Non-interactive Blind Signatures for Random Messages

Lucjan Hanzlik(✉)

CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
hanzlik@cispa.de

Abstract. Blind signatures allow a signer to issue signatures on messages chosen by the signature recipient. The main property is that the recipient's message is hidden from the signer. There are many applications, including Chaum's e-cash system and Privacy Pass, where no special distribution of the signed message is required, and the message can be random. Interestingly, existing notions do not consider this practical use case separately. In this paper, we show that constraining the recipient's choice over the message distribution spawns a surprising new primitive that improves the well-established state-of-the-art. We formalize this concept by introducing the notion of non-interactive blind signatures (NIBS). Informally, the signer can create a presignature with a specific recipient in mind, identifiable via a public key. The recipient can use her secret key to finalize it and receive a blind signature on a random message determined by the finalization process. The key idea is that online interaction between the signer and recipient is unnecessary. We show an efficient instantiation of NIBS in the random oracle model from signatures on equivalence classes. The exciting part is that, in this case, for the recipient's public key, we can use preexisting keys for Schnorr, ECDSA signatures, El-Gamal encryption scheme or even the Diffie-Hellman key exchange. Reusing preexisting public keys allows us to distribute anonymous tokens similarly to cryptocurrency airdropping. Additional contributions include tagged non-interactive blind signatures (TNIBS) and their efficient instantiation. A generic construction in the random oracle or common reference string model based on verifiable random functions, standard signatures, and non-interactive proof systems.

Keywords: Blind Signatures · Non-Interactive Scheme · Random Oracle Model · Signatures on Equivalence Classes

1 Introduction

Blind signatures are a cryptographic primitive introduced by David Chaum [16]. Contrary to standard digital signature schemes, the signing process is an interactive protocol between two parties: the signer and the user (also called the recipient). The main property of blind signature schemes is like the name suggests *blindness*. It ensures that the signer does not 'see' the signed message. Blind

C. Hazay and M. Stam (Eds.): EUROCRYPT 2023, LNCS 14008, pp. 722–752, 2023.
https://doi.org/10.1007/978-3-031-30589-4_25

signatures also require *one-more unforgeability*, where an adversary has access to a signing oracle and must return one-more message-signature pair than the number of queries made.

Blind signature schemes find many applications. In Chaum's seminal work [16], it was shown how to use a blind signature for electronic cash (or e-cash), the forerunner of modern cryptocurrencies. The design of e-cash inspired many follow-up work [4,12,21,37]. The idea is elegant but simple. The bank issues e-cash as signatures on random identifiers chosen by users. To spend, the user shows the identifier (i.e., message) and the corresponding signature to the merchant, who can collect the amount from the bank. The bank keeps a list of 'used' identifiers to prevent double-spending. To make transactions unlinkable, the bank uses a blind signature to create the signature together with the user.

E-cash implements the idea of a single-use unlinkable electronic coins (e-coins), coupons or tokens. This interpretation inspired recent advances. Tumblebit [31] is a cryptographic tumbler that uses blind signatures as a building block. It introduces an intermediary party that issues a single-use coupon in exchange for cryptocurrency. The payer can send this coupon to another user who can redeem it for cryptocurrency. To ensure unlinkability, the intermediary issues several coupons in a given interval, creating an anonymity set. Blindness provides the property that any combination of sender/recipient is equally likely for the given anonymity set. This idea was also used in Privacy Pass [17], which relies on single-use coupons to make web browsing using anonymous networks more user-friendly. One of the use cases of Privacy Pass allows users to redeem coupons while omitting the increased number of CAPTCHAs that service providers challenge when using anonymous networks. Privacy Pass was recently extended with a rate-limiting version [32] also called Private Access Tokens. One of the main changes is that the coupon is not created on a random message but rather on a challenge created by the server. The challenge is to ensure that malicious users do not hoard coupons to circumvent daily limits. Blind signatures are used differently in voting schemes [14] and anonymous credentials (AC) [6,23]. Voters can get a signed receipt without revealing their vote. In AC, blind signatures provide means for issuing unlinkable credentials. Contrary to the e-cash application, the signed message in those applications is not a random identifier.

We observe that the user can randomly choose the blindly signed message in many applications, and the selected message does not need to be from a specific distribution. Following this, we consider the following research question.

Can we use this observation to move research on blind signatures forward?

Prior Work on Blind Signatures

In his seminal work [16], David Chaum introduced the idea of blind signatures. He defined a signing function s that commutes with a function c. The user can now send $c(\mathsf{m})$ to the signer, who signs this blinded message and returns $s(c(\mathsf{m}))$. Because of the commutativeness property, the user can extract $s(\mathsf{m})$ using the inverse function c^{-1} without the signer learning anything about the signed message m. It is worth noting that the user can always pick what gets

Table 1. Comparison between two-move blind signatures, Privacy Pass, and our NIBS and TNIBS. n denotes the number of signatures/tokens issued concurrently. All results are given in bits and refer to a 3072-bit RSA modulus, BLS12-381 [11] parameters for pairing-based schemes and a standard 256-bit elliptic curve for Privacy Pass. We assume that messages/nonces are 128-bit long. We indicate if the first message of user U can be reused by signer S to issue new tokens.

