Equivocal Blind Signatures and Adaptive UC-Security

Aggelos Kiayias* and Hong-Sheng Zhou*

Computer Science and Engineering
University of Connecticut
Storrs, CT, USA
{aggelos,hszhou}@cse.uconn.edu

Abstract. We study the design of adaptively secure blind signatures in the universal composability (UC) setting. First, we introduce a new property for blind signature schemes that is suitable for arguing security against adaptive adversaries: an *equivocal blind signature* is a blind signature where there exists a simulator that has the power of making signing transcripts correspond to any message signature pair. Second, we present a general construction methodology for building adaptively secure blind signatures: the starting point is a 2-move "equivocal lite blind signature", a lightweight 2-party signature protocol that we formalize and implement both generically as well as concretely; formalizing a primitive as "lite" means that the adversary is required to show all private tapes of adversarially controlled parties; this enables us to conveniently separate zero-knowledge (ZK) related security requirements from the remaining security properties in the blind signature design methodology. Next, we focus on the suitable ZK protocols for blind signatures. We formalize two special ZK ideal functionalities, single-verifier-ZK (SVZK) and single-prover-ZK (SPZK), both special cases of multi-session ZK that may be of independent interest, and we investigate the requirements for realizing them in a commit-and-prove fashion as building blocks for adaptively secure UC blind signatures. Regarding SPZK we find the rather surprising result that realizing it only against static adversaries is sufficient to obtain adaptive security for UC blind signatures.

We instantiate all the building blocks of our design methodology both generically based on the blind signature construction of Fischlin as well as concretely based on the 2SDH assumption of Okamoto, thus demonstrating the feasibility and practicality of our approach. The latter construction yields the first practical UC blind signature that is secure against adaptive adversaries. We also present a new more general modeling of the ideal blind signature functionality.

1 Introduction

A blind signature is a cryptographic primitive that was proposed by Chaum [12]; it is a digital signature scheme where the signing algorithm is

* Research partly supported by NSF CAREER Award CNS-0447808.

split into a two-party protocol between a user (or client) and a signer (or server). The signing protocol's functionality is that the user can obtain a signature on a message that she selects in a blind fashion, i.e., without the signer being able to extract some useful information about the message from the protocol interaction. At the same time the existential unforgeability property of digital signatures should hold, i.e., after the successful termination of a number of n corrupted user instantiations, an adversary should be incapable of generating signatures for $(n+1)$ distinct messages.

A blind signature is a very useful privacy primitive that has many applications in the design of electronic-cash schemes, the design of electronic voting schemes as well as in the design of anonymous credential systems. Since the initial introduction of the primitive, a number of constructions have been proposed [13, 32, 30, 35, 23, 36, 34, 37, 3, 1, 2, 5–7, 24, 31, 17, 21, 8]. The first formal treatment of the primitive in a stand-alone model and assuming random oracles (RO) was given by Pointcheval and Stern in [35].

Blind signatures is in fact one of the few complex cryptographic primitives (beyond digital signatures, public-key encryption, and key-exchange) that have been implemented in real world Internet settings (e.g., in the Votopia [27] voting system) and thus the investigation of more realistic attack models for blind signatures is of pressing importance. Juels, Luby and Ostrovsky [23] presented a formal treatment of blind signatures that included the possibility for an adversary to launch attacks that use arbitrary concurrent interleaving of either user or signer protocols. Still, the design of schemes that satisfied such stronger modeling proved somewhat elusive. In fact, Lindell [28] showed that unbounded concurrent security for blind signatures is impossible under a simulation-based security definition without any setup assumption; more recently in [21], the generic feasibility of blind signatures without setup assumptions was shown but using a game-based security formulation.

With respect to practical provably secure schemes, assuming random oracles or some setup assumption, various efficient constructions were proposed: for example, [5, 6] presented efficient two-move constructions in the RO model, while [24, 31] presented efficient constant-round constructions without random oracles employing a common reference string (CRS) model (i.e., when a trusted setup function initializes all parties' inputs) that withstand concurrent attacks. While achieving security under concurrent attacks is an important property for the design of useful blind signatures, a blind signature scheme may still be insecure for a certain deployment. Game-based security definitions [35, 23, 7, 24, 31, 21, 8] capture properties that are intuitively desirable. Nevertheless, the successive amendments of definitions in the literature and the differences between the various models exemplify the following: on the one hand capturing all desirable properties of a complex cryptographic primitive such as a blind signature is a difficult task, while on the other, even if such properties are attained, a "provably secure" blind signature may still be insecure if deployed within a larger system. For this reason, it is important to consider the realization of blind signatures under a general simulation-based security formulation such as the one provided in the Universal Composability (UC) framework of Canetti [9] that enables us to formulate cryptographic primitives so that they remain secure under arbitrary deployments and interleavings of protocol instantiations.

In the UC setting, against static adversaries, it was shown how to construct blind signatures in the CRS model [17] with two moves of interaction. Though the construction in [17] is round-optimal, it is unknown whether it can admit concrete practical instantiations. In addition, security is argued only against static adversaries; and while it should be feasible to extend the construction of [17] in the adaptive setting this can only exacerbate the difficulty of concretely realizing the basic design. Note that using the secure two party computation compiler of [11] one can derive adaptively secure blind signatures but this approach is also generic and does not suggest any concrete design.

1.1 Our Results

In this work we study the design of blind signatures in the UC framework against adaptive adversaries. Our approach is "practice-oriented" in the sense of minimizing communication complexity as well as entailing the following points: (i) a constant number of rounds, (ii) a choice of session scope that is consistent with how a blind signature would be implemented in practice, in particular a multitude of clients and one signer should be supported within a single session, (iii) a trusted setup string that is of constant length in the number of parties within a session, (iv) the avoidance, if possible, of cryptographic primitives that are "per-bit", such as bit-commitment, where one has to spend a communication length of $\Omega(l)$ where l is a security parameter per bit of private input. Our results are as follows:

Equivocal blind signatures. We introduce a new property for blind signatures, called equivocality that is suitable for arguing security against adaptive adversaries. In an equivocal blind signature there exists a simulator that has the power to construct the internal state of a client including all random tapes so that any simulated communication transcript can be mapped to any given valid message-signature pair. This capability should hold true even after a signature corresponding to the simulated transcript has been released to the adversary. Equivocality can be seen as a strengthening of the notion of blindness as typically defined in game-based security formulations of blind signatures: in an equivocal blind signature, signing transcripts can be simulated in an independent fashion to the message-signature pair they correspond to.

General methodology for building UC blind signatures. We present a general methodology for designing adaptively secure UC blind signatures. Our starting point is the notion of an *equivocal lite blind signature*: The idea behind "lite" blind signatures is that security properties should hold under the condition that the adversary deposits the private tapes of the parties he controls. This "open-all-private-tapes" approach simplifies the blind signature definitions substantially and allows one to separate security properties that relate to zero-knowledge compared to other necessary properties for blind signatures. Note that this is not an honest-but-curious type of adversarial formulation as the adversary is not required to be honestly simulating corrupted parties; in particular, the adversary may deviate from honest protocol specification (e.g., bias the random tapes) as long as he can present private tapes that match the communication transcripts.

We then demonstrate two instantiations of an equivocal lite blind signature, one that is based on generic cryptographic primitives that is inspired by the blind signature construction of [17] and one based on the design and the 2SDH assumption of [31].

