J Cryptol (2017) 30:470–494
DOI: 10.1007/s00145-015-9225-1

Journal of CRYPTOLOGY

Security of Blind Signatures Revisited

Dominique Schröder
CISPA, Saarland University, Saarbrücken, Germany
schroeder@me.com

Dominique Unruh
University of Tartu, Tartu, Estonia

Communicated by Serge Vaudenay.

Received 7 January 2015 / Revised 20 November 2015
Online publication 14 January 2016

Abstract. We revisit the security definitions of blind signatures as proposed by Pointcheval and Stern (J Cryptol 13(3):361–396, 2000). Security comprises the notions of one-more unforgeability, preventing a malicious user to generate more signatures than requested, and of blindness, averting a malicious signer to learn useful information about the user's messages. Although this definition is well established nowadays, we show that there are still desirable security properties that fall outside of the model. More precisely, in the original unforgeability definition is not excluded that an adversary verifiably uses the same message m for signing twice and is then still able to produce another signature for a new message $m' \neq m$. Intuitively, this should not be possible; yet, it is not captured in the original definition, because the number of signatures equals the number of requests. We thus propose a stronger notion, called honest-user unforgeability, that covers these attacks. We give a simple and efficient transformation that turns any unforgeable blind signature scheme (with deterministic verification) into an honest-user unforgeable one.
Keywords. Blind signatures, Formalization, Aborts, Probabilistic verification.

1. Introduction

Blind signature schemes have been suggested by Chaum [12,13]. Roughly speaking, this widely studied primitive allows a signer to interactively issue signatures for a user such that the signer learns nothing about the message being signed (*blindness*) while the user cannot compute any additional signature without the help of the signer (*unforgeability*). Typical applications of blind signatures include e-cash, where a bank signs coins withdrawn by users, and e-voting, where an authority signs public keys that voters later use to cast their votes. Another application of blind signature schemes are anonymous credentials, where the issuing authority blindly signs a key [10,11]. Very recently, Microsoft introduced a new technology called U-Prove to "overcome the long-standing dilemma

between identity assurance and privacy" [6,29]. Their solution uses blind signatures as a central building block [6,9].

The two security properties, blindness and unforgeability, have been formalized in [27,31]. The blindness definition [27] basically says that a malicious signer should not be able to link signatures generated in interactions with the user to the individual executions. In other words, the signer cannot tell which session of the signing protocol corresponds to which message. The unforgeability property [31] states that an adversary, even if able to impersonate the user and interact freely with the signer, should not be able to produce more signatures than the number of interactions that took place with the signer.

While the above properties have been formalized unambiguously through the common game-based frameworks, and the definitions seem to capture the basic security requirements appropriately, a closer look reveals that the guarantees are rather fragile with respect to slight changes in the adversary's capabilities: In the traditional definition of unforgeability due to [27,31], the adversary takes the role of the user and needs to output more signatures than interactions with the signer took place. To be precise, it needs to output more *distinct* messages with valid signatures than signer invocations. Assume for the moment that the adversary would be able to compute a signature for a message m' after having faithfully obtained *two* signatures for the *same* message $m \neq m'$ through the signer (but no signatures on other messages). Then the adversary cannot output signatures for more (distinct) messages m, m' than the number of invocations of the signer—namely two—and the scheme would be deemed unforgeable according to the unforgeability notions in [27,31], even though the adversary holds a signature for a fresh message m' which it has never used in the interaction.

The above is not surprising in light of the fact that blindness prevents the signer to know which message has been signed. As such, the adversary above could have let the signer in the two executions sign the messages m and m' instead. Since there is no way to prevent this, the above attack *should* indeed not be considered a success. The situation, however, changes if an honest user would have asked for the two signatures for m. Then the adversary would be able to create the additional signature for m' "out of the blue," without having interacted with the signer and with the assurance that the signer has never issued a signature for m'. A more detailed example is given in Sect. 1.1.

1.1. *Unforgeability in the Presence of Honest Users*

To underline the importance of considering unforgeability in the presence of honest users and motivate our approach, let us first consider an example where this property may be desirable. For this, consider the setting of an online video store such as Netflix. In our setting, we assume that the store is implemented via two entities: the content provider and the reseller. We assume that the contract between client and reseller is a flat rate that allows the client to download a fixed number of movies. For privacy reasons, we do not wish the reseller to know which movies the client actually watches. On the other hand, we wish to ensure that underage clients can only download movies suitable for their age. Suppose that this is implemented through a (trusted) entity, the parental control server whose job is to work as a proxy between reseller and client and to ensure that the client only obtains appropriate movies. Then, to download a movie X, the client first sends her name and X to the parental control server. If X is appropriate for the client, the parental

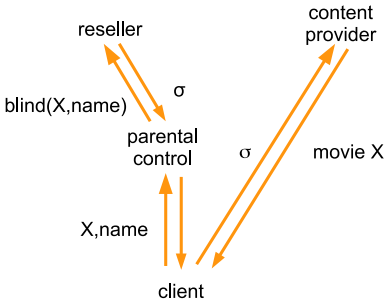

Fig. 1. Setting of an online video store.

control server then runs a blind signature scheme with the reseller to obtain a signature σ on (X, name) (the blind signature is used to protect the privacy of the client, there is no need for the reseller to know which movies the client watches). Then σ is sent to the client, and the client uses σ to download X from the content provider (we assume that all communication is suitably authenticated) (Fig. 1).

At a first glance, it seems that this protocol is secure. In particular, the client will not be able to download a movie that is not approved by the parental control server. It turns out, however, that the client could cheat the parental control server if the scheme does not guarantee unforgeability in the presence of honest users: Assume the client requests a signature on some harmless movie X twice. He will then obtain two signatures σ_1 and σ_2 on X from the parental control server. Then, given σ_1 and σ_2, the client's children, observing the signatures on the computer, may be able to compute a signature on an adult movie Y that has not been approved by the parental control server.

Our first result is to formally confirm that (basic) unforgeability is in general too weak for the above scenario. That is, we show in Sect. 4.2 that blind signature schemes exist that allow such attacks but that are still unforgeable in the sense of [27,31]. (We note that this is independent of the other issues mentioned before, namely unforgeability under aborts and probabilistic verification.)

Defining unforgeability in the presence of honest users To cover attacks like the one above, we thus propose a new *game-based* definition, unforgeability in the presence of honest users, which is a strengthening of unforgeability. Alternatively, one could also define an ideal functionality (see [4,16]) that covers these attacks, but schemes that achieve such strong security properties are usually less efficient. Our definition can be outlined as follows:

Definition 1. (*Unforgeability in the presence of honest users—informal*) If an adversary performs k direct interactions with the signer, and requests signatures for the messages $m_1, \ldots, m_n$ from the user (which produces these signatures by interacting with the signer), then the adversary cannot produce signatures for pairwise distinct messages $m_1^*, \ldots, m_{k+1}^*$ with $\{m_1^*, \ldots, m_{k+1}^*\} \cap \{m_1, \ldots, m_n\} = \varnothing$.

Notice that this definition also covers the hybrid case in which the adversary interacts with an honest user and the signer simultaneously. Alternatively, one could also require security in each of the settings individually: security when there is no honest user (that

is, the regular definition of unforgeability), and security when the adversary may not query the signer directly (we call this $\mathcal{S}+\mathcal{U}$-unforgeability). We show in Sect. 4.4 that requiring these variants of security individually leads to a strictly weaker security notion. Notice that $\mathcal{S}+\mathcal{U}$-unforgeability would be sufficient to solve the problem in our video store example. It seems, however, restrictive to assume that in all protocols, there will always be queries either only from honest users or only from dishonest users, but never from both in the same execution.

Achieving honest-user unforgeability We show that any unforgeable blind signature scheme can be converted into an honest-user unforgeable blind signature scheme. The transformation is very simple and efficient: Instead of signing a message m, in the transformed scheme the user signs the message (m, r) where r is some sufficiently long random string. Furthermore, we show that if a scheme is already strongly unforgeable, then it is strongly honest-user unforgeable (as long as the original scheme is *randomized* which holds for most signature schemes).

1.2. *Insecurity with Probabilistic Verification*

Most (regular or blind) signature schemes have a deterministic verification algorithm. In general, however, having a deterministic verification is not a necessity. Yet, when we allow a probabilistic verification algorithm (and this is usually not explicitly excluded), both the definition of unforgeability and the definition of honest-user unforgeability are subject to an attack: Consider again our video store example. Let λ denote the security parameter. Fix a polynomial $p = p(\lambda) > \lambda$. Assume that the parental control server and the client are malicious and collude. The parental control server interacts with the reseller λ times and produces $p > \lambda$ "half-signatures" on movie names $X_1, \ldots, X_p$. Here, a half-signature means a signature that passes verification with probability $\frac{1}{2}$. Then the client can download the movies $X_1, \ldots, X_p$ from the content provider. (If in some download request, a half-signature does not pass verification, the client just retries his request.) Thus the client got $p > \lambda$ movies, even if his flat rate only allows for downloading λ movies.

Can this happen? It seems that unforgeability would exclude this because $p > \lambda$ signatures were produced using λ queries to the signer. In the definition of unforgeability, however, the adversary succeeds if it outputs $p > \lambda$ signatures such that *all* signatures pass verification. However, the signatures that are produced are half-signatures: That is, the probability that all $p > \lambda$ signatures pass the verification simultaneously is negligible! Thus, producing more than λ half-signatures using λ queries would not be considered an attack by the usual definition of unforgeability. In Sect. 5, we show that blind signature schemes exist that allow such attacks but that satisfy the definition of unforgeability. The same applies to honest-user unforgeability as described so far; we thus need to augment the definition further.

There are two solutions to this problem. One is to explicitly require that the verification algorithm is deterministic. Since most schemes have deterministic verification, this is not a strong restriction. To cover the case of probabilistic verification, we propose an augmented definition of honest-user unforgeability in Sect. 5: This definition considers a list of signatures as a successful forgery if each of them would pass verification with noticeable probability (roughly speaking).

We do not propose a generic transformation that makes schemes with probabilistic verification secure according to our definition. Yet, since most schemes have a deterministic verification anyway, these schemes will automatically satisfy our augmented definition.

1.3. *Related Work*

Many blind signature schemes have been proposed in the literature, and these schemes differ in their round complexity, their underlying computational assumptions, and the model in which the proof of security is given. For example, some schemes rely on the random oracle heuristic [1,4,7,8,31], some constructions are secure in the standard model [2,14,19,25,28,30,33] ([2,19] assume the existence of a common reference string), and some constructions are based on general assumptions [16,22,26,27,33]. Only a few works consider the security of blind signatures [17,27,31] or their round complexity [18,20,22,33].

As mentioned before, Camenisch et al. [15] have already considered the limitations of the standard blindness notion. They have introduced an extension called *selective-failure blindness* in which a malicious signer should not be able to force an honest user to abort the signature issue protocol because of a certain property of the user's message, which would disclose some information about the message to the signer. They present a construction of a simulatable oblivious transfer protocol from the so-called unique selective-failure blind signature schemes (in the random oracle model) for which the signature is uniquely determined by the message. Since the main result of the work [15] is the construction of oblivious transfer protocols, the authors note that Chaum's scheme [12] and Boldyreva's protocol [8] are examples of such selective-failure blind schemes, but do not fully explore the relationship to (regular) blindness.

