random message to the linear encryption; then change Game 5 into Game 6 by similar way for the right instantiation. Again the distances between Game 4 and Game 5, and Game 5 and Game 6 are $\mathsf{Adv}_{\mathsf{DLDH}}$ which is negligible under the DLDH assumption.

Theorem 4.4 (Blindness). *The proposed scheme is blind if both the DLDH assumption and the DCR assumption hold.*

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 $\mathcal{U}^L$ and the right one $\mathcal{U}^R$, where $j = 0,1,\ldots,6$. Also we define S_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. $(ComInfo, CRS, PK_S, SK_S) \leftarrow \mathtt{GEN}(1^\lambda)$; set $PubInfo = (ComInfo, CRS, PK_S)$
3. $\phi' \leftarrow \mathcal{A}^{\mathcal{I}_0^{\phi}}(1^\lambda, PubInfo)$;
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 $\mathcal{U}^L$ (resp. $\mathcal{U}^R$) 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 $\mathcal{U}^L$ (resp. $\mathcal{U}^R$).

- 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 $\mathcal{U}^\rho$ till $\mathcal{U}^\rho$ either terminates or returns a response to the signer. If $\mathcal{U}^\rho$ 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 $\mathcal{U}^\rho$, i.e. $m = m_\phi$ for $\rho = L$ and $m = m_{1-\phi}$ for $\rho = R$, we have,

(a) $(PK_U^\rho, SK_U^\rho) \leftarrow \mathtt{GEN}^{LE}(1^\lambda)$

(b) $k_0 \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$, $l_1, k_1 \xleftarrow{\mathtt{r}} \mathbb{Z}_{\mathsf{n}}^*$, $t, l_2, l_3, k_2, k_3 \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $r \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}$.

(c) $M \leftarrow \mathtt{ENCRYPT}_{CRS_1}^{Pai}(m, l_1)$

(d) $\langle W, U, V\rangle \leftarrow \mathtt{ENCRYPT}_{ComInfo, PK_U^\rho}^{LE}(m, \theta, l_2, l_3)$

(e) $T_M \leftarrow \mathtt{ENCRYPT}_{CRS_1}^{Pai}(k_0, k_1)$

(f) $\langle T_W, T_U, T_V\rangle \leftarrow \mathtt{ENCRYPT}_{ComInfo, PK_U^\rho}^{LE}(k_0, \theta, k_2, k_3)$

(g) $\mathsf{com} = h_M^{\psi_1(T_M)} h_W^{\psi_2(T_W)} h_U^{\psi_2(T_U)} h_V^{\psi_2(T_V)} h_r^r \mod P$

(h) $rsp = \{PK_U, M, \theta, \langle W, U, V\rangle, \mathsf{com}\}$

– If $msg = \{d\}$, then $\mathcal{I}_0^\phi$ recovers the state of st^ρ, and simulates the user instantiation $\mathcal{U}^\rho$ with msg till $\mathcal{U}^\rho$ either terminates or returns a response rsp to the signer. If $\mathcal{U}^\rho$ 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 $\{\langle s_0, s_1, s_2, s_3\rangle, \langle T_M, T_W, T_U, T_V, r\rangle\}$, where $\langle T_M, T_W, T_U, T_V, r\rangle$ is recovered from the previous state of st^ρ, and $\langle s_0, s_1, s_2, s_3\rangle$ is generated as: $s_0 = k_0 - d\cdot m \in \mathbb{Z}$, $s_1 = k_1 \cdot l_1^{-d} \bmod \mathsf{n}$, $s_2 = k_2 - d \cdot l_2 \bmod p$, $s_3 = k_3 - d \cdot l_3 \bmod p$.

- 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 $\mathcal{U}^L$ (resp. $\mathcal{U}^R$) with msg^L (resp. msg^R) till $\mathcal{U}^L$ (resp. $\mathcal{U}^R$) either terminates or returns an output. If both user instantiations return outputs, and the outputs are valid blind signatures for m_0, m_1, then let $rsp = (\sigma_0, \sigma_1)$ be the valid signatures. Otherwise let rsp be $(\perp, \perp)$. The oracle returns rsp to $\mathcal{A}$.

Here msg^ρ is in form of $\{a', b', \langle W', U', V'\rangle\}$, and σ_i is in form of (a, b, c) which are generated as: $a = (a')^t$, $b = (b')^t$, $c = (W'/(U'^\delta V'^\xi))^t$.

Game 1:

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

2. $(ComInfo, CRS_1, PK_S, SK_S) \leftarrow \mathtt{GEN}(1^\lambda)$; $CRS_2 = \langle h_M, h_W, h_U, h_V, h_r, P, Q\rangle$ generated as: $h_r \xleftarrow{\mathtt{r}} \mathbb{Z}_P$, $\tau_M, \tau_W, \tau_U, \tau_V \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$, $h_M = h_r^{\tau_M} \bmod P$, $h_W = h_r^{\tau_W} \bmod P$, $h_U = h_r^{\tau_U} \bmod P$, $h_V = h_r^{\tau_V} \bmod P$. Keep $trapdoor_2 = \langle \tau_M, \tau_W, \tau_U, \tau_V\rangle$ secretly. Let $CRS = (CRS_1, CRS_2)$, and set $PubInfo = (ComInfo, CRS, PK_S)$.