Study of the ZK requirements for UC blind signatures. Having demonstrated equivocal lite blind-signatures as a feasible starting building block, we then focus on the formulation of the appropriate ZK-functionalities that are required for building blind signatures in the adaptive adversary setting. Interestingly, the user and the signer have different ZK "needs" in a blind signature. In particular the corresponding ZK-functionalities turn out to be simplifications of the standard multi-session ZK functionality $\mathcal{F}_{\mathrm{MZK}}$ that restrict the multi-sessions to occur either from many provers to a single verifier (we call this $\mathcal{F}_{\mathrm{SVZK}}$) or from a single prover to many verifiers (we call this $\mathcal{F}_{\mathrm{SPZK}}$). Note that this stems from our blind signature *session scope* that involves a multitude of users interacting with a single signer: this is consistent with the notion that a blind-signature signer is a server within a larger system and is expected that the number of such servers would be very small compared to a much larger population of users and verifiers.

First, regarding $\mathcal{F}_{\mathrm{SVZK}}$, the ZK protocol that users need to execute as provers, we show that it can be realized in a commit-and-prove fashion using a commitment scheme that, as it is restricted to the single-verifier setting, it does not require built-in non-malleability (while such property would be essential for general multi-session UC commitments). We thus proceed to realize $\mathcal{F}_{\mathrm{SVZK}}$ using mixed commitments [16, 29] with only a constant length common reference string (as opposed to linear in the number of parties that is required in the multi-session setting). Second, regarding $\mathcal{F}_{\mathrm{SPZK}}$, the ZK protocol the signer needs to execute as a prover, we find the rather surprising result that it needs only be realized against static adversaries for the resulting blind signature scheme to satisfy adaptive security. This enables a much more efficient realization design for $\mathcal{F}_{\mathrm{SPZK}}$ as we can implement it using merely an extractable commitment and a Sigma protocol (alternatively, using an Ω-protocol [18]). The intuition behind this result is that in a blind signature the signer is not interested in hiding his input in the same way that the user is: this can be seen by the fact that the verification-key itself leaks a lot of information about the signing-key to the adversary/environment, thus, using a full-fledged zero-knowledge instantiation is an overkill from the signer's point of view; this phenomenon was studied in the context of zero-knowledge in [26]. We note that our $\mathcal{F}_{\mathrm{SPZK}}$ functionality can be seen as a special instance of client-server computation as considered in [38] (where the relaxed non-malleability requirement of such protocols was also noted); interestingly $\mathcal{F}_{\mathrm{SVZK}}$ falls outside that framework (despite its client-server nature).

Notations: $a \stackrel{\mathtt{r}}{\leftarrow} \mathtt{RND}$ denotes randomly selecting a in its domain; $\mathsf{negl}()$ denotes negligible function; $\mathsf{poly}()$ denotes polynomial function.

2 Equivocal Lite Blind Signatures

2.1 Building Block: Equivocal Lite Blind Signatures

A signature generation protocol is a tuple $\langle \mathtt{CRSgen}, \mathtt{gen}, \mathsf{lbs}_1, \mathsf{lbs}_2, \mathsf{lbs}_3, \mathtt{verify} \rangle$ where $\mathtt{CRSgen}$ is a common reference string generation algorithm, $\mathtt{gen}$ is a key-pair generation algorithm, lbs_i, $i = 1, 2, 3$, comprise a two-move signature generation protocol between the user U and the signer S as described in Figure 1 and $\mathtt{verify}$ is a signature verification algorithm. A *lite blind signature* is a signature generation protocol that satisfies completeness (see definition 1) as well as two security properties, *lite-unforgeability* and *lite-blindness*, defined below (consistency is another property [10] for signatures that will be trivially satisfied in our design and thus we omit it in this version).

$\boxed{U}$		$\boxed{S}$
CRS $= \langle crs \rangle$ VerificationKey $= \langle vk \rangle$ Plaintext $= \langle m \rangle$		CRS $= \langle crs \rangle$ VerificationKey $= \langle vk \rangle$ SigningKey $= \langle sk \rangle$
$\rho_1 \xleftarrow{r} \texttt{RND}$		
$\mathbf{u} \leftarrow \mathsf{lbs}_1(crs, vk, m; \rho_1)$	$\xrightarrow{\mathbf{u}}$	$\rho_2 \xleftarrow{r} \texttt{RND}$
		$\mathbf{s} \leftarrow \mathsf{lbs}_2(crs, vk, \mathbf{u}, sk; \rho_2)$
$\rho_3 \xleftarrow{r} \texttt{RND}$	$\xleftarrow{\mathbf{s}}$	
$\sigma \leftarrow \mathsf{lbs}_3(crs, vk, m, \rho_1, \mathbf{u}, \mathbf{s}; \rho_3)$		

Fig. 1. Outline of a two-move signature generation protocol.

Definition 1 (Completeness). *A signature generation protocol as in Figure 1 is complete if for all* $(crs, \tau) \leftarrow \texttt{CRSgen}(1^\lambda)$*, for all* $(vk, sk) \leftarrow \texttt{gen}(crs)$*, for all* $\rho_1, \rho_2, \rho_3 \xleftarrow{r} \texttt{RND}$*, whenever* $\mathbf{u} \leftarrow \mathsf{lbs}_1(crs, vk, m; \rho_1)$*,* $\mathbf{s} \leftarrow \mathsf{lbs}_2(crs, vk, \mathbf{u}, sk; \rho_2)$*, and* $\sigma \leftarrow \mathsf{lbs}_3(crs, vk, m, \rho_1, \mathbf{u}, \mathbf{s}; \rho_3)$*, then* $\texttt{verify}(crs, vk, m, \sigma) = 1$*.*

Lite-unforgeability that we define below suggests informally that if we "collapse" the $\mathsf{lbs}_1, \mathsf{lbs}_2$ procedures into a single algorithm this will result to a procedure that combined with lbs_3 will be equivalent to the signing algorithm of an unforgeable digital signature `sign` in the sense of [19]. We note that lite-unforgeability is much weaker compared to regular unforgeability of blind signatures (as defined e.g., in [31, 21]) since it requires from the adversary to open the internal tapes of each user instance (as opposed to hiding such internals in the usual unforgeability definition for blind signatures); note that this is not an honest-but-curious modeling as the adversary is not restricted to flip coins honestly.