Hazay et al. [26] present a concurrently secure blind signature scheme and, as part of this, they also introduce a notion called a posteriori blindness. This notion considers blindness of multiple executions between the signer and the user (as opposed to two sessions as in the basic case) and addresses the question how to deal with executions in which the user cannot derive a signature. However, the definition of a posteriori blindness is neither known to be implied by ordinary blindness, nor does it imply ordinary blindness (as sketched in [26]). Thus, selective-failure blindness does not follow from this notion.

Aborts of players have also been studied under the notion of fairness in two-party and multi-party computations, especially for the exchange of signatures, e.g., [5,21,24]. Fairness should guarantee that one party obtains the output of the joint computation if and only if the other party receives it. Note, however, that in case of blind signatures the protocol only provides a one-sided output to the user (namely, the signature). In addition, solutions providing fairness usually require extra-assumptions like a trusted third party in case of disputes, or they add a significant overhead to the underlying protocol.

2. Blind Signatures

Before presenting our results, we briefly recall some basic definitions. In what follows, we denote by $\lambda \in \mathbb{N}$ the security parameter. Informally, we say that a function is *negligible* if it vanishes faster than the inverse of any polynomial. We call a function non-negligible

if it is not negligible. If S is a set, then $x \xleftarrow{\$} S$ indicates that x is chosen uniformly at random over S (which in particular assumes that S can be sampled efficiently).

To define blind signatures formally, we introduce the following notation for interactive executions between algorithms $\mathcal{X}$ and $\mathcal{Y}$. By $(a, b) \leftarrow \langle \mathcal{X}(x), \mathcal{Y}(y) \rangle$ we denote the joint execution of $\mathcal{X}$ and $\mathcal{Y}$, where x is the private input of $\mathcal{X}$ and y defines the private input of $\mathcal{Y}$. The private output of $\mathcal{X}$ equals a and the private output of $\mathcal{Y}$ is b. We write $\mathcal{Y}^{\langle \mathcal{X}(x), \cdot \rangle^{\infty}}(y)$ if $\mathcal{Y}$ can invoke an unbounded number of executions of the interactive protocol with $\mathcal{X}$ in arbitrarily interleaved order. Accordingly, $\mathcal{X}^{\langle \cdot, \mathcal{Y}(y_0) \rangle^1, \langle \cdot, \mathcal{Y}(y_1) \rangle^1}(x)$ can invoke arbitrarily interleaved executions with $\mathcal{Y}(y_0)$ and $\mathcal{Y}(y_1)$, but interact with each algorithm only once. The invoking oracle machine does not see the private output of the invoked machine. In the above definition, this means that $\mathcal{Y}$ does not learn a, and that $\mathcal{X}$ does not learn $\mathcal{Y}$'s outputs.

Definition 2. (*Interactive signature scheme*) We define an interactive signature scheme as a tuple of efficient[1] algorithms $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Vf})$ (the key generation algorithm KG, the signer $\mathcal{S}$, the user $\mathcal{U}$, and the verification algorithm Vf) where

Key Generation	$\mathsf{KG}(1^\lambda)$ for parameter λ generates a key pair (sk, pk).
Signature Issuing	The execution of algorithm $\mathcal{S}(sk)$ and algorithm $\mathcal{U}(pk, m)$ for message $m \in \{0,1\}^*$ generates an output σ of the user, and some output *out* for the signer (possibly empty, or a status message like ok or $\perp$), $(out, \sigma) \leftarrow \langle \mathcal{S}(sk), \mathcal{U}(pk, m) \rangle$.
Verification	$\mathsf{Vf}(pk, m, \sigma)$ outputs a bit.

It is assumed that the scheme is *complete*, i.e., for any function f, with overwhelming probability in $\lambda \in \mathbb{N}$ the following holds: when executing $(sk, pk) \leftarrow \mathsf{KG}(1^\lambda)$, setting $m := f(\lambda, pk, sk)$, and letting σ be the output of $\mathcal{U}$ in the joint execution of $\mathcal{S}(sk)$ and $\mathcal{U}(pk, m)$, then we have $\mathsf{Vf}(pk, m, \sigma) = 1$.

Note that we assume that the message is $\{0,1\}^*$ and in particular independent of the public key. However, our positive constructions (Sect. 4.5) can easily be seen to work in the same way for smaller messages spaces, and message spaces depending on the public key. (As long as they are big enough to contain the appended random number r, of course.)

3. Basic Security Notions for Blind Signatures

Security of blind signature schemes is defined by unforgeability and blindness. We first present the established notions by [27,31].

[1] More precisely, KG and Vf run in polynomial-time in the total length of their inputs. The total running time of $\mathcal{S}$ is polynomial in the total length of its input (sk) plus the total length of its incoming messages. The total running time of $\mathcal{U}$ is polynomial in the total length of its input (pk, m). (But the running time of $\mathcal{U}$ may not depend on its incoming messages.) The asymmetry between the running time of $\mathcal{S}$ and $\mathcal{U}$ is necessary to ensure that (a) an interaction between $\mathcal{U}$ and $\mathcal{S}$ always runs in polynomial-time, and that (b) the running time of $\mathcal{S}$ may depend on the length of the message m that only $\mathcal{U}$ has in its input.

Unforgeability An adversary $\mathcal{U}^*$ against unforgeability tries to generate $k+1$ valid message/signatures pairs with different messages after at most k completed interactions with the honest signer, where the number of executions is adaptively determined by $\mathcal{U}^*$ during the attack. To identify completed sessions, we assume that the honest signer returns a special symbol ok when having sent the final protocol message in order to indicate a completed execution (from its point of view). We remark that this output is "atomically" connected to the final transmission to the user.

Definition 3. (*Unforgeability*) An interactive signature scheme $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Vf})$ is called *unforgeable* if for any efficient algorithm $\mathcal{A}$ (the malicious user), the probability that experiment $\mathsf{Forge}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$ evaluates to 1 is negligible (as a function of λ) where

Experiment $\mathsf{Forge}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$
$(sk, pk) \leftarrow \mathsf{KG}(1^\lambda)$
$((m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*)) \leftarrow \mathcal{A}^{\langle \mathcal{S}(sk), \cdot \rangle^\infty}(pk)$
Return 1 iff
$m_i^* \neq m_j^*$ for all i, j with $i \neq j$, and
$\mathsf{Vf}(pk, m_i^*, \sigma_i^*) = 1$ for all i, and
$\mathcal{S}$ has returned ok in at most k interactions.

An interactive signature scheme is *strongly unforgeable* if the condition "$m_i^* \neq m_j^*$ for i, j with $i \neq j$" in the above definition is substituted by "$(m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$ for i, j with $i \neq j$".

Observe that the adversary $\mathcal{A}$ does not learn the private output *out* of the signer $\mathcal{S}(sk)$. We assume schemes in which it can be efficiently determined from the interaction between signer and adversary whether the signer outputs ok. If this is not the case, we need to augment the definition and explicitly give the adversary access to the output *out* since *out* might leak information that the adversary could use to produce forgeries.

Blindness The blindness condition says that it should be infeasible for a malicious signer $\mathcal{S}^*$ to decide which of two messages m_0 and m_1 has been signed first in two executions with an honest user $\mathcal{U}$. This condition must hold, even if $\mathcal{S}^*$ is allowed to choose the public key maliciously [3]. If one of these executions has returned $\perp$, then the signer is not informed about the other signature. (Otherwise, the signer could trivially identify one session by making the other abort.)

Definition 4. (*Blindness*) A blind signature scheme $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Vf})$ is called *blind* if for any efficient algorithm $\mathcal{S}^*$ (working in modes find, issue, and guess), the probability that the following experiment $\mathsf{Blind}^{\mathsf{BS}}_{\mathcal{S}^*}(\lambda)$ evaluates to 1 is negligibly close to $1/2$, where

Experiment $\mathsf{Blind}^{\mathsf{BS}}_{\mathcal{S}^*}(\lambda)$
$(pk, m_0, m_1, \mathsf{st}_{\mathsf{find}}) \leftarrow \mathcal{S}^*(\mathsf{find}, 1^\lambda)$
$b \xleftarrow{\$} \{0, 1\}$
$\mathsf{st}_{\mathsf{issue}} \leftarrow \mathcal{S}^{*\langle \cdot, \mathcal{U}(pk, m_b) \rangle^1, \langle \cdot, \mathcal{U}(pk, m_{1-b}) \rangle^1}(\mathsf{issue}, \mathsf{st}_{\mathsf{find}})$
and let σ_b, σ_{1-b} denote the (possibly undefined) local outputs

of $\mathcal{U}(pk, m_b)$ resp. $\mathcal{U}(pk, m_{1-b})$.
set $(\sigma_0, \sigma_1) = (\bot, \bot)$ if $\sigma_0 = \bot$ or $\sigma_1 = \bot$
$b^* \leftarrow \mathcal{S}^*(\mathsf{guess}, \sigma_0, \sigma_1, \mathsf{st}_{\mathsf{issue}})$
return 1 iff $b = b^*$.

4. Unforgeability in the Presence of Honest Users

In this section, we introduce our stronger notion of unforgeability in the presence of honest users.

4.1. *Definition*

Before proposing the new definition, we fix some notation. Let $\mathcal{P}(sk, pk, \cdot)$ be an oracle that on input a message m executes the signature issue protocol $\langle \mathcal{S}(sk), \mathcal{U}(pk, m)\rangle$ obtaining a signature σ. Let trans denote the transcript of the messages exchanged in such an interaction. We assume that the transcript consists of all messages exchanged between the parties. This oracle then returns (σ, trans).

(The execution of $\langle \mathcal{S}(sk), \mathcal{U}(pk, m)\rangle$ by $\mathcal{P}$ is atomic, i.e., during a call to $\mathcal{P}$, no other interactions take place. And if the interaction aborts, $(\bot, \mathsf{trans})$ is returned where trans describes the transcript up to that point.)

Definition 5. (*Honest-user unforgeability*) An interactive signature scheme $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U}\rangle, \mathsf{Vf})$ is *honest-user unforgeable* if Vf is deterministic and the following holds: For any efficient algorithm $\mathcal{A}$ the probability that experiment $\mathsf{HForge}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$ evaluates to 1 is negligible (as a function of λ) where

Experiment $\mathsf{HForge}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$
$(sk, pk) \leftarrow \mathsf{KG}(1^\lambda)$
$((m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*)) \leftarrow \mathcal{A}^{\langle \mathcal{S}(sk), \cdot\rangle^\infty, \mathcal{P}(sk, pk, \cdot)}(pk)$
Let $m_1, \ldots, m_n$ be the messages queried to $\mathcal{P}(sk, pk, \cdot)$.
Return 1 iff
$m_i^* \neq m_j$ for all i, j
$m_i^* \neq m_j^*$ for all i, j with $i \neq j$, and
$\mathsf{Vf}(pk, m_i^*, \sigma_i^*) = 1$ for all i, and
$\mathcal{S}$ has returned ok in at most k interactions with $\mathcal{A}$.

(Note that, when counting the interactions in which $\mathcal{S}$ returns ok, we do not count the interactions simulated by $\mathcal{P}$.)

An interactive signature scheme is *strongly honest-user unforgeable* if the condition "$m_i^* \neq m_j$ for all i, j" in the above definition is substituted by "$(m_i^*, \sigma_i^*) \neq (m_j, \sigma_j)$ for all i, j" and if we change the condition "$m_i^* \neq m_j^*$ for all i, j with $i \neq j$" to "$(m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$ for all i, j with $i \neq j$".

