Round-Optimal Composable Blind Signatures in the Common Reference String Model

Marc Fischlin*

Darmstadt University of Technology, Germany
marc.fischlin@gmail.com www.fischlin.de

Abstract We build concurrently executable blind signatures schemes in the common reference string model, based on general complexity assumptions, and with optimal round complexity. Namely, each interactive signature generation requires the requesting user and the issuing bank to transmit only one message each. We also put forward the definition of universally composable blind signature schemes, and show how to extend our concurrently executable blind signature protocol to derive such universally composable schemes in the common reference string model under general assumptions. While this protocol then guarantees very strong security properties when executed within larger protocols, it still supports signature generation in two moves.

1 Introduction

Blind signatures, introduced by Chaum [8], allow a bank to interactively issue signatures to users such that the signed message is hidden from the bank (blindness) while at the same time users cannot output more signatures than interactions with the bank took place (unforgeability). Numerous blind signature schemes have been proposed, mostly under specific number-theoretic assumptions, some relying also on the random oracle model [26,1,3,4] and some forgoing random oracles [9,18,24]. Only the work by Juels et al. [17] addresses the construction of blind signatures under general assumptions explicitly, and deploys general two-party protocols and oblivious transfer based on trapdoor permutations.

Interestingly, almost all of the aforementioned blind signature schemes require three or more moves (most of them even in the random oracle model) and concurrent executions of the signature generation protocol are often a concern (cf. [25,26,24]). For instance, making the solution by Juels et al. [17] concurrently secure would require a high round complexity. This follows from results by Lindell [19,20] showing, among others, that in the plain model the number of rounds in blind signature schemes with black-box security proofs is bounded from below by the number of concurrent executions.

A notable exception to the problems with interleaving executions are schemes with an optimal two-move signature generation protocol, solving the concurrency

* This work was supported by the Emmy Noether Program Fi 940/2-1 of the German Research Foundation (DFG).

problem immediately. This includes Chaum's original RSA-based blind signature protocol and the pairing-based discrete-log version thereof [4]. Unfortunately, the security proofs [3,4] for these schemes need the random oracle model —by which they can bypass Lindell's lower bound for the plain model— and rely on the so-called one-more RSA or one-more discrete-log assumptions, which are less investigated than the corresponding standard problems.

Here we show that one can build secure blind signature schemes with a two-move signature generation protocol under general assumptions (namely, trapdoor permutations). Our scheme does not rely on random oracles, yet to bridge the lower bound on the number of rounds we work in the common reference string model. We note that instead of falling back on general multi-party paradigms as in [17] and inheriting the round complexity of the underlying oblivious transfer protocols for general assumptions we give a dedicated solution to the blind signature problem.

Construction Idea. The basic (and simplified) construction idea of our blind signature scheme is as follows. The user commits to the message m and sends this commitment U to the bank. The bank signs U with a signature scheme and sends the signature B back to user. The user finally derives the blind signature for m by computing a commitment[1] C of $U||B$ and proving with a non-interactive zero-knowledge proof π (based on the common reference string) that this commitment C contains a valid signature B for U and that U itself is a commitment of the message m. The blind signature to m is given by the commitment C and the proof π.

Using standard non-interactive zero-knowledge (NIZK) proofs our protocol above provides a weaker unforgeability notion than postulated in [26,17]. That is, a malicious user can potentially generate several signatures from a single interaction with the bank by generating multiple proofs $\pi_1, \pi_2, \ldots$ for the same commitment C. All these pairs $C||\pi_i$ would be valid blind signatures for the same message m, while the standard unforgeability definition asks that a malicious user cannot create more signatures than interactions happened. In the full version [14] we show how to thwart such attacks by switching to so-called *unique* non-interactive zero-knowledge proofs, recently introduced by Lepinski et al. [21]. In this version here we only treat the weaker notion where the malicious user cannot create a signature for a new message.

Universal Composition. As explained, our two-move signature generation protocol overcomes the concurrency problem effortlessly. More general a slight variation of our scheme yields a secure blind signature scheme in the universal composition (UC) framework [5]. Secure schemes in this UC framework enable interdependent executions with other protocols while preserving the main security characteristics. Our modified scheme now requires the same assumptions as before as well as a simulation-sound NIZK proof of knowledge, which can also be derived from trapdoor permutations [28,11].

[1] For the security proof we require that the commitment C is actually done through an encryption scheme.

Towards proving the universal composability result we first formalize an ideal functionality $\mathcal{F}_{\text{BlSig}}$, prescinding the basic requirements of blind signatures such as completeness, unforgeability and blindness. Since such UC blind signatures can be used to build UC commitments it follows from an impossibility result in [7] that UC blind signatures cannot be realized through two-party protocols in the plain model. But, as our solution shows, augmenting the model by common reference strings one can build UC blind signatures (against non-adaptive corruptions) under general assumptions. Compared to general feasibility results in the UC framework [10], showing that essentially any functionality can be securely realized in the common random string, our solution still needs only two-moves to generate signatures.

2 Blind Signatures in the Common Reference String Model

In this section we recall the security definition of blind signatures and present our two-move solution and prove its security.

2.1 Blind Signatures and Their Security

For the interactive signature generation protocol of a blind signature scheme we introduce the following notation. For two interactive algorithms $\mathcal{X}$, $\mathcal{Y}$ we denote by $(a, b) \leftarrow \langle \mathcal{X}(x), \mathcal{Y}(y) \rangle$ the joint execution of $\mathcal{X}$ for input x and $\mathcal{Y}$ for input y, where $\mathcal{X}$'s private output at the end of the execution equals a and $\mathcal{Y}$'s private output is b. For an algorithm $\mathcal{Y}$ we write $\mathcal{Y}^{\langle \mathcal{X}(x), \cdot \rangle^{\infty}}$ if $\mathcal{Y}$ can invoke an unbounded number of executions of the interactive protocol with $\mathcal{X}$ in arbitrarily interleaved order. Accordingly, $\mathcal{X}^{\langle \cdot, \mathcal{Y}(y_0) \rangle^1, \langle \cdot, \mathcal{Y}(y_1) \rangle^1}$ can invoke arbitrarily interleaved executions with $\mathcal{Y}(y_0)$ and $\mathcal{Y}(y_1)$ but interact with each algorithm only once.

Definition 1 (Blind Signature Scheme). *A blind signature scheme (in the common reference string model) consists of a tuple of efficient algorithms* $\mathsf{BS} = (\mathcal{C}_{BS}, \mathsf{KG}_{BS}, \langle \mathcal{B}, \mathcal{U} \rangle, \mathsf{Vf}_{BS})$ *where*

CRS generation. $\mathcal{C}_{BS}$ *on input* 1^n *outputs a common reference (or random) string* crs_{BS}.

Key Generation. $\mathsf{KG}_{BS}(crs_{BS})$ *generates a key pair* (sk_{BS}, pk_{BS}).

Signature Issuing. *The joint execution of algorithm* $\mathcal{B}(crs_{BS}, sk_{BS})$ *and algorithm* $\mathcal{U}(crs_{BS}, pk_{BS}, m)$ *generates an output* S *of the user,* $(\perp, S) \leftarrow \langle \mathcal{B}(crs_{BS}, sk_{BS}), \mathcal{U}(crs_{BS}, pk_{BS}, m) \rangle$.

Verification. $\mathsf{Vf}_{BS}(crs_{BS}, pk_{BS}, m, S)$ *outputs a bit.*

It is assumed that the scheme is complete, *i.e., for any* $crs_{BS} \leftarrow \mathcal{C}_{BS}(1^n)$, *any* $(sk_{BS}, pk_{BS}) \leftarrow \mathsf{KG}_{BS}(crs_{BS})$, *any message* $m \in \{0,1\}^n$ *and any* S *output by* $\mathcal{U}$ *in the joint execution of* $\mathcal{B}(crs_{BS}, sk_{BS})$ *and* $\mathcal{U}(crs_{BS}, pk_{BS}, m)$ *we have* $\mathsf{Vf}_{BS}(crs_{BS}, pk_{BS}, m, S) = 1$.

Security of blind signatures consists of two requirements [26,17]. Unforgeability says that it should be infeasible for a malicious user $\mathcal{U}^*$ to generate $k+1$ valid signatures given that k interactions with the honest bank took place (where the adversary adaptively decides on the number k of interactions). Blindness says that it should be infeasible for a malicious bank $\mathcal{B}^*$ to determine the order in which two messages m_0, m_1 have been signed in executions with an honest user. This should hold, of course, as long as the interactive signature generation produces two valid signatures and the bank $\mathcal{B}^*$ for example does not abort deliberately in one of the two executions.