Definition 2 (Lite-unforgeability). *A signature generation protocol as in Figure 1 is lite-unforgeable if for all PPT* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *and for any* $L = \mathsf{poly}(\lambda)$*, we have* $\mathsf{Adv}^{\mathcal{A},L}_{\mathrm{luf}}(\lambda) \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{Adv}^{\mathcal{A},L}_{\mathrm{luf}}(\lambda) \stackrel{\mathsf{def}}{=} \Pr[\mathbf{Exp}^{\mathrm{LUF}}_{\mathcal{A},L}(\lambda) = 1]$ *and the experiment* $\mathbf{Exp}^{\mathrm{LUF}}_{\mathcal{A},L}(\lambda)$ *is defined below:*

Experiment $\mathbf{Exp}^{\mathrm{LUF}}_{\mathcal{A},L}(\lambda)$
$(crs,\tau) \leftarrow \texttt{CRSgen}(1^\lambda)$; $(vk, sk) \leftarrow \texttt{gen}(crs)$; $state := \emptyset$; $k := 0$;
while $k < L$
$(m_k, \rho_{1,k}, state) \leftarrow \mathcal{A}_1(state, crs, vk)$;
$\mathbf{s}_k \leftarrow \mathsf{lbs}_2(crs, vk, \mathsf{lbs}_1(crs, vk, m_k; \rho_{1,k}), sk; \rho_{2,k})$; $\rho_{2,k} \stackrel{\mathtt{r}}{\leftarrow} \mathtt{RND}$;
$state \leftarrow state || \mathbf{s}_k$; $k \leftarrow k+1$;
$(m_1, \sigma_1, \ldots, m_\ell, \sigma_\ell) \leftarrow \mathcal{A}_2(state)$;
if $\ell > L$, *and* $\texttt{verify}(crs, vk, m_i, \sigma_i) = 1$ *for all* $1 \le i \le \ell$,
and $m_i \neq m_j$ *for all* $1 \le i \neq j \le \ell$
then return 1 *else return* 0.

Similarly we can formulate blindness (as defined, e.g. in [8]) in the "lite" setting by requiring the adversary to open the private tape of the signer for each user interaction. Given that blindness is subsumed by our equivocality property (defined below), we will not explore this direction further here (the reader may refer to the full version [25] for more details). For simplicity we define equivocality only for two-move protocols following the skeleton of Figure 1. Informally an equivocal blind signature scheme is accompanied by a simulator procedure $\mathcal{I}$ which can produce signature generation transcripts without using the user input m and furthermore it can "explain" the transcripts to any adversarially selected m even after the signature σ for m has been generated. The property of equivocal blind signatures parallels the property of equivocal commitments [4] or zero-knowledge with state reconstruction, cf. [20]. We define the property formally below (cf. Figure 2). In the definition, we use the relation $\mathsf{KeyPair}$ defined as $(vk, sk) \in \mathsf{KeyPair}$ if and only if $(vk, sk) \leftarrow \mathsf{gen}(crs)$ (omitting crs to avoid cluttering the notation). Note that we require $(vk, sk), (vk, sk') \in \mathsf{KeyPair}$ to imply $sk = sk'$ (otherwise a blind signature *may* be susceptible to an attack due to [22]).

Definition 3 (Equivocality). *We say that a signature generation protocol is equivocal if there exists an interactive machine $\mathcal{I} = (\mathcal{I}_1, \mathcal{I}_2)$, such that for all PPT $\mathcal{A}$, we have $\mathsf{Adv}^{\mathcal{A}}_{\mathrm{eq}}(\lambda) \le \mathsf{negl}(\lambda)$,*

$$\mathsf{Adv}^{\mathcal{A}}_{\mathrm{eq}}(\lambda) \stackrel{\mathsf{def}}{=} \left| \begin{array}{l} \Pr[(crs,\tau) \leftarrow \texttt{CRSgen}(1^\lambda) : \mathcal{A}^{Users(crs,\cdot)}(crs) = 1] \\ \qquad - \Pr[(crs,\tau) \leftarrow \texttt{CRSgen}(1^\lambda) : \mathcal{A}^{\mathcal{I}(crs,\tau,\cdot)}(crs) = 1] \end{array} \right|,$$

where oracle $Users(crs, \cdot)$ operates as:

- *Upon receiving message (i, m, vk) from $\mathcal{A}$, select $\rho_1 \stackrel{\mathtt{r}}{\leftarrow} \mathtt{RND}$ and compute $\mathbf{u} \leftarrow \mathsf{lbs}_1(crs, vk, m; \rho_1)$, record $\langle i, m, vk, \mathbf{u}, \rho_1 \rangle$ into $history_i$, and return message $(i, \mathbf{u})$ to $\mathcal{A}$.*

- *Upon receiving message* $(i, \mathbf{s}, \rho_2, sk)$ *from* $\mathcal{A}$*, if there exists a record* $\langle i, m, vk, \mathbf{u}, \rho_1 \rangle$ *in* $history_i$ *and both* $(vk, sk) \in \mathsf{KeyPair}$ *and* $\mathbf{s} = \mathsf{lbs}_2(crs, vk, \mathbf{u}, sk; \rho_2)$ *hold, then select* $\rho_3 \stackrel{\mathtt{r}}{\leftarrow} \mathtt{RND}$*, compute* $\sigma \leftarrow \mathsf{lbs}_3(crs, vk, m, \rho_1, \mathbf{u}, \mathbf{s}; \rho_3)$*, update* $\langle i, m, vk, \mathbf{u}, \rho_1 \rangle$ *in* $history_i$ *into* $\langle i, m, vk, \mathbf{u}, \sigma, \rho_1, \rho_3 \rangle$*, and return to* $\mathcal{A}$ *the pair* (i, σ)*; otherwise return to* $\mathcal{A}$ *the pair* $(i, \perp)$*.*
- *Upon receiving message* (i, open)*, return to* $\mathcal{A}$ *the pair* $(i, history_i)$*.*

and oracle $\mathcal{I}(crs, \tau, \cdot)$ *operates as:*

- *Upon receiving message* (i, m, vk) *from* $\mathcal{A}$*, run* $(\mathbf{u}, aux) \leftarrow \mathcal{I}_1(crs, \tau, vk)$*, record* $\langle i, m, vk, \mathbf{u}, aux \rangle$ *into temp, and return message* $(i, \mathbf{u})$ *to* $\mathcal{A}$*.*
- *Upon receiving message* $(i, \mathbf{s}, \rho_2, sk)$ *from* $\mathcal{A}$*, if there exists a record* $\langle i, m, vk, \mathbf{u}, aux \rangle$ *in temp and both* $(vk, sk) \in \mathsf{KeyPair}$ *and* $\mathbf{s} = \mathsf{lbs}_2(crs, vk, \mathbf{u}, sk; \rho_2)$ *hold, then select* $\gamma \stackrel{\mathtt{r}}{\leftarrow} \mathtt{RND}$*, compute* $\sigma \leftarrow \mathtt{sign}(crs, vk, sk, m, \gamma)$ *(where* $\mathtt{sign}$ *is the "collapse" of* lbs_i *for* $i = 1, 2, 3$*), update* $\langle i, m, vk, \mathbf{u} \rangle$ *in temp into* $\langle i, m, vk, \mathbf{u}, aux; \mathbf{s}, sk, \rho_2; \sigma, \gamma \rangle$*, and return the pair* (i, σ) *to* $\mathcal{A}$*; otherwise return to* $\mathcal{A}$ *the pair* $(i, \perp)$*.*
- *Upon receiving message* (i, open)*, if there exists a record* $\langle i, m, vk, \mathbf{u}, aux \rangle$ *in temp then run* $\rho_1 \leftarrow \mathcal{I}_2(i, temp)$*, record* $\langle i, m, vk, \mathbf{u}, \rho_1 \rangle$ *into* $history_i$*, and return to* $\mathcal{A}$ *the pair* $(i, history_i)$*; if there exists a record* $\langle i, m, vk, \mathbf{u}, aux; \mathbf{s}, sk, \rho_2; \sigma, \gamma \rangle$ *in temp, then run* $(\rho_1, \rho_3) \leftarrow \mathcal{I}_2(i, temp)$*, record* $\langle i, m, vk, \mathbf{u}, \sigma, \rho_1, \rho_3 \rangle$ *into* $history_i$*, and return message* $(i, history_i)$ *to* $\mathcal{A}$*.*

We call a signature generation protocol that satisfies completeness, lite-unforgeability as well as the equivocality property an *equivocal lite blind signature scheme.*

2.2 Constructions

In this subsection, we present two equivocal lite-blind signature constructions. The first construction is generic and is based on the blind signature design of [17] whereas the second is a concrete construction that is based on [31]. In the full version of this work [25] we present additional constructions.