Notice that we require Vf to be deterministic. When we drop this requirement, the definition does not behave as one would intuitively expect. We explain this problem in detail in Sect. 5. Note further that this definition can be further strengthened by giving

the adversary also the randomness of the honest user and that all our results and proofs also hold for this stronger definition.

4.2. *Unforgeability Does Not Imply Honest-User Unforgeability*

We show that unforgeability does not imply honest-user unforgeability. In particular, a merely unforgeable blind signature scheme does not exclude the attack on the video store described in Sect. 1.1. The high-level idea of our counterexample is to change the verification algorithm of an interactive signature scheme such that it accepts a message m' if it obtains as input two distinct and valid signatures on some message $m \neq m'$ (in addition to accepting honestly generated signatures). More precisely, fix an unforgeable and blind signature scheme $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Vf})$ that is strongly unforgeable. Fix some efficiently computable injective function $f \neq id$ on bitstrings (e.g., $f(m) := 0\|m$). We construct a blind signature scheme $\mathsf{BS}_1 = (\mathsf{KG}_1, \langle \mathcal{S}_1, \mathcal{U}_1 \rangle, \mathsf{Vf}_1)$ as follows:

- $\mathsf{KG}_1 := \mathsf{KG}$, $\mathcal{S}_1 := \mathcal{S}$, and $\mathcal{U}_1 := \mathcal{U}$.
- $\mathsf{Vf}_1(pk, m, \sigma)$ executes the following steps:
  - Invoke $v := \mathsf{Vf}(pk, m, \sigma)$. If $v = 1$, return 1.
  - Otherwise, parse σ as (σ^1, σ^2). If parsing fails or $\sigma^1 = \sigma^2$, return 0.
  - Invoke $v_i := \mathsf{Vf}(pk, f(m), \sigma^i)$ for $i = 1, 2$. If $v_1 = v_2 = 1$, return 1. Otherwise return 0.

Lemma 6. *If* BS *is complete, strongly unforgeable, and blind, then* BS_1 *is complete, unforgeable, and blind.*

Blindness and completeness of BS_1 follow directly from the blindness and completeness of BS. The main idea behind unforgeability is the following: The only possibility for the adversary to forge a signature is to obtain two different signatures σ_1, σ_2 on the same message $f(m)$. Then (σ_1, σ_2) is a valid signature on m. However, since the underlying scheme BS is strongly unforgeable, the adversary can only get σ_1, σ_2 by performing two signing queries. Thus, using two queries, the adversary gets two signatures on the message $f(m)$ and one on m. This is not sufficient to break the unforgeability of BS_1 since the adversary would need to get signatures on three different messages for that.

Proof of Lemma 6. Assume for the sake of contradiction that BS_1 is not unforgeable. Then, there is an efficient adversary $\mathcal{A}$ that succeeds in the unforgeability game for BS_1 with non-negligible probability. This attacker, when given oracle access to the signer $\mathcal{S}_1$, returns a $(k+1)$-tuple $((m_1, \sigma_1), \ldots, (m_{k+1}, \sigma_{k+1}))$ of message/signature pairs, where $\mathsf{Vf}_1(pk, m_i, \sigma_i) = 1$ for all i and $m_i \neq m_j$ for all $i \neq j$ and where $\mathcal{S}$ has returned ok at most k times. In the following, we call such a tuple *k-bad*. We now show how to build an algorithm $\mathcal{B}$ that wins the strong unforgeability game of BS.

The input of the algorithm $\mathcal{B}$ is the public key pk. It runs a black-box simulation of $\mathcal{A}$ on input pk and answers all oracle queries with its own oracle by simply forwarding all messages. Eventually, $\mathcal{A}$ stops, outputting a tuple $F := ((m_1, \sigma_1), \ldots, (m_{k+1}, \sigma_{k+1}))$. Suppose in the following that $\mathcal{A}$ succeeds. Then the tuple F is k-bad. We will show how to efficiently construct from F $k+1$ distinct message/signature pairs (m_i^*, σ_i^*) that

verify under $\mathsf{Vf}(pk, \cdot, \cdot)$. Now, consider a message/signature pair (m, σ) and observe that the verification algorithm Vf_1 outputs 1 if $\mathsf{Vf}(pk, m, \sigma) = 1$ *or* if $\sigma = (\sigma^1, \sigma^2)$ (where $\sigma^1 \neq \sigma^2$) and $\mathsf{Vf}(pk, f(m), \sigma^1) = \mathsf{Vf}(pk, f(m), \sigma^2) = 1$. We define two sets V_0 and V_1 where V_1 is the set that contains a message/signature pairs (m_i, σ_i) that verify under the first condition, and the set V_0 contains all pairs (m_i, σ_i) (with $\sigma_i = (\sigma_i^1, \sigma_i^2)$) that verify under the second condition, i.e.,

$$V_1 := \{(m_i, \sigma_i) : \mathsf{Vf}(pk, m_i, \sigma_i) = 1\} \quad \text{and} \quad V_0 := \{(m_i, \sigma_i) : \mathsf{Vf}(pk, m_i, \sigma_i) = 0\}.$$

Clearly, since $\mathcal{A}$ succeeds and F is k-bad, all messages m_i are distinct and hence $|V_0| + |V_1| = k + 1$. Next, we define the set V_0' that consists of the message/signature pairs $(f(m_i), \sigma_i^1), (f(m_i), \sigma_i^2)$ where (m_i, σ_i) ranges over V_0. Formally,

$$V_0' := \{(f(m_i), \sigma_i^1), (f(m_i), \sigma_i^2) : (m_i, (\sigma_i^1, \sigma_i^2)) \in V_0\}.$$

Note that V_0 contains only elements (m_i, σ_i) with $\mathsf{Vf}_1(m_i, \sigma_i) = 1$ and $\mathsf{Vf}(m_i, \sigma_i) = 0$. By definition of Vf_1 this implies that $\sigma_i = (\sigma_i^1, \sigma_i^2)$ with $\sigma_i^1 \neq \sigma_i^2$ and $\mathsf{Vf}(f(m_i), \sigma_i^1) = \mathsf{Vf}(f(m_i), \sigma_i^2) = 1$. Thus $\left|V_0'\right| = |V_0|$ and for all $(m, \sigma) \in V_0' \cup V_1$ we have that $\mathsf{Vf}(pk, m, \sigma) = 1$. We proceed to show that $\left|V_0' \cup V_1\right| \geq k + 1$ and we then let $\mathcal{B}$ output this set. First note that for any $(m_i, (\sigma_i^1, \sigma_i^2)) \in V_0$, at most one of $(f(m_i), \sigma_i^1), (f(m_i), \sigma_i^2)$ is contained in V_1. Otherwise, V_1 would either contain two pairs (m, σ) with the same m, or $\sigma_i^1 = \sigma_i^2$. Furthermore, since f is injective, for any distinct $(m_i, (\sigma_i^1, \sigma_i^2)), (m_j, (\sigma_j^1, \sigma_j^2)) \in V_0$ we have $m_i \neq m_j$. Hence $(f(m_i), \sigma_i^a) \neq (f(m_j), \sigma_j^b)$ for any $a, b \in \{1, 2\}$ and $i \neq j$. Thus $\left|V_0' \backslash V_1\right| \geq |V_0|$ and therefore

$$\left|V_0' \cup V_1\right| = \left|(V_0' \backslash V_1) \dot{\cup} V_1\right| = \left|V_0' \backslash V_1\right| + |V_1| \geq |V_0| + |V_1| = k + 1.$$

The algorithm $\mathcal{B}$ then computes the set $V_0' \cup V_1$ and then picks distinct pairs

$$(m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*) \in V_0' \cup V_1$$

and outputs $(m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*)$.

Analysis Obviously, $\mathcal{B}$ is efficient because $\mathcal{A}$ is efficient and because the overhead of handling all queries can be done efficiently. Since $\mathcal{A}$ outputs a k-bad tuple with non-negligible probability in the unforgeability game for BS_1, it follows that $\mathcal{B}$ outputs $k+1$ distinct (m_i^*, σ_i^*) with $\mathsf{Vf}(m_i^*, \sigma_i^*) = 1$ in the unforgeability game for BS with at least the same probability. Thus, $\mathcal{B}$ breaks the strong unforgeability of BS. Since we assumed that BS is strongly unforgeable, we have a contradiction, thus our initial assumption that BS_1 is not unforgeable was false. □

Before proving the next lemma, we need to define what a randomized (interactive) signature is. Roughly speaking, schemes that have this property output the same signature in two independent executions with same message only with negligible probability.

Definition 7. (*Randomized signature scheme*) An interactive signature scheme $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Vf})$ is *randomized* if with overwhelming probability in $\lambda \in \mathbb{N}$ the

following holds: for any (sk, pk) in the range of $\mathsf{KG}(1^\lambda)$, any message $m \in \{0,1\}^*$, we have $\sigma_1 \neq \sigma_2$ where $\sigma_1 \leftarrow \langle \mathcal{S}(sk), \mathcal{U}(pk, m)\rangle$ and $\sigma_2 \leftarrow \langle \mathcal{S}(sk), \mathcal{U}(pk, m)\rangle$. The probability is taken over the random coins of KG, $\mathcal{S}$ and $\mathcal{U}$.

Note that any scheme can easily be modified such that it satisfies this definition by letting the user algorithm pick some random value r, setting $m' \leftarrow m\|r$, and including r in the signature. (See Construction 1 on Page 13.)

Lemma 8. *If* BS *is complete and randomized, then* BS_1 *is not honest-user unforgeable.*

Proof. We construct an efficient adversary $\mathcal{A}$ against BS_1 as follows: Let $m \in \{0,1\}^*$ be such that $f(m) \neq m$. Recall that $f \neq id$, and therefore such a value m exists. Note that we can hardcode m directly into the adversary and therefore it is not necessary that m can be efficiently found.

The attacker $\mathcal{A}$ queries $\mathcal{P}$ (the machine simulating $\langle \mathcal{S}_1, \mathcal{U}_1\rangle$) twice, both times with the same message $f(m)$, and obtains the signatures σ_1 and σ_2. Since BS is randomized, and $\mathcal{S}_1 = \mathcal{S}$ and $\mathcal{U}_1 = \mathcal{U}$, with overwhelming probability $\sigma_1 \neq \sigma_2$. Since BS is complete, $\mathsf{Vf}(pk, f(m), \sigma_1) = \mathsf{Vf}(pk, f(m), \sigma_2) = 1$ with overwhelming probability. Hence with overwhelming probability, $\mathsf{Vf}_1(pk, m, \sigma) = 1$ for $\sigma := (\sigma_1, \sigma_2)$. The adversary $\mathcal{A}$ outputs (m, σ). Since $\mathcal{A}$ never queried $\mathcal{S}$, and because $\mathcal{A}$ only queries $f(m) \neq m$ from $\mathcal{P}$, this breaks the honest-user unforgeability of BS_1. □

Theorem 9. *If complete, blind, and strongly unforgeable interactive signature schemes exist, then there are complete, blind, and unforgeable interactive signature schemes that are not honest-user unforgeable.*

Proof. If complete, blind, and strongly unforgeable interactive signature schemes exist, then there is a complete, blind, strongly unforgeable, and randomized interactive signature scheme BS (e.g., by applying the transformation from Sect. 4.5). From BS we construct BS_1 as described at the beginning of the section. By Lemmas 6 and 8, BS_1 is complete, blind, and unforgeable but not honest-user unforgeable. □

4.3. *Strong Honest-User Unforgeability*

In this section, we show that strong unforgeability implies strong honest-user unforgeability.