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 $\mathcal{U}^L$ till $\mathcal{U}^L$ either terminates or returns a response to the signer. If $\mathcal{U}^L$ 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) $(PK_U^L, SK_U^L) \leftarrow \mathtt{GEN}^{LE}(1^\lambda)$

(b) $l_1 \xleftarrow{r} \mathbb{Z}^*_{\mathsf{n}}, t, l_2, l_3 \xleftarrow{r} \mathbb{Z}_p, \tau \xleftarrow{r} \mathbb{Z}_Q, \theta \xleftarrow{r} \mathbb{G}$.

(c) $M \leftarrow \mathtt{ENCRYPT}^{Pai}_{CRS_1}(m, l_1)$

(d) $\langle W, U, V\rangle \leftarrow \mathtt{ENCRYPT}^{LE}_{ComInfo, PK^L_U}(m, \theta, l_2, l_3)$

(e) $\mathsf{com} = h_r^{\tau}$

(f) $rsp = \{PK^L_U, M, \theta, \langle W, U, V\rangle, \mathsf{com}\}$

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

(a) $s_0 \xleftarrow{r} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$, $s_1 \xleftarrow{r} \mathbb{Z}^*_{\mathsf{n}}$, $s_2, s_3 \xleftarrow{r} \mathbb{Z}_p$

(b) $T_M = \mathsf{g}^{s_0} s_1^{\mathsf{n}} M^d \mod \mathsf{n}^2$

(c) $T_W = \theta^{s_0} w^{s_2+s_3} W^d$, $T_U = u^{s_2} U^d$, $T_V = v^{s_3} V^d$

(d) $r = \tau - (\tau_M \cdot \psi_1(T_M) + \tau_W \cdot \psi_2(T_W) + \tau_U \cdot \psi_2(T_U) + \tau_V \cdot \psi_2(T_V)) \bmod Q$

(e) $rsp = \{\langle s_0, s_1, s_2, s_3\rangle, \langle T_M, T_W, T_U, T_V, r\rangle\}$

Game 2:

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

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

Game 3:

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

- Given $\langle \mathtt{challenge}, m_0, m_1\rangle$, the oracle $\mathcal{I}^{\phi}_3$ randomly selects $\widetilde{m}_0$, $\widetilde{m}_1$ from the message space and simulates $\mathcal{U}^L$ (resp. $\mathcal{U}^R$) 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 $\rho = R$, $\mathcal{I}^{\phi}_3$ operates identically as $\mathcal{I}^{\phi}_2$; but if $\rho = L$, $\mathcal{I}^{\phi}_3$ works as follows:

  – If $msg = \perp$, then $\mathcal{I}^{\phi}_3$ recovers the state of st^L, and simulates the user instantiation $\mathcal{U}^L$ till $\mathcal{U}^L$ either terminates or returns a response to the signer. If $\mathcal{U}^L$ returns a response rsp, then $\mathcal{I}^{\phi}_3$ 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) $(PK^L_U, SK^L_U) \leftarrow \mathtt{GEN}^{LE}(1^{\lambda})$

  (b) $l_1 \xleftarrow{r} \mathbb{Z}^*_{\mathsf{n}}, t, l_2, l_3 \xleftarrow{r} \mathbb{Z}_p, \tau \xleftarrow{r} \mathbb{Z}_Q, \theta \xleftarrow{r} \mathbb{G}$.

  (c) $\widetilde{M} \leftarrow \mathtt{ENCRYPT}^{Pai}_{CRS_1}(\widetilde{m}, l_1)$

  (d) $\langle W, U, V\rangle \leftarrow \mathtt{ENCRYPT}^{LE}_{ComInfo, PK^L_U}(m, \theta, l_2, l_3)$

  (e) $\mathsf{com} = h_r^{\tau}$

  (f) $rsp = \{PK^L_U, \widetilde{M}, \theta, \langle W, U, V\rangle, \mathsf{com}\}$

  – If $msg = \{d\}$, then $\mathcal{I}^{\phi}_3$ recovers the state of st^L, and simulates the user instantiation $\mathcal{U}^L$ with msg till $\mathcal{U}^L$ either terminates or returns a response rsp to the signer. If $\mathcal{U}^L$ returns a response rsp, then $\mathcal{I}^{\phi}_3$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$.