Generic equivocal lite blind signature. Our first construction is based on [17]; the main difference here is that we need the equivocality property (the original design employed two encryption steps for the user that are non-equivocal); in our setting, it is sufficient to have just

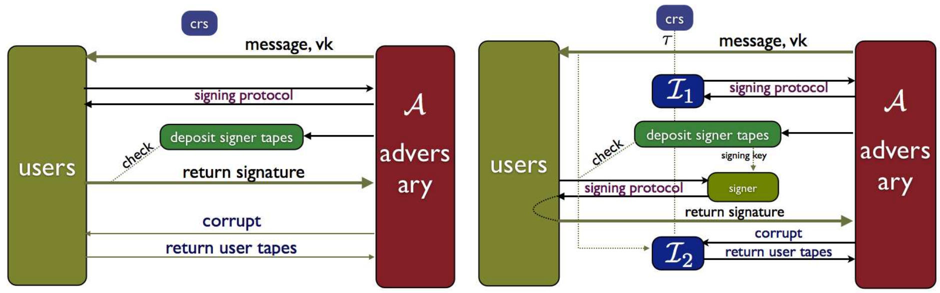

Fig. 2. The two worlds an equivocality adversary is asked to distinguish in Definition 3. In the left-hand the adversary is interacting with a set of users whereas in the right-hand side the users are interacting with an honest signer instantiation whereas the adversary is interacting with the simulator $\mathcal{I}$.

one equivocal commitment (that is not extractable) in the first stage and then employ an extractable commitment in the second (that is not equivocal). Refer to the signature generation protocol in Figure 3: the `CRSgen` algorithm produces $crs = \langle pk_{\text{eqc}}, pk_{\text{exc}}, crs_{\text{nizk}} \rangle$; `EQC` is a commitment scheme with committing key pk_{eqc} and `EQCcom` is its committing algorithm; `EXC` is a commitment scheme with committing key pk_{exc} and `EXCcom` is its committing algorithm; `NIZK` is an NIZK argument scheme with CRS crs_{nizk} where `NIZKprove` is the proof generation algorithm and `NIZKverify` is the proof verification algorithm. The `gen` algorithm produces a key-pair $\langle vk, sk \rangle$ for a signature scheme `SIG` where `SIGsign` is the signature generation algorithm and `SIGverify` is the corresponding verification algorithm. The language $\mathcal{L}_R \stackrel{\text{def}}{=} \{x | (x, w) \in R\}$ where $R \stackrel{\text{def}}{=} \{(crs, vk, E, m), (\mathbf{u}, \mathbf{s}, \rho_1, \rho_3) \,|\, \mathbf{u} = \texttt{EQCcom}(pk_{\text{eqc}}, m; \rho_1) \wedge \texttt{SIGverify}(vk, \mathbf{u}, \mathbf{s}) = 1 \wedge E = \texttt{EXCcom}(pk_{\text{exc}}, \mathbf{u}, \mathbf{s}; \rho_3)\}$. The `verify` algorithm given a message m and signature σ operates as follows: parse σ into E and ϖ, and check that $\texttt{NIZKverify}((crs, vk, E, m), \varpi) =^? 1$.

Theorem 1. *The two-move signature generation protocol in Figure 3 is an equivocal lite blind signature as follows: it satisfies lite-unforgeability provided that (i)* `SIG` *is EU-CMA secure, (ii)* `EQC` *is binding, (iii)* `EXC` *is extractable, and (iv)* `NIZK` *satisfies soundness; and it satisfies equivocality provided that (i)* `EQC` *is equivocal, (ii)* `EXC` *is hiding, and (iii)* `NIZK` *is non-erasure zero-knowledge.*

Concrete equivocal lite blind signature. In Figure 4 we present a lite blind signature $\langle \texttt{CRSgen}, \texttt{gen}, \mathsf{lbs}_1, \mathsf{lbs}_2, \mathsf{lbs}_3, \texttt{verify} \rangle$ that uses the 2SDH

$crs = \langle pk_{\text{eqc}}, pk_{\text{exc}}, crs_{\text{nizk}} \rangle$

$\boxed{U}$		$\boxed{S}$
VerificationKey $= \langle vk \rangle$		VerificationKey $= \langle vk \rangle$
Plaintext $= \langle m \rangle$		SigningKey $= \langle sk \rangle$
$\rho_1 \xleftarrow{r} \texttt{RND}$; $\mathbf{u} \leftarrow \texttt{EQCcom}(pk_{\text{eqc}}, m; \rho_1)$		
	$\xrightarrow{\mathbf{u}}$	$\rho_2 \xleftarrow{r} \texttt{RND}$
$\texttt{SIGverify}(vk, \mathbf{u}, \mathbf{s}) \stackrel{?}{=} 1$	$\xleftarrow{\mathbf{s}}$	$\mathbf{s} \leftarrow \texttt{SIGsign}(vk, sk, \mathbf{u}; \rho_2)$
$\rho_3, \rho_4 \xleftarrow{r} \texttt{RND}$; $E \leftarrow \texttt{EXCcom}(pk_{\text{exc}}, \mathbf{u}, \mathbf{s}; \rho_3)$		
$\varpi \leftarrow \texttt{NIZKprove}((crs, vk, E, m), (\mathbf{u}, \mathbf{s}, \rho_1, \rho_3); \rho_4$		
$: \mathbf{u} = \texttt{EQCcom}(pk_{\text{eqc}}, m; \rho_1)$		
$\wedge\, \texttt{SIGverify}(vk, \mathbf{u}, \mathbf{s}) = 1$		
$\wedge\, E = \texttt{EXCcom}(pk_{\text{exc}}, \mathbf{u}, \mathbf{s}; \rho_3))$		
$\sigma \leftarrow E \| \varpi$		
$\texttt{verify}(crs, vk, m, \sigma) \stackrel{?}{=} 1$		
output $(m; \sigma)$		

Fig. 3. A generic signature generation protocol.

assumption and is based on Okamoto's blind signature scheme [31]; the main contribution here is Theorem 2 that shows that the design is in fact equivocal (instead of merely blind as shown in [31]). In this scheme the CRSgen algorithm produces $crs = \langle p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{\mathsf{e}}, \psi, u_2, v_2 \rangle$, where $\hat{\mathsf{e}} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear map, $\mathbb{G}_1, \mathbb{G}_2$ are groups of order p, the gen algorithm produces a key-pair $vk = \langle X \rangle$, $sk = \langle x \rangle$ such that $X = g_2^x$, and the verify algorithm given a message m and signature $\sigma = \langle \varsigma, \alpha, \beta, V_1, V_2 \rangle$, responds as follows: check that $m, \beta \in \mathbb{Z}_p$, $\varsigma, V_1 \in \mathbb{G}_1$, $\alpha, V_2 \in \mathbb{G}_2$, $\varsigma \neq 1$, $\alpha \neq 1$ and $\hat{\mathsf{e}}(\varsigma, \alpha) = \hat{\mathsf{e}}(g_1, g_2^m u_2 v_2^\beta)$, $\hat{\mathsf{e}}(V_1, \alpha) = \hat{\mathsf{e}}(\psi(X), X) \cdot \hat{\mathsf{e}}(g_1, V_2)$.