Lemma 10. *Assume that* BS *is complete,*[2] *randomized, and strongly unforgeable. Then* BS *is strongly honest-user unforgeable.*

[2] Completeness is actually necessary to show this lemma: For example, let BS' be a scheme derived from a complete and strongly unforgeable scheme BS in the following way: All machines except for the user are the same in BS and BS'. When the user $\mathcal{U}'$ should sign a message m, he signs $m+1$ instead. Since the user does not occur in the definition of strong unforgeability, the strong unforgeability of BS implies the strong unforgeability of BS'. Yet BS' is not strongly honest-user unforgeable: By performing a signature query for m from the user $\mathcal{U}'$, the adversary can get a valid signature for $m+1$.

This lemma shows that for strongly unforgeable schemes, the traditional (non-honest-user) definition of unforgeability is sufficient. It can also easily be shown that strong unforgeability is strictly stronger than honest-user unforgeability. The separating example appends a bit b to the signature that is ignored by the verification algorithm. Then the signature can easily be changed by flipping the bit. Thus honest-user unforgeability lies strictly between unforgeability and strong unforgeability.

Proof of Lemma 10. Assume that BS is not strongly honest-user unforgeable. Then there is an adversary $\mathcal{A}$ in the strong honest-user unforgeability game for BS such that with non-negligible probability, the following holds:

(i) The adversary outputs a tuple $((m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*))$ for some k.
(ii) The signer $\mathcal{S}$ outputs ok at most k times.
(iii) For all $i \neq j$, we have $(m_i^*, \sigma_i^*) \neq (m_j^*, \sigma_j^*)$.
(iv) For all i, we have $\mathsf{Vf}(pk, m_i^*, \sigma_i^*) = 1$.
(v) Let $m_1, \ldots, m_n$ be the messages queried from the user $\mathcal{U}$ (which is part of the oracle $\mathcal{P}$), and let $\sigma_1, \ldots, \sigma_n$ be the corresponding answers. Then $(m_i, \sigma_i) \neq (m_j^*, \sigma_j^*)$ for all i, j.

Furthermore, since BS is complete, with overwhelming probability we have that

(vi) $\mathsf{Vf}(pk, m_i, \sigma_i) = 1$ for all i.

And since BS is randomized, with overwhelming probability we have that

(vii) $(m_i, \sigma_i) \neq (m_j, \sigma_j)$ for all $i \neq j$.

This implies that, with non-negligible probability, properties (i)–(vi) hold. Let $(\tilde{m}_1^*, \tilde{\sigma}_1^*), \ldots, (\tilde{m}_{k+n+1}^*, \tilde{\sigma}_{k+n+1}^*)$ be the sequence $(m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*), (m_1, \sigma_1), \ldots, (m_n, \sigma_n)$. From properties (iii), (v), and (vi), it follows that $(\tilde{m}_i^*, \tilde{\sigma}_i^*) \neq (\tilde{m}_j^*, \tilde{\sigma}_j^*)$ for all $i \neq j$. From (iv) and (vi), it follows that $\mathsf{Vf}(pk, \tilde{m}_i^*, \tilde{\sigma}_i^*) = 1$ for all i.

Let $\mathcal{B}$ be an adversary for the strong unforgeability game, constructed as follows: $\mathcal{B}$ simulates $\mathcal{A}$ and $\mathcal{U}$ in a black-box fashion. Whenever $\mathcal{A}$ queries $\mathcal{U}$, then $\mathcal{B}$ invokes the simulated user algorithm $\mathcal{U}$. If the simulated user $\mathcal{U}$ or the simulated $\mathcal{A}$ communicates with the signer, then $\mathcal{B}$ routes this communication to the external signer $\mathcal{S}$. Finally, $\mathcal{B}$ outputs $(\tilde{m}_1^*, \tilde{\sigma}_1^*), \ldots, (\tilde{m}_{k+n+1}^*, \tilde{\sigma}_{k+n+1}^*)$. Then we have that in the strong unforgeability game, with non-negligible probability, $\mathcal{B}$ outputs a tuple $(\tilde{m}_1^*, \tilde{\sigma}_1^*), \ldots, (\tilde{m}_{k+n+1}^*, \tilde{\sigma}_{k+n+1}^*)$ such that $(\tilde{m}_i^*, \tilde{\sigma}_i^*) \neq (\tilde{m}_j^*, \tilde{\sigma}_j^*)$ for all $i \neq j$ and $\mathsf{Vf}(pk, \tilde{m}_i^*, \tilde{\sigma}_i^*) = 1$ for all i and the signer outputs ok at most $k + n$ times (k times due to the invocations from $\mathcal{A}$, and n times due to the invocations from the simulated $\mathcal{U}$). This violates the strong unforgeability of BS, we have a contradiction, and thus BS is strongly honest-user unforgeable. □

Implications for known blind signature schemes Lemma 10 shows that for strongly unforgeable schemes, the traditional definition of unforgeability is sufficient. This immediately shows that the unique blind signature schemes based on RSA [7], as well as the scheme by Boldyreva [8] are honest-user unforgeable. However, most known blind signature schemes (e.g., [2,17,22,23,26,32]) are not strongly unforgeable and it is an open problem whether these schemes are secure with respect to our definition.

4.4. $\mathcal{S}+\mathcal{U}$-Unforgeability

The motivating example from Sect. 1.1 shows us that there is an attack that is not covered by the usual definition of unforgeability of blind signatures: An adversary may create signatures for messages that he has never queried signatures for. Such behavior is not excluded by the usual definition of unforgeability for blind signature schemes. It is, however, excluded by the usual definition of unforgeability for normal (non-interactive) signatures schemes. Indeed, every blind (and non-blind) interactive signature scheme defines a non-interactive signature scheme Sig in which signing just consists of running an interaction between honest signer and honest user. Unforgeability of Sig then excludes the attack described in the motivating example.

So one may wonder if, instead of requiring honest-user unforgeability, it might not be sufficient to just require the blind signature scheme to be unforgeable both as a non-interactive scheme (we call that "$\mathcal{S}+\mathcal{U}$-unforgeability" below) and as a blind signature scheme according to the usual definition of unforgeability (i.e., Definition 3). At least the motivating example is covered.

We show below that indeed, $\mathcal{S}+\mathcal{U}$-unforgeability together with the usual definition of unforgeability is not be sufficient, since it does not exclude attacks that result from a combination of honest and dishonest signing queries. We first give a formal definition:

Definition 11. ($\mathcal{S}+\mathcal{U}$-*unforgeability*) Let $\mathsf{BS}=(\mathsf{KG},\langle\mathcal{S},\mathcal{U}\rangle,\mathsf{Vf})$ be an interactive signature scheme. We define Sig as the algorithm that gets as input (pk,sk,m) and simulates $(out,\sigma)\leftarrow\langle\mathcal{S}(sk),\mathcal{U}(pk,m)\rangle$ and returns σ. The scheme BS is $\mathcal{S}+\mathcal{U}$-unforgeable (resp. strongly unforgeable), if $(\mathsf{KG},\mathsf{Sig},\mathsf{Vf})$ is unforgeable (resp. strongly unforgeable).

Let us rephrase our question: If a scheme is interactively unforgeable and $\mathcal{S}+\mathcal{U}$-unforgeable, is it then automatically honest-user unforgeable? We settle this question in the negative. The main intuition why this is not implied is that both properties are considered independently of each other. Thus, we construct the following counterexample where we can forge a signature if we combine malicious queries together with honest protocol executions.

Fix an interactive signature scheme $\mathsf{BS}=(\mathsf{KG},\langle\mathcal{S},\mathcal{U}\rangle,\mathsf{Vf})$ that is complete, randomized, blind, and strongly unforgeable. Fix some efficiently computable injective function $f\neq id$ on bitstrings (e.g., $f(m):=0\|m$) and let g be a one-way function. We construct an interactive signature scheme $\mathsf{BS}_2=(\mathsf{KG}_2,\langle\mathcal{S}_2,\mathcal{U}_2\rangle,\mathsf{Vf}_2)$ as follows:

- $\mathsf{KG}_2(1^\lambda)$ computes a key pair $(sk,pk)\leftarrow\mathsf{KG}(1^\lambda)$, it picks a random x in the domain of g, it sets $y:=g(x)$, $sk_2:=(sk,x)$, and $pk_2:=(pk,y)$ and returns (sk_2,pk_2).
- $\mathcal{S}_2((sk,x))$ behaves like $\mathcal{S}(sk)$, except for the following extension: At any point in the interaction, the user may send a message `getx` (which is assumed never to be sent by the honest user $\mathcal{U}$), whereupon $\mathcal{S}_2$ will return x. Thereafter, the interaction continues as with $\mathcal{S}$. (In other words, a malicious user may retrieve x for free.)
- $\mathcal{U}_2((pk,y),m)$ executes $\mathcal{U}(pk,m)$.
- $\mathsf{Vf}_2((pk,y),m,\sigma)$ executes the following steps:
  - Invoke $v:=\mathsf{Vf}(pk,m,\sigma)$. If $v=1$, return 1.

– Otherwise, parse σ as (σ_1, σ_2, x'). If parsing fails or $\sigma_1 = \sigma_2$ or $f(x') \neq y$, return 0.
– Invoke $v_i := \mathsf{Vf}(pk, f(m), \sigma_i)$ for $i = 1, 2$. If $v_1 = v_2 = 1$, return 1. Otherwise return 0.

Notice that the only change with respect to the counterexample from the previous section is that the secret key now contains a secret value x that is needed to "unlock" the possibility of producing additional signatures. This value x can be accessed easily by a malicious user, but an honest user will never get this value.

Lemma 12. *If* BS *is complete, strongly unforgeable, and blind, then* BS_2 *is complete, unforgeable, and blind.*

The proof is analogous to that of Lemma 6 and is omitted.

Lemma 13. *If* BS *is strongly unforgeable, complete, and randomized, then* BS_2 *is strongly* $\mathcal{S} + \mathcal{U}$*-unforgeable.*

Proof. We define Sig_2 as the algorithm that gets as input (pk, sk, m) and simulates $(out, \sigma) \leftarrow \langle \mathcal{S}_2(sk), \mathcal{U}_2(pk, m) \rangle$ and returns σ. Analogously, we define Sig simulating $\mathcal{S}$ and $\mathcal{U}$. By definition, to show that BS_2 is strongly $\mathcal{S} + \mathcal{U}$-unforgeable, we have to show that $(\mathsf{KG}_2, \mathsf{Sig}_2, \mathsf{Vf}_2)$ is strongly unforgeable.

Assume that this is not the case and that there is an adversary $\mathcal{A}$ that breaks the strong unforgeability game for $(\mathsf{KG}_2, \mathsf{Sig}_2, \mathsf{Vf}_2)$. Note that since $\mathcal{U}_2$ never sends `getx`, Sig_2 never accesses x. Thus, in the strong unforgeability game, x is only used to produce $y = f(x)$. Since g is a one-way function, the probability that the signature $\sigma = (\sigma_1, \sigma_2, x')$ output by the adversary $\mathcal{A}$ contains x' such that $f(x') = y$ is negligible. On the other hand, if the signatures do not contain such an x', then Vf_2 coincides with Vf. But then, $\mathcal{A}$ breaks the unforgeability game for $(\mathsf{KG}, \mathsf{Sig}, \mathsf{Vf})$, which would imply that $(\mathsf{KG}, \mathsf{Sig}, \mathsf{Vf})$ is not strongly unforgeable.