Definition 2 (Secure Blind Signature Scheme). *A blind signature scheme* $\mathsf{BS} = (\mathcal{C}_{BS}, \mathsf{KG}_{BS}, \langle \mathcal{B}, \mathcal{U} \rangle, \mathsf{Vf}_{BS})$ *in the common reference string model is called secure if the following holds:*

Unforgeability. *For any efficient algorithm* $\mathcal{U}^*$ *the probability that experiment* $\mathsf{Forge}^{\mathsf{BS}}_{\mathcal{U}^*}(n)$ *evaluates to* 1 *is negligible (as a function of* n*) where*

Experiment $\mathsf{Forge}^{\mathsf{BS}}_{\mathcal{U}^*}(n)$
$crs_{BS} \leftarrow \mathcal{C}_{BS}(1^n)$
$(sk_{BS}, pk_{BS}) \leftarrow \mathsf{KG}_{BS}(crs_{BS})$
$((m_1, S_1), \ldots, (m_{k+1}, S_{k+1})) \leftarrow \mathcal{U}^{*\langle \mathcal{B}(crs_{BS}, sk_{BS}), \cdot \rangle^{\infty}}(crs_{BS}, pk_{BS})$
Return 1 *iff*
$m_i \neq m_j$ *for* $1 \leq i < j \leq k+1$, *and*
$\mathsf{Vf}_{BS}(crs_{BS}, pk_{BS}, m_i, S_i) = 1$ *for all* $i = 1, 2, \ldots, k+1$, *and*
at most k *interactions with* $\langle \mathcal{B}(crs_{BS}, sk_{BS}), \cdot \rangle^{\infty}$ *were initiated.*

Blindness. *For any efficient algorithm* $\mathcal{B}^*$ *(working in modes* find, issue *and* guess*) the probability that the following experiment* $\mathsf{Blind}^{\mathsf{BS}}_{\mathcal{B}^*}(n)$ *evaluates to* 1 *is negligibly close to* $1/2$, *where*

Experiment $\mathsf{Blind}^{\mathsf{BS}}_{\mathcal{B}^*}(n)$
$crs_{BS} \leftarrow \mathcal{C}_{BS}(1^n)$
$(pk_{BS}, m_0, m_1, \beta_{\mathsf{find}}) \leftarrow \mathcal{B}^*(\mathsf{find}, crs_{BS})$
$b \leftarrow \{0, 1\}$
$\beta_{\mathsf{issue}} \leftarrow \mathcal{B}^{*\langle \cdot, \mathcal{U}(crs_{BS}, pk_{BS}, m_b) \rangle^1, \langle \cdot, \mathcal{U}(crs_{BS}, pk_{BS}, m_{1-b}) \rangle^1}(\mathsf{issue}, \beta_{\mathsf{find}})$
and let S_b, S_{1-b} *denote the (possibly undefined) local outputs of* $\mathcal{U}(crs_{BS}, pk_{BS}, m_b)$ *resp.* $\mathcal{U}(crs_{BS}, pk_{BS}, m_{1-b})$.
$b^* \leftarrow \mathcal{B}^*(\mathsf{guess}, S_0, S_1, \beta_{\mathsf{issue}})$
Return a random bit if $S_0 = \perp$ *or* $S_1 = \perp$, *else return* 1 *iff* $b = b^*$.

As pointed out in the introduction, in a stronger unforgeability notion the malicious user is already deemed to be successful if it outputs $k+1$ distinct message-signature pairs (m_i, S_i) after k interactions with the bank. In this case the user may also try to find another valid signature to a previously signed message m. Our scheme here satisfies the unforgeability notion in Definition 2 but can be turned into one achieving the higher level (see the full version [14]).

In a *partially* blind signature scheme [2] the bank and the user first agree on some information info which is attached to the blind signature. There, unforgeability demands that a malicious user cannot find $k+1$ distinct but valid tuples

$(\mathsf{info}_i, m_i, S_i)$. The definition of blindness then allows a malicious bank to determine info together with the messages m_0, m_1 such that the bank still cannot decide the order in which the users execute the issuing protocol for info, m_0 and info, m_1, respectively. Jumping ahead we note that we can easily turn our blind signature scheme into a partially blind one.

2.2 Construction

The high-level idea of our blind signature protocol is as follows. To obtain a blind signature from the bank the user commits to the message m and sends this commitment U to the bank. The bank signs the commitment with a regular signature scheme and returns the signature B to the user. The user derives the blind signature for m by computing another commitment C of the signature B together with U, and a non-interactive zero-knowledge proof π showing the validity of C.

To prevent trivial "size-measuring" attacks of a malicious bank we assume that the signature scheme is *length-invariant*, i.e., that public keys pk_{Sig} as well as signatures for security parameter n are all of the same length $s(n)$. We note that this can always be achieved by standard padding techniques, and length-invariant signature schemes exist if one-way functions exist [23,27].

We furthermore assume that the commitment scheme $(\mathcal{C}_{\mathsf{Com}}, \mathsf{Com})$ in the common random string model, given by algorithms $\mathcal{C}_{\mathsf{Com}}$ generating the string crs_{Com} and algorithm $\mathsf{Com}(crs_{\mathsf{Com}}, \cdot, \cdot) : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{c(n)}$ mapping strings from $\{0,1\}^n$ with n-bit randomness to commitments, is length-invariant, too. That is, the reference strings as well as commitments are all of length $c(n)$ for parameter n. We also need that the commitment scheme is statistically binding. Such commitment schemes can also be derived for example from one-way functions [16,22].

In order to turn the above idea into a provably secure scheme the proof π needs to allow extraction of U and B encapsulated in C. We accomplish this by using an IND-CPA secure encryption scheme $(\mathsf{KG}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ to "commit" to $U||B$ in C, where the public key pk_{Enc} of the encryption algorithm becomes part of the common reference string.[2] We presume that the encryption scheme is also length-invariant and that public keys pk_{Enc} and ciphertexts $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, U||B; v)$ for $U||B \in \{0,1\}^{c(n)+s(n)}$ and randomness $v \in \{0,1\}^n$ are all of length $e(n)$. This is again without loss of generality.

Finally, the non-interactive zero-knowledge (NIZK) proof $(\mathcal{C}_{\mathrm{ZK}}, \mathcal{P}, \mathcal{V})$ with the common reference string generator $\mathcal{C}_{\mathrm{ZK}}$, the prover $\mathcal{P}$ and the verifier $\mathcal{V}$ first of all obeys the two basic properties completeness (the verifier $\mathcal{V}$ accepts all honestly generated proofs of the prover $\mathcal{P}$) and soundness (no malicious prover can make the verifier accepts proofs for invalid statements). Furthermore, the system should be multiple zero-knowledge in the sense of [15], i.e., one

[2] We can also assume that we have dense public-key schemes [13] and that the common *random* string contains such a public key.

can prove several statements in zero-knowledge for the same common reference string $crs_{\mathrm{ZK}} \leftarrow \mathcal{C}_{\mathrm{ZK}}(1^n)$. Such proofs exist if trapdoor permutations exist [15].

The underlying relation of the NIZK proof is described by (a sequence of) circuits $\mathsf{C}_n^{\mathrm{BS}}$ indexed by a parameter n. Circuit $\mathsf{C}_n^{\mathrm{BS}}$ takes as input a statement $x = C||pk_{\mathsf{Enc}}||crs_{\mathsf{Com}}||pk_{\mathsf{Sig}}||m$ of bit length $\chi(n) = c(n) + 2e(n) + s(n) + n$ and a witness $w = u||v||B$ of length $\omega(n) = 2n + s(n)$, and returns an output bit which is determined as follows. The circuit is built from the descriptions of algorithms $\mathsf{Com}, \mathsf{Enc}, \mathsf{Vf}_{\mathsf{Sig}}$ and checks for the signature's verification algorithm that $\mathsf{Vf}_{\mathsf{Sig}}(pk_{\mathsf{Sig}}, \mathsf{Com}(crs_{\mathsf{Com}}, m; u), B) = 1$ and that the value C equals the ciphertext $\mathsf{Enc}(pk_{\mathsf{Enc}}, \mathsf{Com}(crs_{\mathsf{Com}}, m; u)||B; v)$. If and only if both tests evaluate to true then the circuit outputs 1. The corresponding relation (parameterized by n) is given by $R_n^{\mathrm{BS}} = \left\{(x, w) \in \{0,1\}^{\chi(n)} \times \{0,1\}^{\omega(n)} \;\middle|\; \mathsf{C}_n^{\mathrm{BS}}(x, w) = 1\right\}$.