Theorem 2. *The two-move protocol of Figure 4 is an equivocal lite blind signature as follows: it satisfies lite-unforgeability under the 2SDH assumption; and it satisfies equivocality unconditionally.*

3 Designing Adaptively Secure UC Blind Signatures

In this section we present our design methodology for constructing UC-blind signatures secure against adaptive adversaries, i.e., the protocol obtained by our method can UC-realize the blind signature functionality $\mathcal{F}_{\text{BSIG}}$ (defined in Figure 5). A previous formalization of the blind signature primitive in the UC setting was given by [17]. In the full version, we

$$crs = \langle p, g_1, g_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, \hat{\mathsf{e}}, \psi, u_2, v_2 \rangle$$

$$\begin{array}{lcl}
\boxed{U} & & \boxed{S} \\
vk = \langle X = g_2^x \rangle & & vk = \langle X = g_2^x \rangle \\
msg = \langle m \rangle,\ m \in \mathbb{Z}_p & & sk = \langle x \rangle \\
\hline
t, s \xleftarrow{\mathtt{r}} \mathbb{Z}_p;\ W \leftarrow g_1^{mt} u_1^t v_1^{st} & \xrightarrow{\quad W \quad} & \\
 & & r, l \xleftarrow{\mathtt{r}} \mathbb{Z}_p \text{ s.t. } x + r \neq 0 \\
f, h \xleftarrow{\mathtt{r}} \mathbb{Z}_p;\ \varsigma \leftarrow Y^{\frac{1}{ft} \bmod p} & \xleftarrow{\quad Y,l,r \quad} & Y \leftarrow (W v_1^l)^{\frac{1}{x+r}} \\
\alpha \leftarrow X^f g_2^{fr};\ \beta \leftarrow s + \frac{l}{t} \bmod p & & \\
V_1 \leftarrow \psi(X)^{\frac{1}{f}} g_1^h;\ V_2 \leftarrow X^{fh+r} g_2^{frh} & & \\
\sigma \leftarrow \langle \varsigma, \alpha, \beta, V_1, V_2 \rangle & & \\
\mathtt{verify}(crs, vk, m, \sigma) \stackrel{?}{=} 1 & & \\
\hline
\text{output } (m; \sigma) & &
\end{array}$$

Fig. 4. Signature generation protocol based on Okamoto digital signature [31].

include in the ideal functionality an explicit description of how corruptions are handled, and we justify our definition. Note that our $\mathcal{F}_{\text{BSIG}}$ does not require strong unforgeability from the underlying signing mechanism; this makes the presentation more general as strong unforgeability is not necessary for many applications of the blind signature primitive. Further, protocols realizing the functionality of [17] require a single "global trapdoor" that enables the functionality to produce a signature for a given message that will be valid for any given public-key; while this can be handy in the security proof, it is not a mandatory requirement for a UC-blind signature (which may allow for a different trapdoor to be used by the functionality in each signature generation); we reflect this in our ideal functionality by allowing the adversary in the corrupted signer setting to "patch" the ideal functionality with a different signing key for each user. In a blind signature session we allow for a single signer (whose identity is hard-coded into the session identifier *sid*) and a multitude of users. Our signer is active throughout the session and, after key-generation, is responsive to any user communicating with it via the network without waiting authorization by the environment.

Our design for UC-realizing $\mathcal{F}_{\text{BSIG}}$ is modular and delineates the components required for designing UC blind signatures in the adaptive security setting. We present our methodology in two steps. First, we employ an equivocal lite blind signature scheme and we operate in a hybrid world where the following ideal functionalities exist: $\mathcal{F}_{\text{CRS}}, \mathcal{F}_{\text{SVZK}}^{R_U}, \mathcal{F}_{\text{SPZK}}^{R_S}$. Here

Functionality $\mathcal{F}_{\text{BSIG}}$

Key generation: Upon receiving $(\textsc{KeyGen}, \mathit{sid})$ from party S, verify that $\mathit{sid} = (S, \mathit{sid}')$ for some sid'. If not, ignore the input. Else, forward $(\textsc{KeyGen}, \mathit{sid})$ to the adversary $\mathcal{S}$.

Upon receiving $(\textsc{Algorithms}, \mathit{sid}, \mathsf{sig}, \mathsf{ver})$ from the adversary $\mathcal{S}$, record $\langle \spadesuit, \mathsf{sig}, \mathsf{ver}\rangle$ in $\mathit{history}(S)$, and output $(\textsc{VerificationAlg}, \mathit{sid}, \mathsf{ver})$ to party S, where sig is a signing algorithm, and ver is a verification algorithm.

Signature generation: Upon receiving $(\textsc{Sign}, \mathit{sid}, m, \mathsf{ver}')$ from party $U \neq S$, where $\mathit{sid} = (S, \mathit{sid}')$, record $\langle m, \mathsf{ver}'\rangle$ in $\mathit{history}(U)$, and send $(\textsc{Sign}, \mathit{sid}, U, \mathsf{ver}')$ to the adversary $\mathcal{S}$.

Upon receiving $(\textsc{SignStatus}, \mathit{sid}, U, \mathsf{SignerComplete})$ from the adversary $\mathcal{S}$, where U is a user that has requested a signature, output $(\textsc{SignStatus}, \mathit{sid}, U, \mathsf{complete})$ to party S, and record $\langle U, \mathsf{complete}\rangle$ in $\mathit{history}(S)$.

Upon receiving $(\textsc{SignStatus}, \mathit{sid}, U, \mathsf{SignerError})$ from the adversary $\mathcal{S}$, where U is a user that has requested a signature, output $(\textsc{SignStatus}, \mathit{sid}, U, \mathsf{incomplete})$ to party S, and record $\langle U, \perp\rangle$ in $\mathit{history}(S)$.

Upon receiving $(\textsc{Signature}, \mathit{sid}, U, \mathsf{UserComplete})$ from the adversary $\mathcal{S}$, where U is a user that has requested a signature,

- if S is not corrupted and $\langle U, \mathsf{complete}\rangle$ is not recorded in $\mathit{history}(S)$, then halt.
- if S is not corrupted and $\langle U, \mathsf{complete}\rangle$ has been recorded in $\mathit{history}(S)$ that also contains $\langle \spadesuit, \mathsf{sig}, \mathsf{ver}\rangle$, then compute $\sigma \leftarrow \mathsf{sig}(m, \mathit{rnd})$ flipping the required random coins rnd, and do the following: if $\mathsf{ver}'(m, \sigma) \neq 1$, then halt; else if $\mathsf{ver}'(m, \sigma) = 1$ but $\mathsf{ver}(m, \sigma) \neq 1$, output $(\textsc{Signature}, \mathit{sid}, \sigma)$ to party U, and update $\mathit{history}(U)$ into $\langle m, \sigma, \mathit{rnd}\rangle$; else if $\mathsf{ver}'(m, \sigma) = \mathsf{ver}(m, \sigma) = 1$, output $(\textsc{Signature}, \mathit{sid}, \sigma)$ to party U, and update $\mathit{history}(U)$ into $\langle m, \sigma, \mathit{rnd}, \mathsf{done}\rangle$.
- else if S is corrupted, then compute $\sigma \leftarrow \mathsf{sig}'(m, \mathit{rnd})$ flipping the required random coins rnd, where sig' is an algorithm that the adversary $\mathcal{S}$ has provided specifically for U (subject to the restriction that any sig' corresponds to a single ver'), and do the following: if $\mathsf{ver}'(m, \sigma) = 1$, output $(\textsc{Signature}, \mathit{sid}, \sigma)$ to party U, update $\mathit{history}(U)$ into $\langle m, \sigma, \mathit{rnd}\rangle$; else if $\mathsf{ver}'(m, \sigma) \neq 1$, halt.