(a) $s_0 \xleftarrow{r} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$, $s_1 \xleftarrow{r} \mathbb{Z}_{\mathsf{n}}^*$, $s_2, s_3 \xleftarrow{r} \mathbb{Z}_p$

(b) $T_{\widetilde{M}} = \mathsf{g}^{s_0} s_1^{\mathsf{n}} \widetilde{M}^d \mod \mathsf{n}^2$

(c) $T_W = \theta^{s_0} w^{s_2+s_3} W^d$, $T_U = u^{s_2} U^d$, $T_V = v^{s_3} V^d$

(d) $r = \tau - (\tau_M \cdot \psi_1(T_{\widetilde{M}}) + \tau_W \cdot \psi_2(T_W) + \tau_U \cdot \psi_2(T_U) + \tau_V \cdot \psi_2(T_V)) \bmod Q$

(e) $rsp = \{\langle s_0, s_1, s_2, s_3\rangle, \langle T_{\widetilde{M}}, T_W, T_U, T_V, r\rangle\}$

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

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 $\rho = R$, $\mathcal{I}_5^{\phi}$ operates identically as $\mathcal{I}_4^{\phi}$; but if $\rho = L$, $\mathcal{I}_5^{\phi}$ works as follows:
  - If $msg = \perp$, then $\mathcal{I}_5^{\phi}$ recovers the state of st^{ρ}, and simulates the user instantiation $\mathcal{U}^L$ till $\mathcal{U}^L$ either terminates or returns a response to the signer. If $\mathcal{U}^L$ returns a response rsp, then $\mathcal{I}_5^{\phi}$ 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) $(PK_U^L, SK_U^L) \leftarrow \mathtt{GEN}^{LE}(1^{\lambda})$
    (b) $l_1 \xleftarrow{r} \mathbb{Z}_{\mathsf{n}}^*$, $t, l_2, l_3 \xleftarrow{r} \mathbb{Z}_p$, $\tau \xleftarrow{r} \mathbb{Z}_Q$, $\theta \xleftarrow{r} \mathbb{G}$.
    (c) $\widetilde{M} \leftarrow \mathtt{ENCRYPT}_{CRS_1}^{Pai}(\widetilde{m}, l_1)$
    (d) $\langle \widetilde{W}, \widetilde{U}, \widetilde{V}\rangle \leftarrow \mathtt{ENCRYPT}_{ComInfo, PK_U^L}^{LE}(\widetilde{m}, \theta, l_2, l_3)$
    (e) $\mathsf{com} = h_r^{\tau}$
    (f) $rsp = \{PK_U^L, \widetilde{M}, \theta, \langle \widetilde{W}, \widetilde{U}, \widetilde{V}\rangle, \mathsf{com}\}$
  - If $msg = \{d\}$, then $\mathcal{I}_5^{\phi}$ recovers the state of st^L, and simulates the user instantiation $\mathcal{U}^L$ with msg till $\mathcal{U}^L$ either terminates or returns a response rsp to the signer. If $\mathcal{U}^L$ returns a response rsp, then $\mathcal{I}_5^{\phi}$ returns rsp to $\mathcal{A}$. The oracle will record the current state st, i.e. $st^L = st^L||st$.
    (a) $s_0 \xleftarrow{r} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$, $s_1 \xleftarrow{r} \mathbb{Z}_{\mathsf{n}}^*$, $s_2, s_3 \xleftarrow{r} \mathbb{Z}_p$
    (b) $T_{\widetilde{M}} = \mathsf{g}^{s_0} s_1^{\mathsf{n}} \widetilde{M}^d \mod \mathsf{n}^2$
    (c) $T_{\widetilde{W}} = \theta^{s_0} w^{s_2+s_3} \widetilde{W}^d$, $T_{\widetilde{U}} = u^{s_2} \widetilde{U}^d$, $T_{\widetilde{V}} = v^{s_3} \widetilde{V}^d$
    (d) $r = \tau - (\tau_M \cdot \psi_1(T_{\widetilde{M}}) + \tau_W \cdot \psi_2(T_{\widetilde{W}}) + \tau_U \cdot \psi_2(T_{\widetilde{U}}) + \tau_V \cdot \psi_2(T_{\widetilde{V}})) \bmod Q$
    (e) $rsp = \{\langle s_0, s_1, s_2, s_3\rangle, \langle T_{\widetilde{M}}, T_{\widetilde{W}}, T_{\widetilde{U}}, T_{\widetilde{V}}, r\rangle\}$

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 $\rho = L$, $\mathcal{I}_6^{\phi}$ operates identically as $\mathcal{I}_5^{\phi}$; but if $\rho = R$, $\mathcal{I}_6^{\phi}$ operates similarly as the case $\rho = L$ with $\widetilde{m} = \widetilde{m}_1$, i.e. runs the same operations for the right user instantiation $\mathcal{U}^R$.

Compute the statistical distance:

We prove in Game 0 and Game 1, under the DLOG assumption, $|\Pr[S_0] - \Pr[S_1]|$ is negligible. [Note: the DLDH assumption is stronger than the DLOG assumption, i.e. if DLDH assumption holds, so does DLOG assumption.] Observer that, for the probability distributions of the right user instantiations $[\mathcal{U}^R]_2$, $[\mathcal{U}^R]_3$ are identical. We still need to show for the left user instantiations $[\mathcal{U}^L]_2$, $[\mathcal{U}^L]_3$, under the DLOG assumption, the statistical distance of the probability distributions is negligible. First, we prove the statistical distance of $[s_0]_0$ and $[s_0]_1$ are negligible. Observe that in both games $m \in [0, 2^{\ell_p}]$, $k_0 \in \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$, $d \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_0}$. We can obtain that the statistical distance of the random variables $[s_0]_0 = k_0 - d \cdot m$ and $[s_0]_1 \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$ is less than $2^{-\lambda_1-1}$. Then we can observe that $[s_1]_0$ and $[s_1]_1$, $[s_2]_0$ and $[s_2]_1$, $[s_3]_0$ and $[s_3]_1$ are identically distributed. So the statistical distance of $[s_0, s_1, s_2, s_3]_0$ and $[s_0, s_1, s_2, s_3]_1$ is $2^{-\lambda_1-1}$. Note that by Damgård's trick [Dam00], we use a Pedersen multi-commitment scheme (under DLOG assumption) to transform a three-move HVSZK protocol into a SZK protocol. So under the DLOG assumption, the statistical distance of the the two games is $2^{-\lambda_1-1}$, i.e. $|\Pr[S_0] - \Pr[S_1]| \leq 2^{-\lambda_1-1}$