Construction 1 (Blind Signature Scheme). *Let* $(\mathsf{KG}_{\mathsf{Sig}}, \mathsf{Sig}, \mathsf{Vf}_{\mathsf{Sig}})$ *be a signature scheme,* $(\mathsf{KG}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme,* $(\mathcal{C}_{\mathsf{Com}}, \mathsf{Com})$ *be a commitment scheme, and let* $(\mathcal{C}_{ZK}, \mathcal{P}, \mathcal{V})$ *be a non-interactive zero-knowledge proof system for* R^{BS}*. Define the following four procedures:*

CRS Generation. *Algorithm* $\mathcal{C}_{BS}(1^n)$ *generates* $crs_{ZK} \leftarrow \mathcal{C}_{ZK}(1^n)$*,* $crs_{\mathsf{Com}} \leftarrow \mathcal{C}_{\mathsf{Com}}(1^n)$ *as well as* $(pk_{\mathsf{Enc}}, sk_{\mathsf{Enc}}) \leftarrow \mathsf{KG}_{\mathsf{Enc}}(1^n)$*. It outputs the string* $crs_{BS} \leftarrow (crs_{ZK}, crs_{\mathsf{Com}}, pk_{\mathsf{Enc}})$*.*

Key Generation. *The bank's key generation algorithm* $\mathsf{KG}_{BS}(crs_{BS})$ *generates a signature key pair* $(pk_{\mathsf{Sig}}, sk_{\mathsf{Sig}}) \leftarrow \mathsf{KG}_{\mathsf{Sig}}(1^n)$*. It returns* $(pk_{BS}, sk_{BS}) \leftarrow (pk_{\mathsf{Sig}}, sk_{\mathsf{Sig}})$*.*

Signature Issue Protocol. *The interactive signature issue protocol is described in Figure 1.*

Signature Verification. *The verification algorithm* $\mathsf{Vf}_{BS}(crs_{BS}, m, S)$ *parses the signature* S *as* $S = C||\pi$ *and returns the output* $\mathcal{V}(crs_{ZK}, x, \pi)$ *for the value* $x = C||pk_{\mathsf{Enc}}||crs_{\mathsf{Com}}||pk_{\mathsf{Sig}}||m$*.*

Theorem 2. *Let* $(\mathsf{KG}_{\mathsf{Sig}}, \mathsf{Sig}, \mathsf{Vf}_{\mathsf{Sig}})$ *be a length-invariant signature scheme which is unforgeable against adaptive chosen-message attacks,* $(\mathsf{KG}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ *be a length-invariant IND-CPA secure encryption scheme, and let* $(\mathcal{C}_{\mathsf{Com}}, \mathsf{Com})$ *be a length-invariant non-interactive commitment scheme in the common random string model which is statistically binding. Let* $(\mathcal{C}_{ZK}, \mathcal{P}, \mathcal{V})$ *be a non-interactive zero-knowledge proof system for* R^{BS}*. Then the scheme defined in Construction 1 is a secure blind signature scheme.*

Proof. We first show unforgeability and then blindness.

Unforgeability. Assume that there exists an adversary $\mathcal{U}^*$ such that with noticeable probability the following holds. On input $crs_{\mathrm{BS}}, pk_{\mathrm{BS}}$ the adversary $\mathcal{U}^*$ manages to output $k+1$ valid signatures $S_i = C_i||\pi_i$ for messages m_i after at most k interactions with the honest bank $\mathcal{B}$ (and where the messages m_i, m_j are pairwise distinct). Given such an adversary we construct a successful adversary $\mathcal{A}$ against the security of the signature scheme $(\mathsf{KG}_{\mathsf{Sig}}, \mathsf{Sig}, \mathsf{Vf}_{\mathsf{Sig}})$.

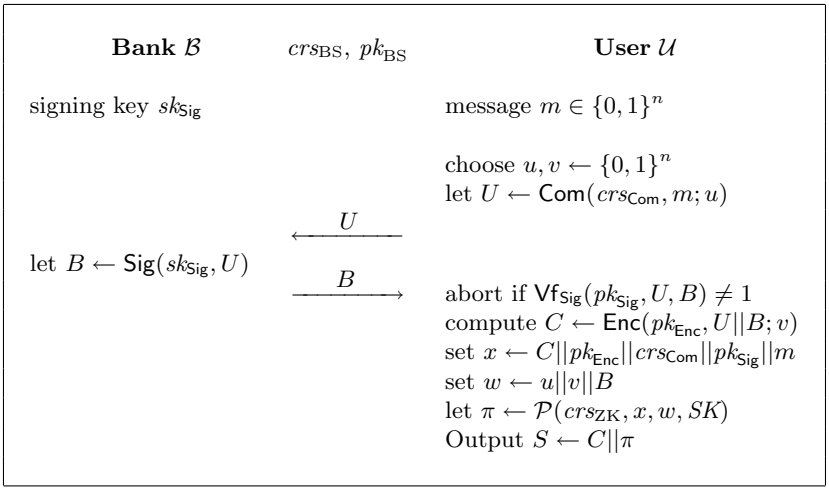

Figure 1. Blind Signature Scheme: Issue Protocol

Adversary $\mathcal{A}$ is given as input a public key pk_{Sig} of the signature scheme and is granted oracle access to a signature oracle $\mathsf{Sig}(sk_{\mathsf{Sig}}, \cdot)$. This adversary first generates $crs_{\mathrm{ZK}} \leftarrow \mathcal{C}_{\mathrm{ZK}}(1^n)$ for the zero-knowledge proof, $crs_{\mathsf{Com}} \leftarrow \mathcal{C}_{\mathsf{Com}}(1^n)$ for the commitments, and $(pk_{\mathsf{Enc}}, sk_{\mathsf{Enc}}) \leftarrow \mathsf{KG}_{\mathsf{Enc}}(1^n)$ and defines $crs_{\mathrm{BS}} \leftarrow (crs_{\mathrm{ZK}}, crs_{\mathsf{Com}}, pk_{\mathsf{Enc}})$. It next invokes a black-box simulation of $\mathcal{U}^*$ for input $(crs_{\mathrm{BS}}, pk_{\mathsf{Sig}})$. Each time the user $\mathcal{U}^*$ initiates the issue protocol with the bank, algorithm $\mathcal{A}$ uses the signature oracle to answer the request U by a signature $B \leftarrow \mathsf{Sig}(sk_{\mathsf{Sig}}, U)$. When the adversary finally outputs the message/signature pairs (m_i, S_i) algorithm $\mathcal{A}$ parses each S_i as $S_i = C_i||\pi_i$ and decrypts each C_i to $U_i||B_i$. Algorithm $\mathcal{A}$ outputs the first one of these pairs (U_i, B_i) for which U_i has not been submitted to its signing oracle (or returns $\perp$ if no such value exists).

For the analysis assume that $\mathcal{U}^*$ succeeds with noticeable probability. Since $k+1$ pairs (m_i, S_i) are valid, all the proofs π_i in $S_i = C_i||\pi_i$ are valid as well. Hence, with overwhelming probability over the choice of crs_{ZK} each C_i is a valid ciphertext of some $U_i||B_i$ under pk_{Enc}, and each U_i commits to the corresponding message m_i and each B_i is a valid signature under pk_{Sig} for U_i. Furthermore, by the statistically-binding property of the commitment scheme all U_i's must be distinct for different m_i's, with overwhelming probability over the choice of crs_{Com}. It follows that $\mathcal{A}$ successfully outputs a valid signature B_i for a previously not signed value U_i with noticeable probability, too, contradicting our initial assumption about the success probability of $\mathcal{U}^*$.

Blindness. To prove blindness we consider an adversarial controlled bank $\mathcal{B}^*$ in experiment $\mathsf{Blind}^{\mathsf{BS}}_{\mathcal{B}^*}(n)$. We gradually transform the way the signatures $S_0 = C_0||\pi_0$ and $S_1 = C_1||\pi_1$ are computed such that, at then end, they are completely independent of bit b.