Upon receiving $(\textsc{Signature}, \mathit{sid}, U, \mathsf{UserError})$ from the adversary $\mathcal{S}$, where U is a user that has requested a signature, output $(\textsc{Signature}, \mathit{sid}, \perp)$ to party U and update $\mathit{history}(U)$ into $\langle m\rangle$.

Signature verification: Upon receiving $(\textsc{Verify}, \mathit{sid}, m, \sigma, \mathsf{ver}')$ from party V, where $\mathit{sid} = (S, \mathit{sid}')$, do: if, (i) the signer S is not corrupted, (ii) $\mathit{history}(S)$ contains $\langle \spadesuit, \mathsf{sig}, \mathsf{ver}\rangle$, (iii) $\mathsf{ver}' = \mathsf{ver}$, (iv) $\mathsf{ver}(m, \sigma) = 1$, and (v) there is no U such that m is recorded with done in $\mathit{history}(U)$, then halt. Else, output $(\textsc{Verified}, \mathit{sid}, \mathsf{ver}'(m, \sigma))$ to party V.

Fig. 5. Blind signature functionality $\mathcal{F}_{\text{BSIG}}$. Each session contains a signer and unlimited number of users. Each user U obtains at most one signature.

$\mathcal{F}_{\text{CRS}}$ will be an appropriate common reference string functionality; on the other hand, $\mathcal{F}^{R_U}_{\text{SVZK}}, \mathcal{F}^{R_S}_{\text{SPZK}}$ will be two *different* zero-knowledge functionalities that are variations of the standard multi-session ZK functionality. This reflects the fact that the ZK "needs" of the user and the signer are different in a blind signature. (1) $\mathcal{F}^{R_U}_{\text{SVZK}}$ is the "single-verifier zero-knowledge functionality for the relation R_U" where the user will be the prover and, (2) $\mathcal{F}^{R_S}_{\text{SPZK}}$ is the "single-prover zero-knowledge functionality for the relation R_S" where the signer will be the prover. They differ from the multi-session ZK ideal functionality $\mathcal{F}_{\text{MZK}}$ (e.g., see $\hat{\mathcal{F}}_{\text{ZK}}$ in figure 7, page 49, in [11]) in the following manner: $\mathcal{F}_{\text{SVZK}}$ assumes that there is only a single verifier that many provers wish to prove to it a certain type of statements; on the other hand, $\mathcal{F}_{\text{SPZK}}$ assumes that only a single prover exists that wishes to convince many verifiers regarding a certain type of statement. Our setting is different from previous UC-formulations of ZK where multiple provers wish to convince multiple verifiers at the same time; while we could use such stronger primitives in our design, recall that we are interested in the simplest possible primitives that can instantiate our methodology as these highlight minimum sufficient requirements for blind signature design in the UC setting.

3.1 Construction in the $(\mathcal{F}_{\text{CRS}}, \mathcal{F}_{\text{SVZK}}, \mathcal{F}_{\text{SPZK}})$-Hybrid World

In this section we describe our blind signature construction in the hybrid world. In Figure 6, we describe a UC blind signature protocol in the $(\mathcal{F}_{\text{CRS}}, \mathcal{F}^{R_U}_{\text{SVZK}}, \mathcal{F}^{R_S}_{\text{SPZK}})$-hybrid world that is based on an equivocal lite blind signature protocol. The relations parameterized with the ZK functionalities are $R_U = \{((crs, vk, \mathbf{u}), (m, \rho_1)) \mid \mathbf{u} = \mathsf{lbs}_1(crs, vk, m; \rho_1)\}$ and $R_S = \{((crs, vk, \mathbf{u}, \mathbf{s}), (sk, \rho_2)) \mid \mathbf{s} = \mathsf{lbs}_2(crs,\ vk, \mathbf{u}, sk; \rho_2) \wedge (vk, sk) \in \mathsf{KeyPair}\}$. We prove the following theorem:

Theorem 3. *Given a signature generation protocol that is an equivocal lite blind signature, the protocol $\pi_{\Sigma(\text{BSIG})}$ in Figure 6 securely realizes $\mathcal{F}_{\text{BSIG}}$ in the $(\mathcal{F}_{\text{CRS}}, \mathcal{F}^{R_U}_{\text{SVZK}},\ \mathcal{F}^{R_S}_{\text{SPZK}})$-hybrid model.*

3.2 Realizing $\mathcal{F}_{\text{SVZK}}$ and $\mathcal{F}_{\text{SPZK}}$

In this subsection we focus on the requirements for the UC-realization of the two ZK functionalities $\mathcal{F}_{\text{SVZK}}$ and $\mathcal{F}_{\text{SPZK}}$. We note that they can be instantiated generically based on non-interactive zero-knowledge as in [11] or [20]. Nevertheless, by focusing on the exact requirements needed

Protocol $\pi_{\Sigma(\mathrm{BSIG})}$ in the $(\mathcal{F}_{\mathrm{CRS}}, \mathcal{F}_{\mathrm{SVZK}}^{R_U}, \mathcal{F}_{\mathrm{SPZK}}^{R_S})$-Hybrid Model

CRS generation: $crs \leftarrow \mathtt{CRSgen}(1^\lambda)$ where λ is the security parameter.

Key generation: When party S is invoked with input $(\textsc{KeyGen}, sid)$ by $\mathcal{Z}$, it verifies that $sid = (S, sid')$ for some sid'; If not, it ignores the input; Otherwise, it runs $(vk, sk) \leftarrow \mathsf{gen}(crs)$, lets the verification algorithm $\mathsf{ver} \stackrel{\mathsf{def}}{=} \mathtt{verify}(crs, vk, \cdot, \cdot)$, and sends output $(\textsc{VerificationAlg}, sid, \mathsf{ver})$ to $\mathcal{Z}$.

Signature generation: On input $(\textsc{Sign}, sid, m, \mathsf{ver}')$ by $\mathcal{Z}$ where $sid = (S, sid')$, party U obtains vk' by parsing ver', selects random ρ_1, computes $\mathbf{u} \leftarrow \mathsf{lbs}_1(crs, vk', m; \rho_1)$ and sends $(\textsc{Prove}\mathrm{SVZK}, sid, U, (crs, vk', \mathbf{u}), (m, \rho_1))$ to $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$.