However, since BS is strongly unforgeable, complete, and randomized, by Lemma 10, BS is strongly honest-user unforgeable which is easily seen to imply that BS is $\mathcal{S} + \mathcal{U}$-unforgeable. By definition, this contradicts the fact that $(\mathsf{KG}, \mathsf{Sig}, \mathsf{Vf})$ is not strongly unforgeable. Hence our assumption that $(\mathsf{KG}_2, \mathsf{Sig}_2, \mathsf{Vf}_2)$ is not strongly unforgeable was false. □

Lemma 14. *If* BS *is complete and randomized, then* BS_2 *is not honest-user unforgeable.*

Proof. We construct an adversary $\mathcal{A}$ against BS_2 as follows: Let $m \in \{0, 1\}^*$ be such that $f(m) \neq m$ and fix some m' with $m \neq m' \neq f(m)$. The adversary $\mathcal{A}$ queries $\mathcal{P}$ (the oracle simulating $\langle \mathcal{S}_2, \mathcal{U}_2 \rangle$) twice, both times with the same message $f(m)$. Call the resulting signatures σ_1 and σ_2. Since BS is randomized, and both $\mathcal{S}_1 = \mathcal{S}$ and $\mathcal{U}_1 = \mathcal{U}$ except for the different format of the public and secret key and for the fact that $\mathcal{S}_1$ additionally reacts to the message `getx`, with overwhelming probability, we have $\sigma_1 \neq \sigma_2$. Since BS is complete, with overwhelming probability, we have $\mathsf{Vf}(pk, f(m), \sigma_1) = \mathsf{Vf}(pk, f(m), \sigma_2) = 1$. Then the adversary $\mathcal{A}$ interacts with

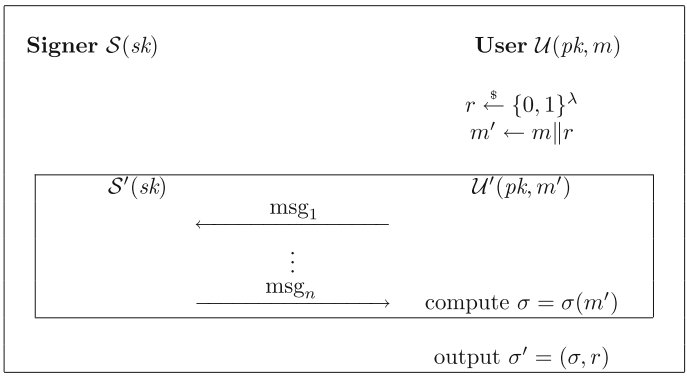

Fig. 2. Issue protocol of the blind signature scheme.

$\mathcal{S}_2$ directly to get a signature σ' for m'. Here $\mathcal{A}$ behaves like an honest $\mathcal{U}_2$, except that it additionally sends the message `getx` and learns x. Since BS is complete, with overwhelming probability, we have $\mathsf{Vf}(pk, m', \sigma') = 1$. Since $y = f(x)$ and $\mathsf{Vf}(pk, f(m), \sigma_1) = \mathsf{Vf}(pk, f(m), \sigma_2) = 1$ and $\sigma_1 \neq \sigma_2$, with overwhelming probability, we have $\mathsf{Vf}_2(pk_2, m, \sigma) = 1$ for $\sigma := (\sigma_1, \sigma_2, x)$. The adversary $\mathcal{A}$ outputs (m, σ) and (m', σ'). Since $\mathcal{A}$ queried $\mathcal{S}$ only once, and because $\mathcal{A}$ only queries $f(m) \neq m, m'$ from $\mathcal{U}$, this breaks the honest-user unforgeability of BS_2. □

Theorem 15. *If complete, blind, and strongly unforgeable interactive signature schemes exist, then there are complete, blind, unforgeable, and strongly $\mathcal{S} + \mathcal{U}$-unforgeable interactive signature schemes that are not honest-user unforgeable.*

Proof. If complete, blind, and strongly unforgeable interactive signature schemes exist, then there is a complete, blind, strongly unforgeable, and randomized interactive signature scheme BS (e.g., by applying the transformation from Sect. 4.5). From BS we construct BS_2 as described at the beginning of this section. By Lemmas 12, 13, and 14, BS_2 is complete, blind, unforgeable, and strongly $\mathcal{S}+\mathcal{U}$-unforgeable, but not honest-user unforgeable. □

4.5. *From Unforgeability to Honest-User Unforgeability*

In this section, we show how to turn any unforgeable interactive signature scheme into an honest-user unforgeable one. Our transformation is extremely efficient as it only adds some randomness to the message. Therefore, it not only adds a negligible overhead to original scheme, but it also preserves all underlying assumptions. The construction is formally defined in Construction 1 and depicted in Fig. 2.

Construction 1. *Let $\mathsf{BS}' = (\mathsf{KG}', \langle \mathcal{S}', \mathcal{U}' \rangle, \mathsf{Vf}')$ be an interactive signature scheme and define the signature scheme BS through the following three procedures:*

Key Generation *The algorithm $\mathsf{KG}(1^\lambda)$ runs $(sk', pk') \leftarrow \mathsf{KG}'(1^\lambda)$ and returns this key pair.*

Signature Issue Protocol *The signer $\mathcal{S}$ is identical to the original signer $\mathcal{S}'$. The user $\mathcal{U}(pk, m)$ picks $r \xleftarrow{\$} \{0,1\}^\lambda$, sets $m' \leftarrow m\|r$, and invokes the original user $\mathcal{U}'(pk, m')$ who then interacts with the signer. When $\mathcal{U}'$ returns a signature σ, $\mathcal{U}$ computes $\sigma' \leftarrow (\sigma, r)$ and outputs σ'. (See also Fig. 2.)*

Signature Verification *The input of the verification algorithm* Vf *is a public key pk, a message m, and a signature $\sigma' = (\sigma, r)$. It sets $m' \leftarrow (m\|r)$ and returns the result of* $\mathsf{Vf}'(pk, m\|r, \sigma)$.

We first show that our transformation preserves completeness and blindness.

Lemma 16. *If* BS' *is a complete and blind interactive signature scheme, so is* BS.

Since the proof follows easily, we omit it here.

Now, we prove that our construction turns any unforgeable scheme into an honest-user unforgeable one.

Lemma 17. *If* BS' *is an unforgeable interactive signature scheme, then* BS *is honest-user unforgeable (Definition 5).*

Proof. Assume for the sake of contradiction that BS is not honest-user unforgeable. Then there exists an efficient adversary $\mathcal{A}$ that wins the honest-user unforgeability game with non-negligible probability. We then show how to build an attacker $\mathcal{B}$ that breaks the unforgeability of BS'.

The input of the algorithm $\mathcal{B}$ is a public key pk. It runs a black-box simulation of $\mathcal{A}$ and simulates the oracles as follows. Whenever $\mathcal{A}$ engages in an interactive signature issue protocol with the signer, i.e., when the algorithm $\mathcal{A}$ plays the role of the user, then $\mathcal{B}$ relays all messages between $\mathcal{A}$ and the signer. If $\mathcal{A}$ invokes the oracle $\mathcal{P}$ on a message m, then $\mathcal{B}$ picks a random $r \xleftarrow{\$} \{0,1\}^\lambda$, sets $m' \leftarrow m\|r$, and engages in an interactive signature issue protocol where $\mathcal{B}$ runs the honest-user algorithm $\mathcal{U}'$. At the end of this protocol, the algorithm $\mathcal{B}$ obtains a signature σ on the message m'. It sets $\sigma' \leftarrow (\sigma, r)$, stores the pair (m', σ') in a list L, and returns σ' together with the corresponding transcript trans to the attacker $\mathcal{A}$.

Eventually, the algorithm $\mathcal{A}$ stops, outputting a sequence of message/signature pairs $(m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*)$. In this case, $\mathcal{B}$ recovers all message/signature pairs $(m_1', \sigma_1'), \ldots, (m_n', \sigma_n')$ stored in L, it parses σ_i^* as $(\tilde{\sigma}_i, r_i')$, it sets $\widetilde{m}_i \leftarrow m_i^*\|r_i^*$ for all $i = 1, \ldots, k+1$ and outputs $(m_1', \sigma_1'), \ldots, (m_n', \sigma_n'), (\widetilde{m}_1, \widetilde{\sigma}_1), \ldots, (\widetilde{m}_{k+1}, \widetilde{\sigma}_{k+1})$.

Analysis For the analysis first observe that $\mathcal{B}$ runs in polynomial-time because $\mathcal{A}$ is efficient and because the handling of all queries can be done efficiently. Suppose that $\mathcal{A}$ succeeds with non-negligible probability. Then it outputs $(k+1)$ message/signature pairs that verify under Vf. Since $\mathcal{B}$ runs the honest-user algorithm to compute the signatures $\sigma_1', \ldots, \sigma_n'$, it follows (from the completeness) that all message/signature pairs that $\mathcal{B}$ returns, verify with overwhelming probability. It is left to show that (a) the algorithm $\mathcal{B}$ outputs one more message/signature pair (than queries to the signing oracle with output ok took place) and (b) all messages are distinct.

The distinctness property follows immediately from the definition of the success probability in the honest-user unforgeability game and from the construction. More precisely, consider the messages $(m'_1, \ldots, m'_n)$ and $(\widetilde{m}_1, \ldots, \widetilde{m}_{k+1})$, where $m'_i = m_i \| r_i$ and $\widetilde{m}_j = m^*_j \| r^*_j$. According to our assumption that $\mathcal{A}$ succeeds, it follows that all message pairs m^*_r and m^*_s (for all $r \neq s$) differ from each other. But then it follows easily that $\widetilde{m}^*_r$ and $\widetilde{m}^*_s$ are also distinct (for all $r \neq s$). Since the r_i are chosen randomly, the messages $(m'_1, \ldots, m'_n)$ also differ from each other with overwhelming probability. Now, consider the messages $(m_1, \ldots, m_n)$ that $\mathcal{A}$ sends to the oracle $\mathcal{P}$. Note that all these messages must differ from the messages $(m^*_1, \ldots, m^*_{k+1})$ returned by $\mathcal{A}$ by definition. This means, however, that $\widetilde{m}^*_r$ differs from m'_i for all i, r.

Finally, we have to show that $\mathcal{B}$ returns one more message/signature pair (property (a)) than protocol executions with the signer $\mathcal{S}'$ took place (and that produced output ok). Since $\mathcal{A}$ wins the game, it follows that in at most k of the protocol executions that $\mathcal{B}$ forwarded between $\mathcal{A}$ and its external signer, the signer returned ok. $\mathcal{B}$ itself has executed n user instances to simulate the oracle $\mathcal{P}$. Since $\mathcal{A}$ outputs $k+1$ message signature pairs (s.t. $m_i \neq m_j$ for all i, j), it follows that $\mathcal{B}$ has asked at most $n+k$ queries in which the signer $\mathcal{S}'$ returned ok, but $\mathcal{B}$ returned $n+k+1$ message/signature pairs. This, however, contradicts the assumption that BS is unforgeable. □

Putting together the above results, we get the following theorem.