Use the similar argument, we can show in Game 1 and Game 2, $|\Pr[S_1] - \Pr[S_2]| \leq 2^{-\lambda_1-1}$

Now we prove in Game 2 and Game 3, under the DCR assumption, $|\Pr[S_2] - \Pr[S_3]|$ is negligible. Observer that, for the probability distributions of the right user instantiations $[\mathcal{U}^R]_2$, $[\mathcal{U}^R]_3$ are identical, and for the left user instantiations $[\mathcal{U}^L]_2$, $[\mathcal{U}^L]_3$, under he DCR assumption, the triples $[M]_3$, $[\widetilde{M}]_4$ are indistinguishable, which also leads that $[T_M]_2$ and $[T_{\widetilde{M}}]_3$, $[r]_2$ and $[r]_3$, are indistinguishable. So $|\Pr[S_2] - \Pr[S_3]| \leq \mathsf{Adv}_{\mathsf{DCR}}$.

We can prove in Game 3 and Game 4, under the DCR assumption, $|\Pr[S_3] - \Pr[S_4]|$ is negligible by the similar argument as above: the probability distributions of the left user instantiations $[\mathcal{U}^L]_3$, $[\mathcal{U}^L]_4$ are identical, and the statistical distance of the probability distributions of the right user instantiations $[\mathcal{U}^R]_3$, $[\mathcal{U}^R]_4$ are indistinguishable, which results in $|\Pr[S_3] - \Pr[S_4]| \leq \mathsf{Adv}_{\mathsf{DCR}}$.

Next we prove in Game 4 and Game 5, under the DLDH assumption, $|\Pr[S_4] - \Pr[S_5]|$ is negligible. Observer that, for the probability distributions of the right user instantiations $[\mathcal{U}^R]_4$, $[\mathcal{U}^R]_5$ are identical, and for the left user instantiations $[\mathcal{U}^L]_4$, $[\mathcal{U}^L]_5$, under he DLDH assumption, the triples $[W, U, V]_4$, $[\widetilde{W}, \widetilde{U}, \widetilde{V}]_5$ are indistinguishable, which also leads that $[T_W, T_U, T_V]_4$ and $[T_{\widetilde{W}}, T_{\widetilde{U}}, T_{\widetilde{V}}]_5$, $[r]_4$ and $[r]_5$, are indistinguishable. So $|\Pr[S_4] - \Pr[S_5]| \leq \mathsf{Adv}_{\mathsf{DLDH}}$.

We can prove in Game 5 and Game 6, under the DLDH assumption, $|\Pr[S_5] - \Pr[S_6]|$ is negligible by the similar argument as above: the probability distributions of the left user instantiations $[\mathcal{U}^L]_5$, $[\mathcal{U}^L]_6$ are identical, and the statistical distance of the probability distributions of the right user instantiations $[\mathcal{U}^R]_5$, $[\mathcal{U}^R]_6$ are indistinguishable, which results in $|\Pr[S_5] - \Pr[S_6]| \leq \mathsf{Adv}_{\mathsf{DLDH}}$.

In Game 6, ϕ is not used, so the adversary $\mathcal{A}$ has only probability $\frac{1}{2}$ to win the game, i.e. $\Pr[S_6] = \frac{1}{2}$.

Based on the argument above, we can get

$$\begin{aligned}\left|\Pr[S_0] - \tfrac{1}{2}\right| &= |\Pr[S_0] - \Pr[S_6]| = |\sum_{j=0}^{5} \Pr[S_j] - \Pr[S_{j+1}]| \leq \sum_{j=0}^{5} |\Pr[S_j] - \Pr[S_{j+1}]| \\ &= 2^{-\lambda_1-1} + 2^{-\lambda_1-1} + \mathsf{Adv}_{\mathsf{DCR}} + \mathsf{Adv}_{\mathsf{DCR}} + \mathsf{Adv}_{\mathsf{DLDH}} + \mathsf{Adv}_{\mathsf{DLDH}} \\ &= 2^{-\lambda_1} + 2\mathsf{Adv}_{\mathsf{DCR}} + 2\mathsf{Adv}_{\mathsf{DLDH}}\end{aligned}$$

Under both the DLDH assumption and the DCR assumption, $|\Pr[S_0] - \frac{1}{2}|$ is negligible. This completes the proof of blindness.