Upon receiving $(\textsc{Verified}\mathrm{SVZK}, sid, U, (crs', vk', \mathbf{u}))$ from $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$, party S verifies $crs' = crs$ and $vk' = vk$. If not, then party S outputs $(\textsc{SignStatus}, sid, U, \mathsf{incomplete})$ to $\mathcal{Z}$. Else party S selects random ρ_2 and computes $\mathbf{s} \leftarrow \mathsf{lbs}_2(crs, vk, \mathbf{u}, sk; \rho_2)$ and sends $(\textsc{Prove}\mathrm{SPZK}, sid, U, (crs, vk, \mathbf{u}, \mathbf{s}), (sk, \rho_2))$ to $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$, and outputs $(\textsc{SignStatus}, sid, U, \mathsf{complete})$ to $\mathcal{Z}$.

Upon receiving $(\textsc{Verified}\mathrm{SVZK}, sid, U, \perp)$ from $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$, party S outputs $(\textsc{SignStatus}, sid, U, \mathsf{incomplete})$ to $\mathcal{Z}$.

Upon receiving $(\textsc{Verified}\mathrm{SPZK}, sid, U, (crs', vk'', \mathbf{u}', \mathbf{s}))$ from $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$, party U verifies that $crs' = crs$ and $vk'' = vk'$ and $\mathbf{u}' = \mathbf{u}$. If not, then party U outputs $(\textsc{Signature}, sid, \perp)$ to $\mathcal{Z}$. Else, party U selects random ρ_3 and computes $\sigma \leftarrow \mathsf{lbs}_3(crs, vk', m, \rho_1, \mathbf{u}, \mathbf{s}; \rho_3)$, and outputs $(\textsc{Signature}, sid, \sigma)$ to $\mathcal{Z}$.

Upon receiving $(\textsc{Verified}\mathrm{SPZK}, sid, U, \perp)$ from $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$, party U outputs $(\textsc{Signature}, sid, \perp)$ to $\mathcal{Z}$.

Signature verification: When party V is invoked with input $(\textsc{Verify}, sid, m, \sigma, \mathsf{ver}')$ by $\mathcal{Z}$ where $sid = (S, sid')$, it outputs $(\textsc{Verified}, sid, \mathsf{ver}'(m, \sigma))$ to $\mathcal{Z}$.

Fig. 6. Blind signature protocol $\pi_{\Sigma(\mathrm{BSIG})}$ in the $(\mathcal{F}_{\mathrm{CRS}}, \mathcal{F}_{\mathrm{SVZK}}^{R_U}, \mathcal{F}_{\mathrm{SPZK}}^{R_S})$-hybrid model based on a lite-blind signature scheme $\langle \mathtt{CRSgen}, \mathsf{gen}, \mathsf{lbs}_1, \mathsf{lbs}_2, \mathsf{lbs}_3, \mathtt{verify} \rangle$. Here functionalities $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$ and $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$ are parameterized with relations $R_U = \{((crs, vk, \mathbf{u}), (m, \rho_1)) \mid \mathbf{u} = \mathsf{lbs}_1(crs, m; \rho_1)\}$ and $R_S = \{((crs, vk, \mathbf{u}, \mathbf{s}), (sk, \rho_2)) \mid \mathbf{s} = \mathsf{lbs}_2(crs, vk, \mathbf{u}, sk; \rho_2) \wedge (vk, sk) \in \mathsf{KeyPair}\}$, respectively.

for the blind signature setting we manage to get more simplified concrete constructions; note that we will opt for minimizing the overall communication length as opposed to round complexity.

Realizing $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$. The functionality $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$ will be realized against adaptive adversaries. We proceed as follows: first given $(x, w) \in R_U$, we will have the prover commit the witness w into a value C; in order to obtain an efficient construction, we employ the mixed commitment primitive of [16, 29]; a critical observation in our setting is that due to the fact that we have a single verifier (the signer) it is possible to maintain a constant size common reference string (independent in the number of committers). In contrast we note that in [16, 29] it was necessary to rely on a linear length common reference string in the number of protocol participants; this was to suppress man-in-the-middle attacks that could be launched within their session scope (while such attacks are not possible within our session scope). Our construction also employs a non-erasure Sigma protocol based on which we show the consistency of the witness between the commitment C and the statement x by performing a proof of language membership; finally to defend against a dishonest verifier, our Sigma protocol will have to be strengthened so that it can be simulated without knowing the witness; this e.g., can be based on Damgård's trick [14].

Based on the above we obtain an efficient number-theoretic instantiation of the functionality that is secure under the Decisional Composite Residuosity assumption of Paillier [33]. The underlying mixed-commitment is based on Damgård-Jurik encryption [15]; it could be also based on other encryption schemes as well.

Realizing $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$. Regarding $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$ we find that, rather surprisingly, our task for attaining an adaptive secure UC blind signature is simpler since security against a static adversary suffices. The reason is that in the UC blind signature security proof, the simulator knows the signing secret which means the witness for $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$ is known by the simulator, and thus no equivocation of dishonestly simulated transcripts is ever necessary! This behavior was explored by the authors in the context of zero-knowledge in [26]; in the framework of that paper, we can say a blind signature protocol falls into the class of protocols where a leaking version of $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$ is sufficient for security and thus $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$ need be realized only against static adversaries.

Similarly to the realization of $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$, for $(x, w) \in R_S$, we have the prover commit to the witness w into the value C, but here we only need employ an extractable commitment considering we only need to realize $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$ against static adversaries; then we develop a Sigma protocol to show the consistency between the commitment C and the statement x by performing a proof of language membership; the first two steps together can be viewed as an Ω-protocol in [18]; further we need to wrap up such Ω-protocol by applying e.g., Damgård's trick to defend against dishonest verifiers.

3.3 Concrete Construction

In this section, we demonstrate how it is possible to derive an efficient UC blind signature instantiation based on Theorem 3 and the realization of its hybrid world with the related ZK-functionalities. Note that we opt for minimizing the overall communication complexity as opposed to round complexity. We need three ingredients: (1) an equivocal lite blind signature scheme, (2) a UC-realization of the ideal functionality $\mathcal{F}_{\mathrm{SVZK}}^{R_U}$, (3) a UC-realization of the ideal functionality $\mathcal{F}_{\mathrm{SPZK}}^{R_S}$. Regarding (1) we will employ the equivocal lite blind signature scheme of Figure 4. Regarding the two ZK functionalities we will follow the design strategy outlined in the previous subsection. Recall that in Figure 6, $R_U = \{((crs, vk, \mathbf{u}), (m, \rho_1)) \mid \mathbf{u} = \mathsf{lbs}_1(crs, m; \rho_1)\}$ and $R_S = \{((crs, vk, \mathbf{u}, \mathbf{s}), (sk, \rho_2)) \mid \mathbf{s} = \mathsf{lbs}_2(crs, vk, \mathbf{u}, sk; \rho_2) \wedge (vk, sk) \in \mathsf{KeyPair}\}$. Instantiating these relations for the protocol of Figure 4 we have that $R_U = \{((crs, X, W), (m, t, s)) \mid W = g_1^{mt} u_1^t v_1^{st}\}$ and $R_S = \{((crs, X, W, Y, l, r), x) \mid Y = (W v_1^l)^{\frac{1}{x+r}} \wedge X = g_2^x\}$. Please refer to the full version for all the details [25] as well as the full description of the blind signature protocol.