Theorem 18. *If complete, blind, and unforgeable interactive signature schemes exist, then there are complete, blind, unforgeable, and honest-user unforgeable interactive signature schemes (with respect to Definition 5).*

The proof of this theorem follows directly from Lemmas 16 and 17.

5. Probabilistic Verification

In this section, we show that, if we allow for a probabilistic verification algorithm, both the definition of honest-user unforgeability and the usual definition of unforgeability will consider schemes to be secure that do not meet the intuitive notion of unforgeability.

One may argue that discussing problems in the definition of blind signature schemes in the case of probabilistic verification is not necessary because one can always just use schemes with deterministic verification. We disagree with this point of view: Without understanding why the definition is problematic in the case of probabilistic verification, there is no reason to restrict oneself to schemes with deterministic verification. Only the awareness of the problem allows us to circumvent it. We additionally give a definition that works in the case of probabilistic verification. This is less important than pointing out the flaws, since in most cases one can indeed use schemes with deterministic verification. But there might be (rare) cases where this is not possible (note that no generic transformation outside the random oracle model is known that makes the verification deterministic).

First, we give some intuition for our counterexample and formalize it afterward. Assume an interactive signature scheme BS_3 that distinguishes two kinds of signatures: a "full-signature" that will pass verification with probability 1, and a "half-signature" that

passes verification with probability $\frac{1}{2}$. An honest interaction between the signer $\mathcal{S}_3$ and the user $\mathcal{U}_3$ will always produce a full-signature. A malicious user, however, may interact with the signer to get half-signatures for arbitrary messages. Furthermore, the malicious user may, by sending λ half-signatures to the signer (λ is the security parameter) and performing a special interaction, get two (or more) further half-signatures instead of one. ("Buy $\lambda + 1$ signatures, get one free.") At the first glance, one would expect that such a scheme cannot be honest-user unforgeable or even unforgeable. But in fact, the adversary has essentially two options: First, he does not request λ half-signatures. Then he will not get a signature for free and thus will not win in the honest-user unforgeability game. Second, he does request λ half-signatures and then performs the extra-query and thus gets $\lambda + 2$ half-signatures using $\lambda + 1$ queries. Then, to win, he needs that all $\lambda + 2$ signatures pass verification (since the definition of unforgeability/honest-user unforgeability requires that $\mathsf{Vf}_3(pk, m_i^*, \sigma_i^*)$ evaluates to 1 for all signatures (m_i^*, σ_i^*) output by the adversary). However, since each half-signature passes verification with probability $\frac{1}{2}$, the probability that all signatures pass verification is negligible ($< 2^{-\lambda}$). Thus, the adversary does not win, and the scheme is honest-user unforgeable. Clearly, this is not what one would expect; so Definition 5 should not be applied to the case where the verification is probabilistic (and similarly the normal definition of unforgeability should not be applied either in that case).

More precisely, let $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Vf})$ be a randomized, complete, blind, and honest-user unforgeable interactive signature scheme with deterministic verification. Let Q be an efficiently decidable set such that the computation of arbitrarily many bitstrings $m \in Q$ and $m' \notin Q$ is efficiently feasible.

We define the scheme $\mathsf{BS}_3 = (\mathsf{KG}_3, \langle \mathcal{S}_3, \mathcal{U}_3 \rangle, \mathsf{Vf}_3)$ as follows:

- $\mathsf{KG}_3 := \mathsf{KG}$.
- $\mathcal{S}_3(sk)$ behaves like $\mathcal{S}(sk)$, except when the first message from the user is of the form $(\texttt{extrasig}, m_1^\circ, \ldots, m_\lambda^\circ, \sigma_1^\circ, \ldots, \sigma_\lambda^\circ, m_1', \ldots, m_q')$ where λ is the security parameter. Then $\mathcal{S}_3$ executes the following steps:
  - Check whether $m_1^\circ, \ldots, m_\lambda^\circ \in Q$ are pairwise distinct messages, and for all $i = 1, \ldots, q$ we have $m_i' \notin Q$, and for all $i = 1, \ldots, \lambda$ we have $\mathsf{Vf}(pk, 1\|m_i^\circ, \sigma_i^\circ) = 1$.[3] If not, ignore the message.
  - If the check passes, run $\langle \mathcal{S}(sk), \mathcal{U}(pk, 1\|m_i') \rangle$ for each $i = 1, \ldots, q$, resulting in signatures $\tilde{\sigma}_i$, and set $\sigma_i' := 1\|\tilde{\sigma}_i$.
    Then $\mathcal{S}_3$ sends $(\sigma_1', \ldots, \sigma_n')$ to the user, outputs ok, and does not react to any further messages in this session.
- $\mathcal{U}_3(pk, m)$ runs $\sigma \leftarrow \mathcal{U}(pk, 0\|m)$ and returns $0\|\sigma$.
- $\mathsf{Vf}_3(pk, m, \sigma)$ performs the following steps:
  - If $\sigma = 0\|\sigma'$ and $\mathsf{Vf}(pk, 0\|m, \sigma') = 1$, Vf_3 returns 1.
  - If $\sigma = 1\|\sigma'$ and $\mathsf{Vf}(pk, 1\|m, \sigma) = 1$, Vf_3 returns 1 with probability $p := \frac{1}{2}$ and 0 with probability $1 - p$.
  - Otherwise, Vf_3 returns 0.

[3] Without loss of generality, we assume that the public key pk can efficiently be computed from the secret key sk.

Thus here a signature $0\|\sigma$ where σ signs $0\|m$ is a full-signature, and a signature $1\|\sigma$ where σ signs $1\|m$ is a half-signature.

Lemma 19. *If* BS *is blind and complete, so is* BS_3.

Proof. Blindness and completeness of BS_3 follow directly from that of BS. The only difference between the schemes is that instead of a message m, a message $0\|m$ is signed and 0 is prepended to the signatures (as long as the user is honest as is the case in the definitions of blindness and completeness). □

Lemma 20. *If* BS *is honest-user unforgeable, so is* BS_3.

Proof. We first fix some notation. A pair (m, σ) is

- a *full-signature* if $\sigma = 0\|\sigma'$ and $\mathsf{Vf}(pk, 0\|m, \sigma') = 1$;
- a *half-signature* if $\sigma = 1\|\sigma'$ and $\mathsf{Vf}(pk, 1\|m, \sigma') = 1$;
- and a *non-signature* otherwise.

Note that if (m, σ) is a full-, half-, or non-signature, then $\mathsf{Vf}_3(pk, m, \sigma)$ is 1, $p = \frac{1}{2}$, or 0, respectively. An interaction between $\mathcal{A}$ and $\mathcal{S}_3$ that begins with a $(\texttt{extrasig}, \ldots)$-message passing the check in the definition of $\mathcal{S}_3$ is called an extra-query. Other interactions between $\mathcal{A}$ and $\mathcal{S}_3$ that lead to an output ok from $\mathcal{S}_3$ are called standard queries.

Fix an efficient adversary $\mathcal{A}$ against the honest-user unforgeability game for BS_3. Without loss of generality, we assume that the output of $\mathcal{A}$ is always of the form $((m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*))$ for some k. Let k_e denote the number of extra-queries and k_s the number of standard queries performed by $\mathcal{A}$. Let $m_1, \ldots, m_n$ be the messages queried by $\mathcal{A}$ to the oracle $\mathcal{P}$ (which simulates $\langle \mathcal{S}_3, \mathcal{U}_3 \rangle$), and let $\sigma_1, \ldots, \sigma_n$ be the answers from $\mathcal{P}$. In an execution of the game, we distinguish the following cases:

(i) $k_e + k_s > k$, or for some i, (m_i^*, σ_i^*) is a non-signature, or for some $i \neq j$, $m_i^* = m_j^*$, or for some i, j, $m_i^* = m_j$.
(ii) For $h > \lambda$ different indices i, (m_i^*, σ_i^*) is a half-signature. And (i) does not hold.
(iii) No extra-query was performed. And (i), (ii) do not hold.
(iv) All other cases, i.e., (i), (ii), (iii) do not hold.

In case (i), by definition, the adversary does not win.

In case (ii), the probability that $\mathsf{Vf}_3(pk, m_i^*, \sigma_i^*) = 1$ for all i is upper-bounded by the probability that $\mathsf{Vf}_3(pk, m, \sigma) = 1$ for all half-signatures (m, σ) output by $\mathcal{A}$. That probability, in turn, is bounded by $p^h = 2^{-h} \leq 2^{-\lambda}$ because each invocation of $\mathsf{Vf}(pk, m, \sigma)$ succeeds with probability p for a half-signature (m, σ). Thus the adversary wins with negligible probability in case (ii).

Hence $\mathcal{A}$ only wins with non-negligible probability, if either case (iii) or (iv) occurs with non-negligible probability.

Assume that case (iii) happens with non-negligible probability, and observe that any full- or half-signature on a message m can be efficiently transformed into a signature on $0\|m$ or $1\|m$, respectively (with respect to the original scheme BS). We construct an adversary $\mathcal{B}$ against the honest-user unforgeability game for the original scheme BS. $\mathcal{B}$ runs a black-box simulation of $\mathcal{A}$ and behaves as follows: Whenever $\mathcal{A}$ performs an

extra-query, then $\mathcal{B}$ aborts. If $\mathcal{A}$ queries $\sigma_i \leftarrow \mathcal{P}(m_i)$, then $\mathcal{B}$ sets $m_i' = 0\|m_i$ and sends m_i' to its own oracle $\mathcal{P}$ (which simulates $\langle \mathcal{S}, \mathcal{U} \rangle$); it then obtains a signature σ_i and returns $0\|\sigma_i$ to $\mathcal{A}$. Whenever $\mathcal{A}$ queries the signer directly, then $\mathcal{B}$ forwards all messages in both directions.

When $\mathcal{A}$ outputs $((m_1^*, b_1^*\|\sigma_1^*), \ldots, (m_{k+1}^*, b_{k+1}^*\|\sigma_{k+1}^*))$ with $b_i^* \in \{0,1\}$, then the algorithm $\mathcal{B}$ outputs $((b_1^*\|m_1^*, \sigma_1^*), \ldots, (b_{k+1}^*\|m_{k+1}^*, \sigma_{k+1}^*))$. Obviously, if all m_i^* are distinct and different from all m_i, then all $b_i^*\|m_i^*$ are distinct and different from all $0\|m_i$. And if $\mathsf{Vf}'(m_i^*, b_i^*\|\sigma_i^*) = 1$, then $\mathsf{Vf}(b_i^*\|m_i^*, \sigma_i^*) = 1$. Thus, when (iii) occurs with non-negligible probability in the honest-user unforgeability game with $\mathcal{A}$ and BS_3, then $\mathcal{B}$ wins with non-negligible probability in the honest-user unforgeability game with BS. By assumption, BS is honest-user unforgeable, so we have a contradiction. Thus our assumption that case (iii) occurs with non-negligible probability was false. Hence case (iii) occurs with negligible probability.