In the first step we replace all the steps involving the prover by the output of the zero-knowledge simulator $\mathcal{Z}$ (generating a string crs together with some

trapdoor information σ in mode crs and faking proofs π for statements x with the help of σ in mode prove). More precisely, consider the following modified procedures of the blind signature scheme (key generation and verification remain unchanged):

CRS Generation. Algorithm $\mathcal{C}_{\mathrm{BS}}(1^n)$ generates $(crs_{\mathrm{ZK}}, \sigma) \leftarrow \mathcal{Z}(\mathsf{crs}, 1^n)$, and $crs_{\mathsf{Com}} \leftarrow \mathcal{C}_{\mathsf{Com}}(1^n)$ and $(pk_{\mathsf{Enc}}, sk_{\mathsf{Enc}}) \leftarrow \mathsf{KG}_{\mathsf{Enc}}(1^n)$. It outputs the string $crs_{\mathrm{BS}} \leftarrow (crs_{\mathrm{ZK}}, crs_{\mathsf{Com}}, pk_{\mathsf{Enc}})$.

Signature Issue Protocol. For the signature issuing the user now also picks $u, v \leftarrow \{0,1\}^n$, and again sends the commitment $U \leftarrow \mathsf{Com}(crs_{\mathsf{Com}}, m; u)$ to the bank $\mathcal{B}^*$ which replies with some B. The user aborts if verification fails, $\mathsf{Vf}_{\mathsf{Sig}}(pk_{\mathsf{Sig}}, U, B) \neq 1$, and else computes $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, U||B; v)$ as well as $\pi \leftarrow \mathcal{Z}(\mathsf{prove}, \sigma, x)$. Output $S \leftarrow C||\pi$.

Denote the modified scheme by BS'. It follows easily from the zero-knowledge property that experiments $\mathsf{Blind}^{\mathsf{BS}}_{\mathcal{B}^*}(n)$ and $\mathsf{Blind}^{\mathsf{BS}'}_{\mathcal{B}^*}(n)$ return 1 with the same probability (except for a negligible probability).

In the next step we further modify the signature scheme by replacing the commitments U by commitments to all-zero strings. More precisely, we change the signature issue protocol of the blind signature scheme BS' as follows (recall that this modified scheme already uses the zero-knowledge simulator to prepare the signatures):

Signature Issue Protocol. The user picks $u, v \leftarrow \{0,1\}^n$, but now sends $U \leftarrow \mathsf{Com}(crs_{\mathsf{Com}}, 0^n; u)$ to the bank to get a signature B (which is also checked). It then computes again $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, U||B; v)$, $\pi \leftarrow \mathcal{Z}(\mathsf{prove}, \sigma, x)$ and outputs $S \leftarrow C||\pi$.

We call this modified scheme BS''. It is easy to see that, by the secrecy of the commitment scheme, the difference in the output distributions of experiments $\mathsf{Blind}^{\mathsf{BS}'}_{\mathcal{B}^*}(n)$ and $\mathsf{Blind}^{\mathsf{BS}''}_{\mathcal{B}^*}(n)$ is negligible.

Finally, we replace the encryption of U and B by an encryption of the all-zero string, i.e., we modify the signature issuing protocol from BS'' further to obtain:

Signature Issue Protocol. The user selects $u, v \leftarrow \{0,1\}^n$ as before and again computes $U \leftarrow \mathsf{Com}(crs_{\mathsf{Com}}, 0^n; u)$. For the bank's reply B to U it checks validity and this time sets $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, 0^{c(n)+s(n)}; v)$, $\pi \leftarrow \mathcal{Z}(\mathsf{prove}, \sigma, x)$ and outputs $S \leftarrow C||\pi$.

We call this modified scheme by BS'''. By the IND-CPA security of the encryption scheme we conclude that the difference in the output distributions of experiments $\mathsf{Blind}^{\mathsf{BS}''}_{\mathcal{B}^*}(n)$ and $\mathsf{Blind}^{\mathsf{BS}'''}_{\mathcal{B}^*}(n)$ is also negligible.

In experiment $\mathsf{Blind}^{\mathsf{BS}'''}_{\mathcal{B}^*}(n)$ the signatures $S = C||\pi$ are independent of the data U, B in the signature generation protocol. Hence, the adversary $\mathcal{B}^*$ cannot predict b better than with probability $1/2$. Conclusively, the probability of the original experiment $\mathsf{Blind}^{\mathsf{BS}}_{\mathcal{B}^*}(n)$ to return 1 must be negligibly close to $1/2$, proving blindness. □

To get a partially blind signature scheme, where the signer and the user share some public information info to be included in the blind signature, we let the bank simply sign the user's commitment U together with info, i.e., $B \leftarrow \mathsf{Sig}(sk, \mathsf{info}||U)$ and also let the user base the correctness proof π on info. The security proof carries over straightforwardly.

3 Universally Composable Blind Signatures

As mentioned, the blind signature scheme in the previous section allows concurrent executions of the signature generation protocol, i.e., when the protocol is interleaved with itself. More generally, one would like to have a guarantee that such a scheme still supports the basic security properties, even when run as a building block within larger protocols, independently how the execution is intertwined with other steps. Such a guarantee is provided by the universal composition (UC) framework [5].

In the UC framework one defines an idealized version of the primitive in question, capturing the desired security properties in an abstract way and ensuring that the functionality is secure in interdependent settings. Given an appropriate formalization of some functionality $\mathcal{F}$ in the UC framework, one next shows that this functionality can be securely realized by an interactive protocol between the parties (without the trusted interface). Here, securely realizing means that, in any environment (modeled through an algorithm $\mathcal{Z}$) in which the protocol may be run, for this environment executions of the interactive protocol in presence of an adversary $\mathcal{A}$ are indistinguishable from executions in the ideal model with the trustworthy functionality $\mathcal{F}$ and an ideal-model adversary $\mathcal{S}$. A formal introduction to the UC framework is beyond the scope of our paper; we refer to [5] to a comprehensive definition. We remark that we consider non-adaptive adversaries here which corrupt parties at the beginning of executions only.

3.1 Definition

Our definition of an ideal blind signature functionality $\mathcal{F}_{\mathrm{BlSig}}$ follows the one $\mathcal{F}_{\mathrm{Sig}}$ of regular signature schemes given by Canetti [6]. The definition of $\mathcal{F}_{\mathrm{Sig}}$ essentially lets the adversary choose the public verification key and determine the signature value S upon a signing request $(\mathtt{Sign}, sid, m)$. Verification requests for previously generated signatures are always accepted and otherwise the adversary is again allowed to specify whether a tested signature S to a message m is valid or not. See [6] for a discussion of this definition.

The formal description of the blind signature functionality is given in Figure 2. It is partly a verbatim copy of the functionality $\mathcal{F}_{\mathrm{Sig}}$ in [6]. An important difference for blind signatures is that the adversary should not learn the message of honest users and the signatures must not be linkable to the signing request. To ensure this we let the adversary (instead of the bank, analogously to the choice of the public verification key in $\mathcal{F}_{\mathrm{Sig}}$) in $\mathcal{F}_{\mathrm{BlSig}}$ provide (the description of) a stateless, possibly probabilistic algorithm BlSig to the ideal functionality $\mathcal{F}_{\mathrm{BlSig}}$.

This is already done in the key generation step where the adversary chooses the public verification key, such that BlSig is used in all subsequent signature generation runs. Whenever an honest user later requests a signature for a message m this algorithm $\mathsf{BlSig}(m)$ generates a signature S *but without disclosing the message m to the adversary*, enforcing unlinkability of signatures.

If a corrupt user —that is, the adversary on behalf of the corrupt user— requests a signature, however, the ideal functionality does not run BlSig. Instead it asks the adversary about the potential signature S this user could produce from an interaction with the bank. Note that a corrupt user may not output any signature right after an interaction with the bank (or ever), the interaction merely guarantees that this user can generate this signature in principle. Hence, the functionality does not return the adversary's potential signature S to the user.

Finally, for any signature request we inform the bank $\mathcal{B}$ about the request, but without disclosing the actual message m nor the signature. This captures the fact that signature generations require active participation of the bank.