□

5 Extensions and Variants

5.1 Stronger Blindness

We present a variant of our scheme where we prove the blindness property only conditional on the DLDH assumption without relying on the DCR assumption. We will still employ the DCR assumption but this will be transferred to the unforgeability property. This modification strengthens the blindness property in the following sense: in our basic construction blindness relies on the security of a long-lived parameter of the system (the factorization of the modulus n) as well as on DLDH. On the other hand, in the modified scheme that we present on this paragraph, the blindness property relies only on the DLDH assumption which refers to the short-lived keys for Linear Encryption that are generated by the user himself.

In the modified scheme, we replace the Paillier encryption $M = \mathsf{g}^m l_1^{\mathsf{n}} \bmod \mathsf{n}^2$, where $\mathsf{g} = (1+\mathsf{n})$ with a commitment $M = \mathsf{g}_0^m \mathsf{h}_0^{l_1} \bmod \mathsf{n}^2$, where $\mathsf{g}_0, \mathsf{h}_0 \xleftarrow{\mathsf{r}} \mathbb{Z}^*_{\mathsf{n}^2}$. Note that n is same as that in our basic scheme, and $\mathsf{n}, \mathsf{g}_0, \mathsf{h}_0$ are also included into the CRS.

Observe that this modification transforms the computationally hiding commitment M into a perfectly hiding commitment: the CRS contains the values $\mathsf{g}_0 = (1+\mathsf{n})^{\alpha_1}\beta_1^{\mathsf{n}} \bmod \mathsf{n}^2$ and $\mathsf{h}_0 = (1+\mathsf{n})^{\alpha_2}\beta_2^{\mathsf{n}} \bmod \mathsf{n}^2$ where $\alpha_1, \alpha_2 \xleftarrow{\mathsf{r}} \mathbb{Z}_{\mathsf{n}}$ and $\beta_1, \beta_2 \xleftarrow{\mathsf{r}} \mathbb{Z}^*_{\mathsf{n}}$. As a result $M = \mathsf{g}_0^m \mathsf{h}_0^{l_1} = (1+\mathsf{n})^{\alpha_1 m + \alpha_2 l_1}(\beta_1^m \beta_2^{l_1})^{\mathsf{n}} \bmod \mathsf{n}^2$. It follows that, if l_1 is randomly selected from $[1..\lfloor \frac{n^2}{4} \rfloor]$ the commitment C does reveal any information about m in the information-theoretic sense.

When we prove the unforgeability, we can modify the CRS with $\mathsf{g}_0 = (1+\mathsf{n})\beta_1^{\mathsf{n}} \bmod \mathsf{n}^2$ and $\mathsf{h}_0 = \beta_2^{\mathsf{n}} \bmod \mathsf{n}^2$ where $\beta_1, \beta_2 \xleftarrow{\mathsf{r}} \mathbb{Z}^*_{\mathsf{n}}$. Now $M = \mathsf{g}_0^m \mathsf{h}_0^{l_1} = ((1+\mathsf{n})\beta_1^{\mathsf{n}})^m (\beta_2^{\mathsf{n}})^{l_1} = (1+\mathsf{n})^m (\beta_1^m \beta_2^{l_1})^{\mathsf{n}} \bmod \mathsf{n}^2$ which is a Paillier ciphertext over $\mathbb{Z}^*_{\mathsf{n}^2}$. So the simulator can use the corresponding trapdoor "open" M into m which leads to a successful simulation.

5.2 Revokable Blindness

In this case we modify our scheme in the opposite direction: we introduce a trusted third party $\mathcal{T}$ with the key pair (PK_T, SK_T) that is capable of receiving a transcript of the signing protocol and recovering the message that was submitted for signing by the user, i.e., revoke the user's blindness from a signing protocol transcript.

In the new scheme, besides the actions taken by the user in the signing protocol, when the user sends out the Paillier ciphertext M of m in parallel he sends the signer a ciphertext $\bar{M}$ that encrypts the message m under the public-key PK_T of the trusted third party; he couples this with a proof of equality of plaintexts for the two ciphertexts M and $\bar{M}$ that is AND-composed to the other proofs that the adversary performs in the protocol.

When the trusted third party wants to revoke the blindness from a blind signing protocol transcript, he just needs to "open" $\bar{M}$ into m by his secret key SK_T.