Now assume that case (iv) occurs with non-negligible probability. In this case, let Σ_f be the set of all full-signatures output by $\mathcal{A}$. Note that this is not the set of all $k+1$ signatures output by $\mathcal{A}$ because $\mathcal{A}$ may also output half-signatures. Let Σ_h denote the set of all half-signatures used in the first extra-query. More precisely, $(m, \sigma) \in \Sigma_h$ iff the first extra-query was of the form $(\texttt{extrasig}, m_1^\circ, \ldots, m_\lambda^\circ, \sigma_1^\circ, \ldots, \sigma_\lambda^\circ, m_1', \ldots, m_q')$ with $(m, \sigma) = (m_i^\circ, \sigma_i^\circ)$ for some i. Let Σ_e denote the half-signatures returned by extra-queries, i.e., $(m', \sigma') \in \Sigma_e$ iff an extra-query $(\texttt{extrasig}, m_1^\circ, \ldots, m_\lambda^\circ, \sigma_1^\circ, \ldots, \sigma_\lambda^\circ, m_1', \ldots, m_q')$ was answered with $(\sigma_1', \ldots, \sigma_q')$ such that $(m', \sigma') = (m_i', \sigma_i')$ for some i. Let Σ_u be the set of all signatures received from the oracle $\mathcal{P}$, i.e., $\Sigma_u = \{(m_1, \sigma_1), \ldots, (m_n, \sigma_n)\}$. By completeness, with overwhelming probability Σ_u contains only full-signatures. Let ℓ be the number of half-signatures in the output of $\mathcal{A}$. We have $\ell \leq \lambda$ since otherwise we would be in case (ii).

Given a set Σ of pairs of messages and signatures, let Σ^* denote the set $\Sigma^* := \{(b\|m, \sigma') : (m, b\|\sigma') \in \Sigma, b \in \{0,1\}\}$.

Since the messages in Σ_f are distinct, and the messages in Σ_h are distinct, and Σ_f contains only full-signatures, and Σ_h contains only half-signatures, we have that all messages in $\Sigma_f^* \cup \Sigma_h^*$ are distinct, that $|\Sigma_f^* \cup \Sigma_h^*| = |\Sigma_f| + |\Sigma_h| \geq (k+1-\ell)+\lambda \geq k+1$, and that all $(m, \sigma) \in \Sigma_f^* \cup \Sigma_h^*$ satisfy $\mathsf{Vf}(pk, m, \sigma) = 1$.

Furthermore, the messages in Σ_h^* are different from those in Σ_u^* because Σ_h contains only half- and Σ_u only full-signatures. The messages in Σ_f^* are different from those in Σ_u^* because the messages in Σ_f are different from those in Σ_u (otherwise we would be in case (i)). The messages in Σ_h^* are different from those in Σ_e^* since by definition of extra-queries, the messages in Σ_h are in Q while the messages in Σ_e are not in Q. The messages in Σ_f^* are different from those in Σ_e^* because Σ_f contains only full- and Σ_e only half-signatures. Thus, the messages in $\Sigma_f^* \cup \Sigma_h^*$ are different from the messages in $\Sigma_u^* \cup \Sigma_e^*$.

Summarizing, in case (iv), we have $|\Sigma_f^* \cup \Sigma_h^*| \geq k+1$, the messages in $\Sigma_f^* \cup \Sigma_h^*$ are pairwise distinct and different from the messages in $\Sigma_u^* \cup \Sigma_e^*$, and all $(m, \sigma) \in \Sigma_f^* \cup \Sigma_h^*$ satisfy $\mathsf{Vf}(pk, m, \sigma) = 1$.

We then construct an adversary $\mathcal{B}$ against the original scheme BS. The attacker $\mathcal{B}$ simulates $\mathcal{A}$ with the following modifications. When $\mathcal{A}$ queries the oracle $\mathcal{P}$ on a message m_i, then $\mathcal{B}$ invokes its external oracle $\mathcal{P}$ (which simulates $\langle \mathcal{S}, \mathcal{U} \rangle$) on

input $(0\|m_i)$, gets an answer σ'_i, and returns $\sigma_i := 0\|\sigma'_i$ to $\mathcal{A}$. If $\mathcal{A}$ performs an extra-query $(\texttt{extrasig}, \ldots, m'_1, \ldots, m'_q)$, then $\mathcal{B}$ answers with $(\sigma'_1, \ldots, \sigma'_q) := (1\|\mathcal{P}(1\|m'_1), \ldots, 1\|\mathcal{P}(1\|m'_q))$ instead. Suppose that $\mathcal{A}$ outputs a message/signature sequence, then $\mathcal{B}$ computes the sets Σ^*_u, Σ^*_h, Σ^*_f, and Σ^*_e instead and outputs the message/signature pairs contained in the set $\Sigma^*_f \cup \Sigma^*_h$. Notice that $\mathcal{B}$ only queries messages from $\mathcal{P}$ that are in the set $\Sigma^*_u \cup \Sigma^*_e$. If (iv) occurs with non-negligible probability, then we have an adversary $\mathcal{B}$ that, with non-negligible probability, outputs at least $k+1$ message/signature pairs (m, σ) that are valid (i.e., $\mathsf{Vf}(pk, m, \sigma) = 1$), that are pairwise distinct, and that also differ from all message queried to $\mathcal{P}$. Thus, $\mathcal{B}$ breaks the honest-user unforgeability of BS. Since by assumption, BS is honest-user unforgeable, our assumption that case (iv) occurs with non-negligible probability was false.

Summing up, we have shown that both case (iii) and case (iv) happen only with negligible probability. Since in cases (i) and (ii), the adversary $\mathcal{A}$ wins only with negligible probability, it follows that overall, $\mathcal{A}$ wins only with negligible probability. Since this holds for any adversary $\mathcal{A}$, BS_3 is honest-user unforgeable. □

The following lemma shows that, although BS_3 is honest-user unforgeable (and thus also unforgeable) according to the definitions of these notions, it should not be considered secure. Namely, an adversary can, given λ queries, produce $\lambda + 1$ message/signature pairs, each of which passes verification with probability $\frac{1}{2}$. In particular in a setting where the machine which verifies the signatures is stateless and where the adversary may thus just resubmit a rejected signature, such signatures are as good as signatures that pass verification with probability 1. Thus, the adversary has essentially forged one signature.

Lemma 21. *We call (m, σ) a half-signature (with respect to some implicit public key pk) if the probability that $\mathsf{Vf}(pk, m, \sigma) = 1$ is $1/2$. If BS is complete, then for any polynomial p, there is an adversary $\mathcal{A}$ that performs $\lambda + 1$ interactions with $\mathcal{S}_3$ and does not query $\mathcal{P}$ and that, with overwhelming probability, outputs $p(\lambda)$ half-signatures $(m^*_1, \sigma^*_1), \ldots, (m^*_{p(\lambda)}, \sigma^*_{p(\lambda)})$ such that all m^*_i are distinct.*

Proof. The adversary $\mathcal{A}$ that performs λ interactions with $\mathcal{S}_3$ and that never queries $\mathcal{P}$ works as follows. It picks λ distinct messages $m^\circ_1, \ldots, m^\circ_\lambda$ from Q and chooses $p(\lambda)$ additional distinct messages $m'_j \notin Q$. It then queries the signer sequentially on the message $1\|m^\circ_i$ and obtains the corresponding signature σ°_i for $i = 1, \ldots, \lambda$. Since BS is complete, with overwhelming probability the $(m^\circ_i, \sigma^\circ_i)$ are half-signatures. Afterward, the adversary $\mathcal{A}$ initiates another signature issue protocol session with the signer and sends as the first message: $(\texttt{extrasig}, m^\circ_1, \ldots, m^\circ_\lambda, \sigma^\circ_1, \ldots, \sigma^\circ_\lambda, m^*_1, \ldots, m^*_{p(\lambda)})$. The signer answers with signatures $\sigma^*_1, \ldots, \sigma^*_{p(\lambda)}$. Since BS is complete, with overwhelming probability the (m^*_i, σ^*_i) are half-signatures.

Finally, $\mathcal{A}$ stops, outputting $(m^*_1, \sigma^*_1), \ldots, (m^*_{p(\lambda)}, \sigma^*_{p(\lambda)})$.

Thus $\mathcal{A}$ outputs $p(\lambda)$ half-signatures while performing only $\lambda + 1$ queries. □

5.1. *Adapting the Definition*

We have shown that, if we allow for a probabilistic verification algorithm in the definition of honest-user unforgeability (and similarly in the definition of unforgeability), schemes that are intuitively insecure will be considered secure by the definition. There are two possible ways to cope with this problem.

The simplest solution is to require that the verification algorithm is deterministic. This is what we did in Sect. 4.1 (Definition 5). This choice is justified since almost all known blind signature schemes have a deterministic verification algorithm anyway. Thus restricting the verification algorithm to be deterministic may be preferred in order to obtain a simpler definition.[4]

In some cases, however, it might not be possible to make the verification deterministic. In such cases, it is necessary to strengthen the definition of honest-user unforgeability. Looking back at our counterexample, the problem was the following: If the adversary produces many signatures that each pass verification with non-negligible but not overwhelming probability, this is not considered an attack: The probability that all signatures pass verification simultaneously is negligible. In order to fix this problem, we thus need to change the definition in such a way that a signature that is accepted with non-negligible probability is always considered a successful forgery. More precisely, if a signature passes verification at least once when running the verification algorithm a polynomial number of times, then the signature is considered valid. This idea leads to the following definition:

Definition 22. (*Honest-user unforgeability with probabilistic verification*) Given a probabilistic algorithm Vf and an integer t, we define Vf^t as follows: $\mathsf{Vf}^t(pk, m, \sigma)$ runs $\mathsf{Vf}(pk, m, \sigma)$ t times. If one of the invocations of Vf returns 1, Vf^t returns 1. If all invocations of Vf return 0, Vf^t returns 0.

A blind signature scheme $\mathsf{BS} = (\mathsf{KG}, \langle \mathcal{S}, \mathcal{U} \rangle, \mathsf{Vf})$ is called *honest-user unforgeable (with probabilistic verification)* if the following holds: For any efficient algorithm $\mathcal{A}$ and any polynomial p, the probability that experiment $\mathsf{HUnforge}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$ evaluates to 1 is negligible (as a function of λ) where

Experiment $\mathsf{HUnforge}^{\mathsf{BS}}_{\mathcal{A}}(\lambda)$
$(sk, pk) \leftarrow \mathsf{KG}(1^\lambda)$
$((m_1^*, \sigma_1^*), \ldots, (m_{k+1}^*, \sigma_{k+1}^*)) \leftarrow \mathcal{A}^{\langle \mathcal{S}(sk), \cdot \rangle^\infty, \mathcal{P}(sk, pk, \cdot)}(pk)$
Let $m_1, \ldots, m_n$ be the messages queried to $\mathcal{P}(sk, pk, \cdot)$.
Return 1 iff
$m_i^* \neq m_j$ for all i, j
$m_i^* \neq m_j^*$ for all i, j with $i \neq j$, and
$\mathsf{Vf}^{p(\lambda)}(pk, m_i^*, \sigma_i^*) = 1$ for all i, and
$\mathcal{S}$ has returned ok in at most k interactions.

(When counting the interactions in which $\mathcal{S}$ returns ok, we do not count the interactions simulated by $\mathcal{P}$.)

[4] Notice that one could weaken the requirement and only require that two invocations of the verification algorithm output the same value with overwhelming probability.