It follows quite easily that one can realize universally composable commitment schemes in the presence of functionality $\mathcal{F}_{\mathrm{BlSig}}$. As a consequence of the hardness of constructing such commitments [7,12] we conclude (for the proof see the full version [14]):

Proposition 1. *Bilateral and terminating (i.e., only two parties are active and honest parties faithfully give output) blind signature schemes securely realizing $\mathcal{F}_{BlSig}$ in the plain model do not exist. Furthermore, blind signature schemes securely realizing $\mathcal{F}_{BlSig}$ in the common reference string model imply key agreement, and imply oblivious transfer in the common random string model. This all holds for non-adaptive corruptions.*

By the general feasibility results of Canetti et al. [5,10] functionality $\mathcal{F}_{\mathrm{BlSig}}$ can be realized in the multi-party setting for honest majorities (in the plain model) and dishonest majorities (in the common random string model). Instead of relying on the general construction in [10] we construct a simpler two-move scheme in the common reference string model directly, based on the scheme in the previous section.

3.2 Construction

The construction in the standard model gives a good starting point for a solution in the UC framework. We augment the scheme by the following steps. First, we let the user in the first round when sending U also encrypt all the data from which the proof is later derived. This ciphertext E should contain the randomness u, v and message m. The encryption scheme itself needs to be IND-CPA secure and we simply use the same scheme as for the computation for $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, U||B; v)$ but with an independent key pair $(pk'_{\mathsf{Enc}}, sk'_{\mathsf{Enc}})$.

In addition to the ciphertext E the user should prove with another NIZK proof that the data is valid and matches the data committed to by U. For this we require *simulation-sound* NIZK proofs [28,11] where a malicious prover is not

Functionality $\mathcal{F}_{\mathsf{BlSig}}$

Key Generation: Upon receiving a value $(\texttt{KeyGen}, sid)$ from a party $\mathcal{B}$, verify that $sid = (\mathcal{B}, sid')$ for some sid'. If not, then ignore. Else, hand $(\texttt{KeyGen}, sid)$ to the adversary. Upon receiving $(\texttt{VerificationKey}, sid, pk_{\mathrm{BS}}, \mathsf{BlSig})$ from the adversary, output $(\texttt{VerificationKey}, sid, pk_{\mathrm{BS}})$ to $\mathcal{B}$ and record the pair $(\mathcal{B}, pk_{\mathrm{BS}}, \mathsf{BlSig})$.

Signature Generation: Upon receiving a value $(\texttt{Sign}, sid, m, pk_{\mathrm{BS}})$ for $m \in \{0,1\}^n$ from some party $\mathcal{U}$, verify that $sid = (\mathcal{B}, sid')$ for some sid'. If not, then ignore. Else, do the following:

- If the user $\mathcal{U}$ is honest then inform $\mathcal{B}$ and the adversary through $(\texttt{Signature}, sid)$ that a signature request takes place and then generate $S \leftarrow \mathsf{BlSig}(m)$ and output $(\texttt{Signature}, sid, m, S)$ to $\mathcal{U}$.
- If the user $\mathcal{U}$ is corrupt then send $(\texttt{Sign}, sid, m)$ to the adversary to obtain $(\texttt{Signature}, sid, m, S)$; abort if $(m, S, pk_{\mathrm{BS}}, 0)$ has been recorded before. Send $(\texttt{Signature}, sid)$ to $\mathcal{B}$.

In either case record $(m, S, pk_{\mathrm{BS}}, 1)$.

Signature Verification: Upon receiving a value $(\texttt{Verify}, sid, m, S, pk'_{\mathrm{BS}})$ from some party $\mathcal{P}$ hand $(\texttt{Verify}, sid, m, S, pk'_{\mathrm{BS}})$ to the adversary. Upon receiving $(\texttt{Verified}, sid, m, S, \phi)$ from the adversary do:

1. If $pk_{\mathrm{BS}} = pk'_{\mathrm{BS}}$ and the entry $(m, S, pk_{\mathrm{BS}}, 1)$ is recorded, then set $f = 1$ (completeness condition).
2. Else, if $pk_{\mathrm{BS}} = pk'_{\mathrm{BS}}$, the bank is not corrupt, and no entry $(m, \star, pk_{\mathrm{BS}}, 1)$ is recorded, then set $f = 0$ and record the entry $(m, S, pk_{\mathrm{BS}}, 0)$ (unforgeability condition).
3. Else, if there is an entry $(m, S, pk'_{\mathrm{BS}}, f')$ recorded, then let $f = f'$ (consistency condition).
4. Else, let $f = \phi$ and record the entry $(m, S, pk'_{\mathrm{BS}}, \phi)$.

Output $(\texttt{Verified}, sid, m, S, f)$ to $\mathcal{P}$.

Figure 2. Blind Signature Functionality $\mathcal{F}_{\mathrm{BlSig}}$

able to find an accepted proof for an invalid statement, even if it sees proofs of the zero-knowledge simulator before (possibly for invalid but different theorems). In our case here the underlying relation R^{ss} is defined by (a sequence of) circuits $\mathsf{C}_n^{\mathrm{ss}}$ evaluating to 1 if and only if for statement $x = U||E||pk'_{\mathsf{Enc}}||crs_{\mathsf{Com}}$ of length $\chi(n) = 2c(n) + 2e'(n)$ and witness $w = m||u||v||u'$ of length $\omega(n) = 4n$ it holds that $E = \mathsf{Enc}(pk'_{\mathsf{Enc}}, m||u||v; u')$ and $U = \mathsf{Com}(crs_{\mathsf{Com}}, m; u)$. Such simulation-sound NIZK proofs exist if trapdoor permutations exist [11].

The final step is to make the signature algorithm Sig of the bank's unforgeable signature scheme deterministic. This can be accomplished by adding a key of a pseudorandom function to the secret signing key. Each time a signature for U is requested the signing algorithm first applies the pseudorandom function to U to get the randomness s with which the signature $B \leftarrow \mathsf{Sig}(sk_{\mathsf{Sig}}, U; s)$ is computed deterministically. The advantage of this modification, which does not require an additional complexity assumption, is that identical requests are now

also answered identically. For the same consistency reason we also presume that the verification algorithm of the regular UNIZK proof is deterministic.

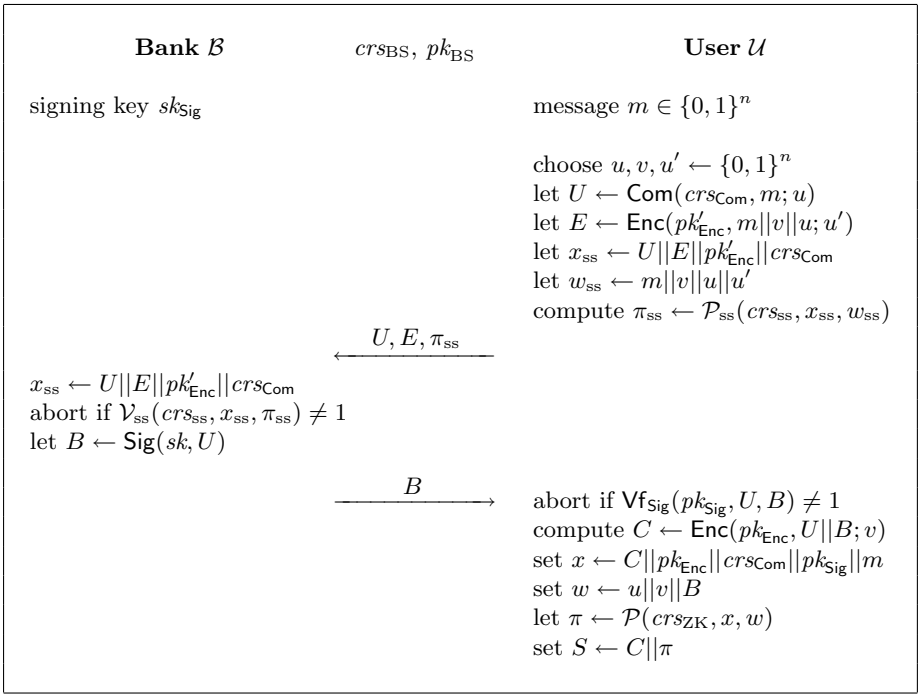

Figure 3. UC Blind Signature Scheme: Issue Protocol

We note that our protocol is defined in the common reference string model. In contrast to the case of UC commitments in [7] where a fresh common reference string for each commitment through $\mathcal{F}_{\mathrm{Com}}$ is required, in our case the once generated common reference string can be used for *several* signature generations by different users; we merely need an independent common reference string for each party taking the role of a bank.