In Figure 3, we give a detail description of such blind signature generation. In the key generation algorithm, $ComInfo, PK_S, SK_S$ are same as that in the basic scheme; we slightly change the Pedersen commitment scheme for 4 components into that for 5 components and change the CRS into $CRS = \langle \mathsf{n}, \mathsf{g}; h_M, h_{\bar{M}}, h_W, h_U, h_V, h_r, P, Q \rangle$. The third party $\mathcal{T}$ is associated with another Paillier encryption with $PK_T = \langle \bar{\mathsf{n}}, \bar{\mathsf{g}} \rangle$, and $SK_T = \langle \bar{\mathsf{p}}, \bar{\mathsf{q}} \rangle$, where $\bar{\mathsf{p}}$ and $\bar{\mathsf{q}}$ are random primes and $\bar{\mathsf{n}} = \bar{\mathsf{p}}\bar{\mathsf{q}}$ such that $\bar{\mathsf{p}}, \bar{\mathsf{q}} > 2$, $\bar{\mathsf{p}} \neq \bar{\mathsf{q}}$, $|\bar{\mathsf{p}}| = |\bar{\mathsf{q}}|$, $\gcd(\bar{\mathsf{p}}\bar{\mathsf{q}}, (\bar{\mathsf{p}}-1)(\bar{\mathsf{q}}-1)) = 1$, and $|\bar{\mathsf{n}}| = |\mathsf{n}|$. The secret key $SK_T = \langle \bar{\mathsf{p}}, \bar{\mathsf{q}} \rangle$ which is only known by $\mathcal{T}$, and the public key $PK_T = \langle \bar{\mathsf{n}}, \bar{\mathsf{g}} \rangle$ where $\bar{\mathsf{g}} = (1+\bar{\mathsf{n}})$.

The proposed blind signature with revocable blindness is based on the CL-signature, which is generated as: select a random $a \in \mathbb{G}$ and output the signature $\sigma = \langle a, a^y, a^{x+xym} \rangle$. And when we obtain (m, σ) which is generated by the signing protocol of the proposed scheme, we can verify it as: $e(a, Y) = e(g, b)$; $e(X, a)e(X, b)^m = e(g, c)$.

$\boxed{\mathcal{U}}$		$\boxed{\mathcal{S}}$
$ComInfo, CRS, PK_S, PK_T$ MSG		$ComInfo, CRS, PK_S, PK_T$ SK_S
$(PK_U, SK_U) \leftarrow \mathtt{GEN}^{LE}(1^\lambda)$ $PK_U = \langle u, v, w\rangle$, $SK_U = \langle \delta, \xi\rangle$ $k_0, \bar{k}_0 \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$ $l_1, k_1 \xleftarrow{\mathtt{r}} \mathbb{Z}^*_{\mathsf{n}}$, $\bar{l}_1, \bar{k}_1 \xleftarrow{\mathtt{r}} \mathbb{Z}^*_{\bar{\mathsf{n}}}$ $t, l_2, l_3, k_2, k_3, \bar{l}_2, \bar{l}_3, \bar{k}_2, \bar{k}_3 \xleftarrow{\mathtt{r}} \mathbb{Z}_p$ $\theta \xleftarrow{\mathtt{r}} \mathbb{G}$, $r \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$ $M = \mathsf{g}^m l_1^{\mathsf{n}} \bmod \mathsf{n}^2$ $\bar{M} = \bar{\mathsf{g}}^m l_1^{\bar{\mathsf{n}}} \bmod \bar{\mathsf{n}}^2$ $W = \theta^m w^{l_2+l_3}$, $U = u^{l_2}$, $V = v^{l_3}$ $T_M = \mathsf{g}^{k_0} k_1^{\mathsf{n}} \bmod \mathsf{n}^2$ $T_{\bar{M}} = \bar{\mathsf{g}}^{k_0} k_1^{\bar{\mathsf{n}}} \bmod \bar{\mathsf{n}}^2$ $T_W = \theta^{k_0} w^{k_2+k_3}$, $T_U = u^{k_2}$, $T_V = v^{k_3}$ $t_M = \psi_1(T_M)$, $t_{\bar{M}} = \psi_1(T_{\bar{M}})$ $t_W = \psi_2(T_W)$, $t_U = \psi_2(T_U)$ $t_V = \psi_2(T_V)$		
$\mathsf{com} = h_M^{t_M} h_{\bar{M}}^{t_{\bar{M}}} h_W^{t_W} h_U^{t_U} h_V^{t_V} h_r^r \bmod P$	$\xrightarrow{PK_U, M, \bar{M}, \theta, \langle W,U,V\rangle, \mathsf{com}}$	
		$d \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_0}$
$s_0 = k_0 - d \cdot m$(in $\mathbb{Z}$)	$\xleftarrow{d}$	
$s_1 = k_1 \cdot l_1^{-d} \bmod \mathsf{n}$ $s_2 = k_2 - d \cdot l_2 \bmod p$ $s_3 = k_3 - d \cdot l_3 \bmod p$		
	$\xrightarrow{\substack{\langle s_0,s_1,s_2,s_3\rangle, \\ \langle T_M,T_{\bar{M}},T_W,T_U,T_V\rangle, r}}$	
		$M \in^? \mathbb{Z}^*_{\mathsf{n}^2}$, $\bar{M} \in^? \mathbb{Z}^*_{\bar{\mathsf{n}}^2}$ $s_0 \in^? \pm[0, 2^{\lambda_0+\lambda_1+\ell_p+1}]$ $t_M = \psi_1(T_M)$, $t_{\bar{M}} = \psi_1(T_{\bar{M}})$ $t_W = \psi_2(T_W)$, $t_U = \psi_2(T_U)$ $t_V = \psi_2(T_V)$ $\mathsf{com} =^? h_M^{t_M} h_{\bar{M}}^{t_{\bar{M}}} h_W^{t_W} h_U^{t_U} h_V^{t_V} h_r^r \bmod P$ $T_M =^? \mathsf{g}^{s_0} s_1^{\mathsf{n}} \bar{M}^d \bmod \mathsf{n}^2$ $T_{\bar{M}} =^? \bar{\mathsf{g}}^{s_0} s_1^{\bar{\mathsf{n}}} \bar{M}^d \bmod \bar{\mathsf{n}}^2$ $T_W =^? \theta^{s_0} w^{s_2+s_3} W^d$ $T_U =^? u^{s_2} U^d$, $T_V =^? v^{s_3} V^d$ $t', l'_2, l'_3 \xleftarrow{\mathtt{r}} \mathbb{Z}_p$ $a' = (\theta)^{t'}$, $b' = (\theta^y)^{t'}$ $W' = (W^{xy} \theta^x w^{l'_2+l'_3})^{t'}$ $U' = (U^{xy} u^{l'_2})^{t'}$, $V' = (V^{xy} v^{l'_3})^{t'}$
$a = (a')^t$, $b = (b')^t$	$\xleftarrow{a', b', \langle W', U', V'\rangle}$	
$c = (W'/(U'^{\delta} V'^{\xi}))^t$ $\sigma = \langle a, b, c\rangle$ $\mathtt{VERIFY}(m, \sigma) =^? 1$		
output $(m; \sigma)$		