Notice that the only difference to Definition 5 is that we additionally quantify over a polynomial p and use $\mathsf{Vf}^{p(\lambda)}$ instead of Vf. If a signature is accepted with non-negligible probability, then there is a polynomial p such that $\mathsf{Vf}^{p(\lambda)}$ will accept that signature with overwhelming probability. (For our counterexample BS_3, one can choose $p(\lambda) := \lambda$ to show that it does not satisfy Definition 22.)

Notice that there is no obvious transformation for taking a signature scheme satisfying the regular unforgeability definition and constructing a scheme secure with respect to Definition 22 out of it. One obvious approach would be to include the randomness for verification in the message and thus to make the scheme deterministic. This might, however, make the scheme totally insecure because in this case a forger might include just the right randomness to get a signature accepted (if that signature would be accepted with negligible but nonzero probability otherwise). Another obvious approach would be to change the verification algorithm such that it verifies each signature p times (for a suitable polynomial p) and only accepts when all verifications succeed. This would make, e.g., half-signatures into signatures with negligible acceptance probability. But also this approach does not work in general: For any p, the adversary might be able to produce signatures that fail each individual verification with probability $1/2p$ and thus pass the overall verification with constant probability.

6. Conclusion and Open Problems

We revisited the well-established definition of unforgeability proposed by Pointcheval and Stern (Journal of Cryptology, 2000). Our results show that the original unforgeability definition does not exclude that an adversary verifiably uses the same message m for signing twice and is then still able to produce another signature for a new message $m' \neq m$. Intuitively, this should not be possible; yet, it is not captured in the original definition, because the number of signatures equals the number of requests. To handle these types of attacks, we proposed a stronger notions, called honest-user unforgeability, and we gave a simple and efficient transformation that turns any unforgeable blind signature scheme (with deterministic verification) into an honest-user unforgeable one. We also discussed the problem of defining blind signatures with probabilistic verification. The main observation is that if we allow for a probabilistic verification algorithm, both the definition of honest-user unforgeability and the usual definition of unforgeability will consider schemes to be secure that do not meet the intuitive notion of unforgeability.

Since we do not propose a generic transformation that makes schemes with probabilistic verification secure according to our definition, it would be interesting to see wether such a transformation exists. Alternatively, an impossibility result would also improve our understanding in this area.

Furthermore, it is an interesting question whether existing, not strongly unforgeable, blind signature schemes in the literature (e.g., [2,17,22,23,26,32]) are already honest-user unforgeable (so that our transformation would not have to be applied in those cases).

References

[1] M. Abe, A secure three-move blind signature scheme for polynomially many signatures, in Birgit Pfitzmann, editor, *Advances in Cryptology—EUROCRYPT 2001*, vol. 2045 of *Lecture Notes in Computer Science*, Innsbruck, Austria, May 6–10 (Springer, Berlin, 2001), pp. 136–151

[2] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. Ohkubo, Structure-preserving signatures and commitments to group elements, in Tal Rabin, editor, *Advances in Cryptology—CRYPTO 2010*, vol. 6223 of *Lecture Notes in Computer Science*, Santa Barbara, CA, USA, August 15–19, (Springer, Berlin, 2010), pp. 209–236

[3] M. Abdalla, C. Namprempre, G. Neven, On the (im)possibility of blind message authentication codes, in David Pointcheval, editor, *Topics in Cryptology—CT-RSA 2006*, vol. 3860 of *Lecture Notes in Computer Science*, San Jose, CA, USA, February 13–17, (Springer, Berlin, 2006), pp. 262–279

[4] M. Abe, M. Ohkubo, A framework for universally composable non-committing blind signatures, in Mitsuru Matsui, editor, *Advances in Cryptology—ASIACRYPT 2009*, vol. 5912 of *Lecture Notes in Computer Science*, Tokyo, Japan, December 6–10, (Springer, Berlin, 2009), pp. 435–450

[5] N. Asokan, V. Shoup, M. Waidner, Optimistic fair exchange of digital signatures (extended abstract), in Kaisa Nyberg, editor, *Advances in Cryptology—EUROCRYPT'98*, vol. 1403 of *Lecture Notes in Computer Science*, Espoo, Finland, May 31 – June 4, (Springer, Berlin, 1998), pp. 591–606

[6] R. Bjones, U-prove technology overview. http://www.itforum.dk/downloads/Ronny_Bjones_Uprove.pdf, (2010)

[7] M. Bellare, C. Namprempre, D. Pointcheval, M. Semanko, The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. *J. Cryptol.*, **16**(3):185–215 (2003)

[8] A. Boldyreva, Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme, in Yvo Desmedt, editor, *PKC 2003: 6th International Workshop on Theory and Practice in Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, Miami, USA, January 6–8, (Springer, Berlin, 2003), pp. 31–46

[9] S. Brands, C. Paquin, U-prove cryptographic specification v1.0. http://connect.microsoft.com/site642/Downloads/DownloadDetails.aspx?DownloadID=26953, March 2011

[10] S.A. Brands, *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*, (MIT Press, Cambridge, 2000)

[11] J. Camenisch, T. Groß, Efficient attributes for anonymous credentials, in Peng Ning, Paul F. Syverson, Somesh Jha, editors, *ACM CCS 08: 15th Conference on Computer and Communications Security*, Alexandria, Virginia, USA, October 27–31, (ACM Press, New York, 2008), pp. 345–356

[12] D. Chaum, Blind signatures for untraceable payments, in David Chaum, Ronald L. Rivest, Alan T. Sherman, editors, *Advances in Cryptology—CRYPTO'82*, Santa Barbara, CA, USA, (Plenum Press, New York, 1982), pp. 199–203

[13] D. Chaum, Blind signature system, in David Chaum, editor, *Advances in Cryptology—CRYPTO'83*, Santa Barbara, CA, USA, (Plenum Press, New York, 1983), p. 153

[14] J. Camenisch, M. Koprowski, B. Warinschi, Efficient blind signatures without random oracles, in Carlo Blundo and Stelvio Cimato, editors, *SCN 04: 4th International Conference on Security in Communication Networks*, vol. 3352 of *Lecture Notes in Computer Science*, Amalfi, Italy, September 8–10, (Springer, Berlin, 2004), pp. 134–148

[15] J. Camenisch, G, Neven, A. Shelat, Simulatable adaptive oblivious transfer, in Moni Naor, editor, *Advances in Cryptology—EUROCRYPT 2007*, vol. 4515 of *Lecture Notes in Computer Science*, Barcelona, Spain, May 20–24, (Springer, Berlin, 2007), pp. 573–590

[16] M. Fischlin, Round-optimal composable blind signatures in the common reference string model, in Cynthia Dwork, editor, *Advances in Cryptology—CRYPTO 2006*, vol. 4117 of *Lecture Notes in Computer Science*, Santa Barbara, CA, USA, August 20–24, (Springer, Berlin, 2006), pp. 60–77

[17] M. Fischlin, D. Schröder, Security of blind signatures under aborts, in Stanislaw Jarecki and Gene Tsudik, editors, *PKC 2009: 12th International Conference on Theory and Practice of Public Key Cryptography*, vol. 5443 of *Lecture Notes in Computer Science*, Irvine, CA, USA, March 18–20, (Springer, Berlin, 2009), pp. 297–316

[18] M. Fischlin, D. Schröder, On the impossibility of three-move blind signature schemes, in Henri Gilbert, editor, *Advances in Cryptology—EUROCRYPT 2010*, vol. 6110 of *Lecture Notes in Computer Science*, French Riviera, May 30 – June 3, (Springer, Berlin, 2010), pp. 197–215

[19] G. Fuchsbauer, Automorphic signatures in bilinear groups and an application to round-optimal blind signatures. Cryptology ePrint Archive, Report 2009/320, 2009. http://eprint.iacr.org/.

[20] S. Garg, D. Gupta, Efficient round optimal blind signatures, in Phong Q. Nguyen and Elisabeth Oswald, editors, *Advances in Cryptology—EUROCRYPT 2014*, vol. 8441 of *Lecture Notes in Computer Science*, Copenhagen, Denmark, May 11–15, (Springer, Berlin, 2014), pp. 477–495

[21] J.A. Garay, P.D. MacKenzie, M. Prabhakaran, K. Yang, Resource fairness and composability of cryptographic protocols, in Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, vol. 3876 of *Lecture Notes in Computer Science*, New York, NY, USA, March 4–7, (Springer, Berlin, 2006) pp. 404–428

[22] S. Garg, V. Rao, A. Sahai, D. Schröder, D. Unruh, Round optimal blind signatures, in *Advances in Cryptology—CRYPTO 2011—31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14–18, 2011. Proceedings*, vol. 6841 of *Lecture Notes in Computer Science*, (Springer, Berlin, 2011), pp. 630–648

[23] E. Ghadafi, N.P. Smart, Efficient two-move blind signatures in the common reference string model. Cryptology ePrint Archive, Report 2010/568, 2010. http://eprint.iacr.org/

[24] O. Goldreich, *The Foundations of Cryptography*, vol. 2 (Cambridge University Press, New York, NY, 2004).

[25] O. Horvitz, J. Katz, Universally-composable two-party computation in two rounds, in Alfred Menezes, editor, *Advances in Cryptology—CRYPTO 2007*, vol. 4622 of *Lecture Notes in Computer Science*, Santa Barbara, CA, USA, August 19–23, (Springer, Berlin, 2007), pp. 111–129

[26] C. Hazay, J. Katz, C.-Y. Koo, Y. Lindell, Concurrently-secure blind signatures without random oracles or setup assumptions, in Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, vol. 4392 of *Lecture Notes in Computer Science*, Amsterdam, The Netherlands, February 21–24, (Springer, Berlin, 2007), pp. 323–341

[27] A. Juels, M. Luby, R. Ostrovsky, Security of blind digital signatures (extended abstract), in Burton S. Kaliski Jr., editor, *Advances in Cryptology—CRYPTO'97*, vol. 1294 of *Lecture Notes in Computer Science*, Santa Barbara, CA, USA, August 17–21, (Springer, Berlin, 1997), pp. 150–164

[28] A. Kiayias, H.-S. Zhou, Equivocal blind signatures and adaptive UC-security, in Ran Canetti, editor, *TCC 2008: 5th Theory of Cryptography Conference*, vol. 4948 of *Lecture Notes in Computer Science*, San Francisco, CA, USA, March 19–21, (Springer, Berlin, 2008), pp. 340–355

[29] MICROSOFT U-PROVE. Microsoft u-prove ctp release 2. http://connect.microsoft.com/site642/Downloads/DownloadDetails.aspx?DownloadID=26953, March 2011

[30] T. Okamoto, Efficient blind and partially blind signatures without random oracle, in Shai Halevi and Tal Rabin, editors, *TCC 2006: 3rd Theory of Cryptography Conference*, vol. 3876 of *Lecture Notes in Computer Science*, New York, NY, USA, March 4–7, (Springer, Berlin, 2006), pp. 80–99

[31] D. Pointcheval, J. Stern, Security arguments for digital signatures and blind signatures. *J. Cryptol.*, 13(3):361–396 (2000)

[32] M. Rückert, Lattice-based blind signatures, in Masayuki Abe, editor, *Advances in Cryptology—ASIACRYPT 2010*, vol. 6477 of *Lecture Notes in Computer Science*, Singapore, December 5–9, (Springer, Berlin, 2010), pp. 413–430

[33] D. Schröder, D. Unruh, Round optimal blind signatures. Cryptology ePrint Archive, Report 2011/264, 2011. http://eprint.iacr.org/