Construction 3 (Universally Composable Blind Signature Scheme). *Let* $(\mathsf{KG}_{\mathsf{Sig}}, \mathsf{Sig}, \mathsf{Vf}_{\mathsf{Sig}})$ *be a signature scheme,* $(\mathsf{KG}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ *be an encryption scheme, and* $(\mathcal{C}_{\mathsf{Com}}, \mathsf{Com})$ *be a commitment scheme. Let* $(\mathcal{C}_{ZK}, \mathcal{P}, \mathcal{V})$ *be a non-interactive zero-knowledge proof system for* R^{BS} *and let* $(\mathcal{C}_{ss}, \mathcal{P}_{ss}, \mathcal{V}_{ss})$ *be a non-interactive zero-knowledge proof system for* R^{ss}*. Define the following four procedures:*

CRS Generation. *Algorithm* $\mathcal{C}_{BS}$ *on input* 1^n *generates* $crs_{ZK} \leftarrow \mathcal{C}_{ZK}(1^n)$*,* $crs_{\mathsf{Com}} \leftarrow \mathcal{C}_{\mathsf{Com}}(1^n)$*,* $crs_{ss} \leftarrow \mathcal{C}_{ss}(1^n)$ *and pairs* $(pk_{\mathsf{Enc}}, sk_{\mathsf{Enc}}), (pk'_{\mathsf{Enc}}, sk'_{\mathsf{Enc}}) \leftarrow \mathsf{KG}_{\mathsf{Enc}}(1^n)$*. It outputs* $crs_{BS} \leftarrow (crs_{ZK}, crs_{\mathsf{Com}}, pk_{\mathsf{Enc}}, pk'_{\mathsf{Enc}}, crs_{ss})$*.*

Key Generation. *If party* $\mathcal{B}$ *with access to* crs_{BS} *receives* $(\mathtt{KeyGen}, sid)$ *it checks that* $sid = (\mathcal{B}, sid')$ *for some* sid'*. If not, it ignores. Else it generates a signa-*

ture key pair $(pk_{\mathsf{Sig}}, sk_{\mathsf{Sig}}) \leftarrow \mathsf{KG}_{\mathsf{Sig}}(1^n)$. *It sets* $(pk_{BS}, sk_{BS}) \leftarrow (pk_{\mathsf{Sig}}, sk_{\mathsf{Sig}})$, *stores* sk_{BS} *and outputs* $(\texttt{VerificationKey}, sid, pk_{\mathrm{BS}})$.

Signature Issue Protocol. *If party* $\mathcal{U}$ *is invoked with input* $(\texttt{Sign}, sid, m, pk_{\mathrm{BS}})$ *for* $sid = (\mathcal{B}, sid')$ *it initiates a run of the interactive protocol in Figure 3 with the bank* $\mathcal{B}$, *where the user gets* m *and* pk_{BS} *as input and the bank uses* sk_{BS} *as input. The user outputs* $(\texttt{Signature}, sid, m, S)$ *for the derived signature value* S.

Signature Verification. *If a party receives* $(\texttt{Verify}, sid, m, S, pk'_{\mathrm{BS}})$ *it parses* S *as* $S = C||\pi$, *computes* $\phi \leftarrow \mathcal{V}(crs, x, \pi)$ *for* $x = C||pk_{\mathsf{Enc}}||crs_{\mathsf{Com}}||pk_{\mathsf{Sig}}||m$ *and outputs* $(\texttt{Verified}, sid, m, \sigma, \phi)$.

Theorem 4. *Let* $(\mathsf{KG}_{\mathsf{Sig}}, \mathsf{Sig}, \mathsf{Vf}_{\mathsf{Sig}})$ *be a length-invariant signature scheme which is unforgeable against adaptive chosen-message attacks and for which* Sig *is deterministic. Let* $(\mathsf{KG}_{\mathsf{Enc}}, \mathsf{Enc}, \mathsf{Dec})$ *be a length-invariant IND-CPA secure encryption scheme,* $(\mathcal{C}_{\mathsf{Com}}, \mathsf{Com})$ *be a length-invariant non-interactive commitment scheme in the common reference string model which is statistically binding. Also let* $(\mathcal{C}_{ZK}, \mathcal{P}, \mathcal{V})$ *be a non-interactive zero-knowledge proof system for* R^{BS} *with deterministic verifier* $\mathcal{V}$. *Let* $(\mathcal{C}_{ss}, \mathcal{P}_{ss}, \mathcal{V}_{ss})$ *be a simulation-sound non-interactive zero-knowledge proof for* R^{ss}. *Then the scheme defined in Construction 3 securely realizes functionality* $\mathcal{F}_{BlSig}$ *for non-adaptive corruptions.*

The idea of the proof is as follows. Algorithm BlSig for functionality $\mathcal{F}_{\mathrm{BlSig}}$, supposed to be determined by the ideal-model adversary and to create signatures for honest users, ignores its input m entirely, but instead prepares a dummy encryption $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, 0^{c(n)+s(n)}; v)$ and appends a fake correctness proof π generated by the zero-knowledge simulator. The output $C||\pi$ is thus indistinguishable from genuinely generated signatures of honest users in the actual scheme. On the other hand, the additional encryption E and the simulation-sound zero-knowledge proof allows the ideal-model adversary to extract potential signatures $C||\pi$ of malicious users (in black-box simulations), and to provide them to the functionality. The completeness and consistency condition of $\mathcal{F}_{\mathrm{BlSig}}$ is realized by the completeness of the underlying scheme, and unforgeability follows as for the basic scheme with concurrent security.

Proof. (of Theorem 4) We have to show that for each adversary $\mathcal{A}$ attacking the real-world protocol there exist an ideal-model adversary (aka. simulator) $\mathcal{S}$ in the ideal world with dummy parties and functionality $\mathcal{F}_{\mathrm{BlSig}}$ such that no environment $\mathcal{Z}$ can distinguish whether it is facing an execution in the real world with $\mathcal{A}$ or one in the ideal world with $\mathcal{S}$.

We build the ideal-model adversary $\mathcal{S}$ by black-box simulation of $\mathcal{A}$, relaying all communication between the environment $\mathcal{Z}$ and the (simulated) adversary $\mathcal{A}$, and acting on behalf of the honest parties in this simulation. Algorithm $\mathcal{S}$ also corrupts a dummy party in the ideal model whenever $\mathcal{A}$ asks to corrupt the corresponding party in the simulation. By assumption this is only done before the execution starts.

The ideal-model simulator $\mathcal{S}$ first generates a reference string crs_{BS} for the black-box simulation by picking encryption keys $(sk_{\mathsf{Enc}}, pk_{\mathsf{Enc}}), (sk'_{\mathsf{Enc}}, pk'_{\mathsf{Enc}}) \leftarrow$

$\mathsf{KG}_{\mathsf{Enc}}(1^n)$, generating $crs_{\mathsf{Com}} \leftarrow \mathcal{C}_{\mathsf{Com}}(1^n)$ and running the zero-knowledge simulators to generates crs_{ss} as well as crs_{ZK} for the zero-knowledge proofs. It outputs $crs_{\text{BS}} = (pk_{\mathsf{Enc}}, pk'_{\mathsf{Enc}}, crs_{\text{ss}}, crs_{\mathsf{Com}}, crs_{\text{ZK}})$. We next describe the simulation of the honest parties in the black-box simulation:

- Suppose the simulator is woken up through a call $(\texttt{KeyGen}, sid)$ from $\mathcal{F}_{\text{BlSig}}$ in the ideal model where $sid = (\mathcal{B}, sid')$. Then the simulator generates $(pk_{\text{BS}}, sk_{\text{BS}}) \leftarrow \mathsf{KG}_{\text{BS}}(1^n)$ as specified by the scheme's description and lets BlSig be the algorithm that on input $m \in \{0,1\}^n$ computes the encryption $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, 0^{c(n)+s(n)}; v)$, and generates π through the zero-knowledge simulator for relation R^{BS} where $x \leftarrow C||pk_{\mathsf{Enc}}||crs_{\mathsf{Com}}||pk_{\mathsf{Sig}}||m$, and outputs $S \leftarrow C||\pi$. Simulator $\mathcal{S}$ returns $(\texttt{VerificationKey}, sid, pk_{\text{BS}}, \mathsf{BlSig})$ to $\mathcal{F}_{\text{BlSig}}$ in the ideal model. In the black-box simulation it sends pk_{BS} to all parties.
- Suppose that the adversary lets a corrupt user in the black-box simulation initiate a protocol run with the honest bank by sending values (U, E, π_{ss}). Then the simulator first checks the validity of the proof π_{ss}; if this check fails then it ignores this message. Else, $\mathcal{S}$ uses the secret key sk'_{Enc} to recover $m||v||u$ from E (and aborts if it fails) and submits $(\texttt{Sign}, sid, m, pk_{\text{BS}})$ on behalf of the user to the ideal functionality. It immediately receives a request $(\texttt{Signature}, sid, m)$ from $\mathcal{F}_{\text{BlSig}}$. To answer, $\mathcal{S}$ computes the signature $B \leftarrow \mathsf{Sig}(sk_{\mathsf{Sig}}, U)$ under the unforgeable signature scheme, an encryption $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, U||B; v)$ and a proof $\pi \leftarrow \mathcal{P}(crs_{\text{ZK}}, x, w)$ for the extracted values and sends $(\texttt{Signature}, sid, m, S)$ for $S \leftarrow C||\pi$ back to the functionality. It also returns B in the black-box simulation to the corrupt user.
- If an honest user requests a signature $(\texttt{Sign}, sid, m, pk_{\text{BS}})$ in the ideal model and waits to receive $(\texttt{Signature}, sid, m, S)$, generated by the functionality through algorithm BlSig, then the ideal-model adversary generates strings $U \leftarrow \mathsf{Com}(crs_{\mathsf{Com}}, 0^n; u)$, $E \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, 0 \ldots 0; u')$ and a proof π_{ss} via the zero-knowledge simulator of the simulation-sound scheme and lets the user in the black-box simulation send these values (U, E, π_{ss}). If the bank is honest, then $\mathcal{S}$ uses the secret signing key sk_{Sig} to compute $B \leftarrow \mathsf{Sig}(sk_{\mathsf{Sig}}, U)$, else it waits to receive a value B from the adversarial controlled bank.
- If $\mathcal{S}$ in the ideal model gets a request $(\texttt{Verify}, sid, m, S, pk'_{\text{BS}})$ then it computes $\phi \leftarrow \mathcal{V}(crs, PK, x, \pi)$ and returns $(\texttt{Verified}, sid, m, S, \phi)$.

This gives a full description of the ideal-model simulator. For the analysis note that there are two main differences between the ideal-model and the black-box simulation compared to an actual attack on the protocol. First, in the black-box simulation we use fake values (commitments and encryptions of zero-strings, simulated proofs etc.). The second point to address is that the verification algorithm in the ideal model returns 0 if there is no recorded pair $(m, \star, pk_{\text{BS}}, 1)$ while in the real-life protocol Vf_{BS} may output 1; for any other verification requests the answers are identical as the verification algorithm Vf_{BS} merely runs the deterministic verification algorithm of the NIZK system (and thus guarantees completeness and, especially, consistency).

We claim that the differences are undetectable for the environment $\mathcal{Z}$. This is proven through a sequence of games transforming an execution in the ideal-

model scenario into one which is equal to the one of the actual protocol. In these games we will have full control over the setting, in particular over the functionality and, in contrast to the ideal-model adversary, we will also read the inputs of $\mathcal{Z}$ to honest users. This is admissible since our goal is to emulate $\mathcal{Z}$'s environment and to use differences in the output behavior to contradict the security of the underlying cryptographic primitives.

- Experiment $\mathsf{Game}_0(n)$ describes the original attack of $\mathcal{Z}$ on the ideal-model simulation (including the black-box simulation of $\mathcal{A}$).
- In $\mathsf{Game}_1(n)$ we change the way the commitments U on behalf of honest users are computed in $\mathcal{A}$'s black-box simulation. Originally, the simulator $\mathcal{S}$ computes a fake commitment $U \leftarrow \mathsf{Com}(crs_{\mathsf{Com}}, 0^n; u)$. Now, whenever the simulator is supposed to create such a commitment, we let $U \leftarrow \mathsf{Com}(crs_{\mathsf{Com}}, m; u)$ for the right value m (given as input to the honest user by $\mathcal{Z}$), and $u, v \leftarrow \{0,1\}^n$ are picked at random. Because of the secrecy of Com it is easy to see that $\mathcal{Z}$'s output behavior will not change significantly when facing $\mathsf{Game}_1(n)$ instead of $\mathsf{Game}_0(n)$.
- Next, in $\mathsf{Game}_2(n)$, we replace every encryption $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, 0^{c(n)+s(n)}; v)$ in the computations of algorithm BlSig through an encryption of the actual values $U||B$, i.e., $C \leftarrow \mathsf{Enc}(pk_{\mathsf{Enc}}, U||B; v)$, where we provide the values U and B transmitted in the black-box simulation "adaptively" to algorithm BlSig. By the security of the encryption scheme this is indistinguishable from the environment's viewpoint.
- In $\mathsf{Game}_3(n)$ we replace every steps of the zero-knowledge proof in the computation of BlSig through steps of the actual proof system, i.e., generation of crs_{ZK} through $crs_{\mathrm{ZK}} \leftarrow \mathcal{C}_{\mathrm{ZK}}(1^n)$, and every generation of π through the prover (for the now genuine witness $w = u||v||B$). By the zero-knowledge property this substitution is indistinguishable for the environment $\mathcal{Z}$.
- Now we turn to the difference between the ideal-model verification through list comparisons and the real-life verification through $\mathsf{Vf}_{\mathrm{BS}}$. In $\mathsf{Game}_4(n)$ every time the verification algorithm in $\mathsf{Game}_3(n)$ was called by some honest user about input $(\texttt{Verify}, sid, m, S, pk_{\mathrm{BS}})$ then we run the verification $\mathsf{Vf}_{\mathrm{BS}}(crs_{\mathrm{BS}}, pk_{\mathrm{BS}}, m, S)$ instead (the case $pk'_{\mathrm{BS}} \neq pk_{\mathrm{BS}}$ is treated as before).
  Consider in $\mathsf{Game}_3(n)$ the event that some user requests the functionality to verify a signature $(\texttt{Verify}, sid, m, S, pk_{\mathrm{BS}})$ such that there is no entry $(m, \star, pk_{\mathrm{BS}}, 1)$ stored by the functionality (and the bank $\mathcal{B}$ is honest) but such that $\mathsf{Vf}_{\mathrm{BS}}$ returns 1. This would clearly allow to distinguish the two games (note that the other direction, that $\mathsf{Vf}_{\mathrm{BS}}$ yields 0 but there is an entry, cannot happen by the completeness of the blind signature scheme). We claim that if such a request should occur with noticeable probability this would contradict the unforgeability of the blind signature scheme in the previous section (there, unforgeability was proven for arbitrary, unforgeable signature scheme for the bank, and thus holds for the deterministic one we consider here as well).
  Specifically, we show how to turn a run of $\mathsf{Game}_3(n)$ with $\mathcal{A}$ and $\mathcal{Z}$ into an attack according to experiment $\mathsf{Forge}^{\mathsf{BS}}_{\mathcal{U}^*}(n)$. For this we run the same experiment as in $\mathsf{Game}_3(n)$ but this time we are use the oracle access to the bank's

signing oracle $\mathsf{Sig}(sk_{\mathsf{Sig}}, \cdot)$ instead of generating the key pair $(sk_{\mathsf{Sig}}, pk_{\mathsf{Sig}})$ ourselves (recall that the bank is assumed to be honest). Each time an honest user receives input $(\mathtt{Sign}, sid, m, pk_{\mathrm{BS}})$ we generate U, E, π_{ss} as in $\mathsf{Game}_3(n)$, including a valid commitment $U \leftarrow \mathsf{Com}(crs_{\mathsf{Com}}, m; u)$, and submit U to the bank to receive B. From this answer we honestly compute the signature $S \leftarrow C||\pi$. We memorize the pair (m, S) and the values (U, B).
If a corrupt user submits U, E, π_{ss} in the black-box simulation then we also check the proof π_{ss} (and do nothing if it is invalid). If the proof is accepted then we check if we have stored a pair (U, B) for some B. If so, we return the same B as before in the black-box simulation but without contacting the bank in the attack $\mathsf{Forge}^{\mathsf{BS}}_{\mathcal{U}^*}(n)$ (since the bank's signature would be identical we answer consistently). If there is no such pair (U, B) we use sk'_{Enc} to extract appropriate values $m||v||u$. By the simulation soundness of the NIZK this extraction works with overwhelming probability (because U has never appeared for an honest user in the execution before). We submit U to the bank's signature oracle to receive a value B, which we return to the corrupt user in the black-box simulation. We also deduce the signature S with the help of the extracted values and record (m, S) and (U, B).
Suppose that a user at some point sends a request $(\mathtt{Verify}, sid, m, S, pk_{\mathrm{BS}})$ for a message m which we have not stored (and which is thus new) but for which $\mathsf{Vf}_{\mathrm{BS}}$ accepts. Then we immediately stop and output all previously stored k message/signature pairs together with (m, S). Note that this implies that all our $k+1$ pairs are accepted by $\mathsf{Vf}_{\mathrm{BS}}$, although we only had k interactions with the bank. Hence, if a mismatch in $\mathsf{Game}_3(n)$ happens with noticeable probability it would refute the unforgeability of the blind signature scheme of the previous section. It follows that $\mathsf{Game}_3(n)$ and $\mathsf{Game}_4(n)$ are indistinguishable.