Finally, we can obtain the corollary below:

Corollary 1. *Under the DCR assumption, the DLOG assumption, and the 2SDH assumption, and assuming existence of collision resistant hash function, there exists a blind signature protocol that securely realizes $\mathcal{F}_{\mathrm{BSIG}}$ in the $\mathcal{F}_{\mathrm{CRS}}$-hybrid model.*

Acknowledgements. We thank Jesper Nielsen for helpful clarifications on some of his zero-knowledge and commitment protocols and models. We also thank the anonymous referees for their constructive comments.

References

1. M. Abe. A secure three-move blind signature scheme for polynomially many signatures. In B. Pfitzmann, editor, *EUROCRYPT 2001*, pages 136–151. Springer, 2001.
2. M. Abe and M. Ohkubo. Provably secure fair blind signatures with tight revocation. In C. Boyd, editor, *ASIACRYPT 2001*, pages 583–602. Springer, 2001.

3. M. Abe and T. Okamoto. Provably secure partially blind signatures. In M. Bellare, editor, *CRYPTO 2000*, pages 271–286. Springer, 2000.
4. D. Beaver. Adaptive zero knowledge and computational equivocation (extended abstract). In *STOC 1996*, pages 629–638. ACM, 1996.
5. M. Bellare, C. Namprempre, D. Pointcheval, and M. Semanko. The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. *J. Cryptology*, 16(3):185–215, 2003. The preliminary version entitled as "The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme" appeared in Financial Cryptography 2001, Springer-Verlag(LNCS 2339).
6. A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Y. Desmedt, editor, *PKC 2003*, pages 31–46. Springer, 2003.
7. J. Camenisch, M. Koprowski, and B. Warinschi. Efficient blind signatures without random oracles. In C. Blundo and S. Cimato, editors, *SCN 2004*, pages 134–148. Springer, 2004.
8. J. Camenisch, G. Neven, and A. Shelat. Simulatable adaptive oblivious transfer. In M. Naor, editor, *EUROCRYPT 2007*, pages 573–590. Springer, 2007.
9. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS 2001*, pages 136–145. IEEE, 2001.
10. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Cryptology ePrint Archive, Report 2000/067*, December 2005. Latest version at `http://eprint.iacr.org/2000/067/`.
11. R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *STOC 2002*, pages 494–503. ACM, 2002. Full version at `http://www.cs.biu.ac.il/~lindell/PAPERS/uc-comp.ps`.
12. D. Chaum. Blind signatures for untraceable payments. In D. Chaum, R. L. Rivest, and A. T. Sherman, editors, *CRYPTO 1982*, pages 199–203. Plemum Press, 1982.
13. I. Damgård. Payment systems and credential mechanisms with provable security against abuse by individuals. In S. Goldwasser, editor, *CRYPTO 1988*, pages 328–335. Springer, 1988.
14. I. Damgård. Efficient concurrent zero-knowledge in the auxiliary string model. In *EUROCRYPT 2000*, pages 418–430. Springer, 2000.
15. I. Damgård and M. Jurik. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In K. Kim, editor, *PKC 2001*, pages 119–136. Springer, 2001.
16. I. Damgård and J. B. Nielsen. Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In M. Yung, editor, *CRYPTO 2002*, pages 581–596. Springer, 2002.
17. M. Fischlin. Round-optimal composable blind signatures in the common reference string model. In *CRYPTO 2006*.
18. J. A. Garay, P. D. MacKenzie, and K. Yang. Strengthening zero-knowledge protocols using signatures. *J. Cryptology*, 19(2):169–209, 2006. An extended abstract appeared in Eurocrypt 2003, Springer-Verlag (LNCS 2656), pages 177-194, 2003.
19. S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
20. J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. In S. Vaudenay, editor, *EUROCRYPT 2006*, pages 339–358. Springer, 2006.
21. C. Hazay, J. Katz, C.-Y. Koo, and Y. Lindell. Concurrently-secure blind signatures without random oracles or setup assumptions. In S. P. Vadhan, editor, *TCC 2007*, pages 323–341. Springer, 2007.

22. O. Horvitz and J. Katz. Universally-composable two-party computation in two rounds. In A. Menezes, editor, *CRYPTO 2007*, pages 111–129. Springer, 2007.
23. A. Juels, M. Luby, and R. Ostrovsky. Security of blind digital signatures (extended abstract). In B. S. K. Jr., editor, *CRYPTO 1997*, pages 150–164. Springer, 1997.
24. A. Kiayias and H.-S. Zhou. Concurrent blind signatures without random oracles. In R. D. Prisco and M. Yung, editors, *SCN 2006*, pages 49–62. Springer, 2006.
25. A. Kiayias and H.-S. Zhou. Equivocal blind signatures and adaptive UC-security. In *Cryptology ePrint Archive: Report 2007/132*, 2007. Full version.
26. A. Kiayias and H.-S. Zhou. Trading static for adaptive security in universally composable zero-knowledge. In L. Arge, C. Cachin, T. Jurdzinski, and A. Tarlecki, editors, *ICALP 2007*, pages 316–327. Springer, 2007.
27. K. Kim. Lessons from Internet voting during 2002 FIFA WorldCup Korea/Japan(TM). In *DIMACS Workshop on Electronic Voting – Theory and Practice*, 2004.
28. Y. Lindell. Bounded-concurrent secure two-party computation without setup assumptions. In *STOC 2003*, pages 683–692. ACM, 2003. Full version at `http://www.cs.biu.ac.il/~lindell/PAPERS/conc2party-upper.ps`.
29. J. B. Nielsen. On protocol security in the cryptographic model. *Dissertation Series DS-03-8, BRICS*, 2003. `http://www.brics.dk/DS/03/8/BRICS-DS-03-8.pdf`.
30. T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In E. F. Brickell, editor, *CRYPTO 1992*, pages 31–53. Springer, 1992.
31. T. Okamoto. Efficient blind and partially blind signatures without random oracles. In S. Halevi and T. Rabin, editors, *TCC 2006*, pages 80–99. Springer, 2006.
32. T. Okamoto and K. Ohta. Divertible zero knowledge interactive proofs and commutative random self-reducibility. In J.-J. Quisquater and J. Vandewalle, editors, *EUROCRYPT 1989*, pages 134–148. Springer, 1989.
33. P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In J. Stern, editor, *EUROCRYPT 1999*, pages 223–238. Springer, 1999.
34. D. Pointcheval. Strengthened security for blind signatures. In K. Nyberg, editor, *EUROCRYPT 1998*, pages 391–405. Springer, 1998.
35. D. Pointcheval and J. Stern. Provably secure blind signature schemes. In K. Kim and T. Matsumoto, editors, *ASIACRYPT 1996*, pages 252–265. Springer, 1996.
36. D. Pointcheval and J. Stern. New blind signatures equivalent to factorization (extended abstract). In *CCS 1997*, pages 92–99. ACM, 1997.
37. D. Pointcheval and J. Stern. Security arguments for digital signatures and blind signatures. *J. Cryptology*, 13(3):361–396, 2000.
38. M. Prabhakaran and A. Sahai. Relaxing environmental security: Monitored functionalities and client-server computation. In J. Kilian, editor, *TCC 2005*, pages 104–127. Springer, 2005.