Figure 3: The signing protocol of blind signature with revocable blindness.

In the unforgeability attack, even $\mathcal{T}$ can be corrupted by the adversary user, we can use the similar argument as that in the proof of Theorem 4.3: extract m from M (or from $\bar{M}$); and then use m to complete the simulation. The unforgeability then reduces to the unforgeability of the CL-signature [CL04] which is based on the LRSW assumption, and the binding property of the Pedersen commitment which is based on the DLOG assumption.

In the blindness attack, assume $\mathcal{T}$ is not corrupted by the adversary signer. Under the DCR and the DLDH assumptions, the adversary cannot distinguish the Paillier ciphertexts $M, \bar{M}$, the Linear Encryption ciphertexts $\langle W, U, V \rangle$ for message m from that for a random message, which allows us to develop the blindness proof by the same way in the proof of Theorem 4.4.

Based on the argument above, we can obtain the security theorem for the blind signature with revocable blindness as below:

Theorem 5.1. *Under the LRSW and the DLOG assumptions, the blind signature with revocable blindness defined above is unforgeable even if the trusted third party can be corrupted by the adversary; Under the DLDH and the DCR assumptions, the blind signature with revocable blindness defined above satisfies blindness, assuming that the trusted third party is not corrupted by the adversary.*

5.3 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 this signature messages are of the form $\langle m, \mathsf{info} \rangle$. The public information info is included into $ComInfo$. Here $\mathsf{info} \in [0, 2^{\ell_p}]$. Note that the exact choice for the value of info 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 $PK_S = \langle X, Y \rangle$ and $SK_S = \langle x, y \rangle$ they are substituted with $PK_S = \langle X, Y, Z \rangle$, $SK_S = \langle x, y, z \rangle$, where $X = g^x$, $Y = g^y$, $Z = g^z$. Signing a message $\langle m, \mathsf{info} \rangle$ corresponds to the following operation: select a random $a \in \mathbb{G}$ and output the signature $\sigma = \langle a, a^z, a^y, a^{yz}, a^{x+xym+xyz \cdot \mathsf{info}} \rangle$.

The modified signature has the following verification process: Given a message-signature pair $(m, \mathsf{info}; \sigma)$, 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 $e(X, a)e(X, b)^m e(X, B)^{\mathsf{info}} = e(g, c)$.

The detailed partially blind signature generation is similar to our basic blind signature protocol (i.e., it retains the 2-round structure with short communication) and is shown in detail in Figure 4.

Obviously, keeping info fixed across protocol executions it is straightforward to extract the blindness of the above scheme in a similar fashion as in the basic primitive, which is also based on the DLDH and the DCR assumptions. Unfogerability on the other hand reduces to the security of the Camenisch-Lysyanskaya two message-block signature [CL04] which is also based on the LRSW assumption, and the binding property of the Pedersen multi-commitment[Ped91] which is based on the DLOG assumption. Now we can obtain the security theorem of the proposed scheme.

Theorem 5.2. *Under the LRSW and the DLOG assumptions, the proposed partially blind signature scheme is unforgeable even if the public-tag is adversarially selected for each signature; Under the DLDH and the DCR assumptions, the proposed scheme is blind for signatures with the same public-tag.*