- In $\mathsf{Game}_5(n)$ we can omit the extraction step where the ideal-model simulator decrypts E from submissions of corrupt users, in particular, we do not need to know the secret key sk'_{Enc} anymore for the simulation. This is so since the verification now only relies on $\mathsf{Vf}_{\mathrm{BS}}$ instead of lists. Furthermore every time we gave a dummy encryption $E \leftarrow \mathsf{Enc}(pk'_{\mathsf{Enc}}, 0 \ldots 0; u')$ for an honest user, we now encrypt the true values $E \leftarrow \mathsf{Enc}(pk'_{\mathsf{Enc}}, m||v||u; u')$ to prepare the correct values U and C. By the security of the encryption scheme this is indistinguishable for the environment.
- In $\mathsf{Game}_6(n)$ we replace the simulation of proofs π_{ss} for honest users through proofs computed by the prover's algorithm for witness $m||v||u$ (with respect to a truly random string crs_{ss}). The zero-knowledge property ensures that this will not significantly affect the environment's output.

All the steps in the final game now are exactly as in an attack on the real protocol with adversary $\mathcal{A}$. Therefore, the environment's output in the ideal-model simulation ($\mathsf{Game}_0(n)$) and the real-world execution ($\mathsf{Game}_6(n)$) are indistinguishable. □

We finally note that, analogously to the case of our concurrently-secure scheme, we can also extend the functionality $\mathcal{F}_{\mathrm{BlSig}}$ to handle partially blind

signatures, and can realize this functionality by starting with the partially blind version of our concurrently-secure blind signature scheme and adding the encryption and the simulation-sound proof as before.

Acknowledgments

We thank the anonymous reviewers for comprehensive comments.

References

1. Masayuki Abe. *A Secure Three-Move Blind Signature Scheme for Polynomially Many Signatures*. Advances in Cryptology — Eurocrypt 2001, Volume 2045 of Lecture Notes in Computer Science, pages 136–151. Springer-Verlag, 2001.
2. Masayuki Abe and Eiichiro Fujisaki. *How to Date Blind Signatures*. Advances in Cryptology — Asiacrypt'96, Volume 1163 of Lecture Notes in Computer Science, pages 244–251. Springer-Verlag, 1996.
3. Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. *The One-More-RSA-Inversion Problems and the Security of Chaum's Blind Signature Scheme*. *Journal of Cryptology*, 16(3):185–215, 2003.
4. Alexandra Boldyreva. *Efficient Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme*. Public-Key Cryptography (PKC) 2003, Volume 2567 of Lecture Notes in Computer Science, pages 31–46. Springer-Verlag, 2003.
5. Ran Canetti. *Universally Composable Security: A new Paradigm for Cryptographic Protocols*. Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS) 2001. IEEE Computer Society Press, for an updated version see `eprint.iacr.org`, 2001.
6. Ran Canetti. *On Universally Composable Notions of Security for Signature, Certification and Authentication*. Proceedings of Computer Security Foundations Workshop (CSFW) 2004. IEEE Computer Society Press, for an updated version see `eprint.iacr.org`, 2004.
7. Ran Canetti and Marc Fischlin. *Universally Composable Commitments*. Advances in Cryptology — Crypto 2001, Volume 2139 of Lecture Notes in Computer Science, pages 19–40. Springer-Verlag, 2001.
8. David Chaum. *Blind Signatures for Untraceable Payments*. Advances in Cryptology — Crypto'82, pages 199–203. Plemum, New York, 1983.
9. Jan Camenisch, Maciej Koprowski, and Bogdan Warinschi. *Efficient Blind Signatures Without Random Oracles*. Security in Communication Networks, Volume 3352 of Lecture Notes in Computer Science, pages 134–148. Springer-Verlag, 2004.
10. Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. *Universally Composable Two-Party and Multi-Party Secure Computation*. Proceedings of the Annual Symposium on the Theory of Computing (STOC) 2002, pages 494–503. ACM Press, 2002.
11. Alfredo De Santis, Giovanni Di Crescenzo, Rafail Ostrovsky, Giuseppe Persiano, and Amit Sahai. *Robust Non-interactive Zero Knowledge*. Advances in Cryptology — Crypto 2001, Volume 2139 of Lecture Notes in Computer Science, pages 566–598. Springer-Verlag, 2001.

12. Ivan Damgård and Jens Groth. *Non-interactive and Reusable Non-Malleable Commitment Schemes*. Proceedings of the Annual Symposium on the Theory of Computing (STOC) 2003, pages 426–437. ACM Press, 2003.
13. Alfredo De Santis and Giuseppe Persiano. *Zero-Knowledge Proofs of Knowledge Without Interaction*. Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)'92, pages 427–436. IEEE Computer Society Press, 1992.
14. Marc Fischlin. *Round-Optimal Composable Blind Signatures in the Common Reference String Model*. full version. available at `www.fischlin.de`, 2006.
15. Uriel Feige, Dror Lapidot, and Adi Shamir. *Multiple NonInteractive Zero Knowledge Proofs Under General Assumption. SIAM Journal on Computing*, 29(1):1–28, 1999.
16. Johan Håstad, Russel Impagliazzo, Leonid Levin, and Michael Luby. *A Pseudorandom Generator from any One-way Function. SIAM Journal on Computing*, 28(4):1364–1396, 1999.
17. Ari Juels, Michael Luby, and Rafail Ostrovsky. *Security of Blind Digital Signatures*. Advances in Cryptology — Crypto'97, Volume 1294 of Lecture Notes in Computer Science, pages 150–164. Springer-Verlag, 1997.
18. Aggelos Kiayias and Hong-Sheng Zhou. *Two-Round Concurrent Blind Signatures without Random Oracles*. Number 2005/435 in Cryptology eprint archive. `eprint.iacr.org`, 2005.
19. Yehuda Lindell. *Bounded-Concurrent Secure Two-Party Computation Without Setup Assumptions*. Proceedings of the Annual Symposium on the Theory of Computing (STOC) 2003, pages 683–692. ACM Press, 2003.
20. Yehuda Lindell. *Lower Bounds for Concurrent Self Composition*. Theory of Cryptography Conference (TCC) 2004, Volume 2951 of Lecture Notes in Computer Science, pages 203–222. Springer-Verlag, 2004.
21. Matt Lepinski, Silvio Micali, and Abhi Shelat. *Fair Zero-Knowledge*. Theory of Cryptography Conference (TCC) 2005, Volume 3378 of Lecture Notes in Computer Science, pages 245–263. Springer-Verlag, 2005.
22. Moni Naor. *Bit Commitment Using Pseudo-Randomness. Journal of Cryptology*, 4(2):151–158, 1991.
23. Moni Naor and Moti Yung. *Universal One-Way Hash Functions and Their Cryptographic Applications*. Proceedings of the Annual Symposium on the Theory of Computing (STOC) 1989, pages 33–43. ACM Press, 1989.
24. Tatsuaki Okamoto. *Efficient Blind and Partially Blind Signatures Without Random Oracles*. Theory of Cryptography Conference (TCC) 2006, Volume 3876 of Lecture Notes in Computer Science, pages 80–99. Springer-Verlag, 2006.
25. David Pointcheval. *Strengthened Security for Blind Signatures*. Advances in Cryptology — Eurocrypt'98, Volume 1403 of Lecture Notes in Computer Science, pages 391–405. Springer-Verlag, 1998.
26. David Pointcheval and Jacques Stern. *Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology*, 13(3):361–396, 2000.
27. John Rompel. *One-Way Functions are Necessary and Sufficient for Secure Signatures*. Proceedings of the Annual Symposium on the Theory of Computing (STOC) 1999, pages 387–394. ACM Press, 1990.
28. Amit Sahai. *Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security*. Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS) 1999. IEEE Computer Society Press, 1999.