$\boxed{\mathcal{U}}$		$\boxed{\mathcal{S}}$
$ComInfo = \langle p, g, \mathbb{G}, \mathbb{G}_T, e; \psi_1, \psi_2; \mathsf{info}\rangle$		$ComInfo = \langle p, g, \mathbb{G}, \mathbb{G}_T, e; \psi_1, \psi_2; \mathsf{info}\rangle$
$CRS = \langle \mathsf{n}, \mathsf{g}; h_M, h_W, h_U, h_V, h_r, P, Q\rangle$		$CRS = \langle \mathsf{n}, \mathsf{g}; h_M, h_W, h_U, h_V, h_r, P, Q\rangle$
$PK_S = \langle X, Y, Z\rangle$		$PK_S = \langle X, Y, Z\rangle$, $SK_S = \langle x, y, z\rangle$
$MSG = \langle m\rangle$, $m \in [0, 2^{\ell_p}]$		
$(PK_U, SK_U) \leftarrow \mathtt{GEN}^{LE}(1^\lambda)$		
$PK_U = \langle u, v, w\rangle$, $SK_U = \langle \delta, \xi\rangle$		
$k_0 \xleftarrow{\mathtt{r}} \pm[0, 2^{\lambda_0+\lambda_1+\ell_p}]$, $l_1, k_1 \xleftarrow{\mathtt{r}} \mathbb{Z}^*_{\mathsf{n}}$		
$t, l_2, l_3, k_2, k_3 \xleftarrow{\mathtt{r}} \mathbb{Z}_p$, $\theta \xleftarrow{\mathtt{r}} \mathbb{G}$, $r \xleftarrow{\mathtt{r}} \mathbb{Z}_Q$		
$M = \mathsf{g}^m l_1^{\mathsf{n}} \bmod \mathsf{n}^2$		
$W = \theta^m w^{l_2+l_3}$, $U = u^{l_2}$, $V = v^{l_3}$		
$T_M = \mathsf{g}^{k_0} k_1^{\mathsf{n}} \bmod \mathsf{n}^2$		
$T_W = \theta^{k_0} w^{k_2+k_3}$, $T_U = u^{k_2}$, $T_V = v^{k_3}$		
$t_M = \psi_1(T_M)$, $t_W = \psi_2(T_W)$		
$t_U = \psi_2(T_U)$, $t_V = \psi_2(T_V)$		
$\mathsf{com} = h_M^{t_M} h_W^{t_W} h_U^{t_U} h_V^{t_V} h_r^r \bmod P$		
	$\xrightarrow{PK_U, M, \theta, \langle W,U,V\rangle, \mathsf{com}}$	
		$d \xleftarrow{\mathtt{r}} \{0,1\}^{\lambda_0}$
	$\xleftarrow{d}$	
$s_0 = k_0 - d \cdot m (\text{in } \mathbb{Z})$		
$s_1 = k_1 \cdot l_1^{-d} \bmod \mathsf{n}$		
$s_2 = k_2 - d \cdot l_2 \bmod p$		
$s_3 = k_3 - d \cdot l_3 \bmod p$		
	$\xrightarrow{\langle s_0,s_1,s_2,s_3\rangle, \langle T_M,T_W,T_U,T_V,r\rangle}$	
		$M \in^? \mathbb{Z}^*_{\mathsf{n}^2}$
		$s_0 \in^? \pm[0, 2^{\lambda_0+\lambda_1+\ell_p+1}]$
		$t_M = \psi_1(T_M)$, $t_W = \psi_2(T_W)$
		$t_U = \psi_2(T_U)$, $t_V = \psi_2(T_V)$
		$\mathsf{com} =^? h_M^{t_M} h_W^{t_W} h_U^{t_U} h_V^{t_V} h_r^r \bmod P$
		$T_M =^? \mathsf{g}^{s_0} s_1^{\mathsf{n}} M^d \bmod \mathsf{n}^2$
		$T_W =^? \theta^{s_0} w^{s_2+s_3} W^d$
		$T_U =^? u^{s_2} U^d$, $T_V =^? v^{s_3} V^d$
		$t', l'_2, l'_3 \xleftarrow{\mathtt{r}} \mathbb{Z}_p$
		$a' = (\theta)^{t'}$, $A' = (\theta^z)^{t'}$
		$b' = (\theta^y)^{t'}$, $b' = (\theta^{yz})^{t'}$
		$W' = (W^{xy} \theta^{x+xyz \cdot \mathsf{info}} w^{l'_2+l'_3})^{t'}$
		$U' = (U^{xy} u^{l'_2})^{t'}$, $V' = (V^{xy} v^{l'_3})^{t'}$
	$\xleftarrow{a', A', b', B', \langle W',U',V'\rangle}$	
$a = (a')^t$, $A = (A')^t$,		
$b = (b')^t$, $B = (B')^t$,		
$c = (W'/(U'^{\delta} V'^{\xi}))^t$		
$\sigma = \langle a, A, b, B, c\rangle$		
$\mathtt{VERIFY}(m, \mathsf{info}, \sigma) =^? 1$		
output $(m, \mathsf{info}; \sigma)$		

Figure 4: 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.

[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.

[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. Lessions from internet voting during 2002 fifa worldcup korea/japan(tm). In *DIMACS Workshop on Electronic Voting – Theory and Practice*, 2004.

[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.

[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.

[SPC95] Markus Stadler, Jean-Marc Piveteau, and Jan Camenisch. Fair blind signatures. In Louis C. Guillou and Jean-Jacques Quisquater, editors, *EUROCRYPT 1995*, volume 921 of *Lecture Notes in Computer Science*, pages 209–219. Springer, 1995.

[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.