Scheme	Communication complexity		$\lvert\mathsf{m}\rvert + \lvert\mathsf{sig}\rvert$	$\lvert\mathsf{pk}\rvert$	Reusable 1st msg.	Security
	$\mathsf{U} \to \mathsf{S}$	$\mathsf{S} \to \mathsf{U}$				
Blind BLS [10]	$n \times 382$	$n \times 382$	510	3072	✗	ROM
Privacy Pass [17]	$n \times 257$	$n \times 257 + 512$	385	257	✗	ROM
RSA (PAT) [16,35]	$n \times 3072$	$n \times 3072$	3200	3072	✗	ROM
NIBS	$382^{\dagger}$	$n \times 1655$	1909	1526	✓	ROM+GGM
TNIBS	$382^{\dagger}$	$n \times 2546$	2800	1526	✓	ROM+GGM
Our Generic	$O(1)^{\dagger}$	$O(n)$			✓	ROM or CRS

†– The recipient's public key must be sent to the signer. There is no cost if a PKI is available.

signed. The main security properties defined by Chaum are that the inverse function s^{-1} does not leak anything about s, and $c(x)$ does not leak anything about x. Those intuitions for unforgeability and blindness were more formally captured in follow-up work.

Pointcheval and Stern [40,41] defined unforgeability using a so-called one more forgery. Instead of defining it as the inability to extract the secret key s from the public key s^{-1}, they introduced a security experiment where the adversary is given oracle access to the signer. The winning condition is to output $k+1$ message-signature pairs while only making k queries to the oracle. Juels, Luby, and Ostrovsky [33] introduced a formal experiment defining the notion of blindness, where the adversary must guess the order in which it issues two messages $(\mathsf{m}_0, \mathsf{m}_1)$ of its choosing. This definition considers the signing keys to be *honestly* generated by the experiment and given to the adversary. Contrary to that, the adversary outputs the public key in the *maliciously* generated key model [18]. The adversary is not required to execute the key generation algorithm or even to know the corresponding secret key. At the end of the blindness experiment, the adversary also receives the corresponding signatures (σ_0, σ_1). If one of the signatures was incorrect or the user algorithm aborted, the adversary gets $(\perp, \perp)$ instead information about which interaction failed. Camenisch, Neven, and Shelat introduced blindness under selective failure [13], which considers this additional information. Fischlin and Schröder later showed [19] how to turn every blind signature scheme into a selective-failure blind one.

Chaum's definition considers what we call today a two-move blind signature scheme with one message from the user and one from the signer. While follow-up work considers the interaction between the user and signer as an interactive protocol with multiple rounds, two-move blind signatures are considered round-optimal and provide concurrent security. Many round-optimal schemes were proposed [18,22,34], including the practical blind BLS scheme [10] with

concise signature size. Interestingly, in his work Pass [38], calls two-move blind signatures non-interactive. Notably, interaction is inherent since the user must keep an internal state to de-blind the signer's response. Three-move blind signatures were also explicitly defined [29] since they can be generically constructed from linear identification schemes in the random oracle model. Constructing such schemes from standard assumptions without the random oracle or common reference string model seems hard [20]. We can build blind signatures from various assumptions, including post-quantum secure ones [3,30]. Recently, Chairattana-Apirom et al. [15] showed how to build concurrent secure blind signature schemes from discrete logarithm and RSA-type assumptions.

Paraphrasing the research question stated above. Can we use the observation that in many applications, the user can randomly choose the message to design something new that was not considered in prior work and opens a new chapter in the blind signature literature?

Our Contribution

In the case of two-move blind signatures, the user/recipient sends the requested message in the blinded form to the signer and later uses the interactions state to unblind the response. Our main idea is that since the recipient does not require any specific distribution or structure of the message, the message can be an output of the unblinding step. In other words, *there is no need for interaction.*

We capture this by defining a new cryptographic primitive called non-interactive blind signatures for random messages or NIBS for short. As it turned out, defining meaningful notation for such signatures is not simple, and the following strawman approach *does not work*. Since no interaction is required, the signer can create a presignature psig on some random message/nonce and share it. The recipient can then finalize the presignature to a signature on a random message. As already mentioned, such a notion does not work and cannot provide meaningful security properties. The problem with defining non-interactive blind signatures in that way is that the recipient can repeat the process and get a new message-signature pair. The returned pair must be a signature on a new message. Otherwise, the signer could link the issuing process to the final signature. Fortunately, our notion of NIBS does not have the same problems as the strawman approach.

What is needed is some secret input from the recipient. The natural idea is to use the recipient's public key. The signer must include the recipient's public key as input to the signing process. The returned presignature can be finalized using the recipient's secret key and the signer's nonce. Later, we will show a scheme where we can use, for example, the recipient's PKI public key or ephemeral keys from TLS connections. Using the preexisting public key of the recipient, we can use NIBS in various applications where we cannot use standard blind signatures.

The main contribution of this paper is the formal definition of NIBS and an appropriate security model. The intuition behind NIBS can be easily explained using an analogous notion to the one used by Chaum [16]. The signer computes a presignature $s(\mathsf{nonce}, c)$ using issuing function s, a nonce of its choosing,

and the recipient's function c. The recipient can finalize the presignature using $c^{-1}(\mathsf{nonce}, s(\mathsf{nonce}, c)) = (\mathsf{m}, s(\mathsf{m}))$, i.e., use the corresponding secret key c^{-1} to its public key c. We require that m and $s(\mathsf{m})$ not leak any information about c and the used nonce. We capture those properties using two definitions called *recipient blindness* and *nonce blindness*. The former captures the property that m and $s(\mathsf{m})$ do not leak information about c while the latter property that they do not leak any information about nonce. We formally define recipient blindness via an experiment where the adversary is explicitly given two honestly generated public keys of recipients and outputs two presignatures finalized by the experiment to $(\mathsf{m}_0, \mathsf{sig}_0)$ and $(\mathsf{m}_1, \mathsf{sig}_1)$. Finally, the adversary is given $(\mathsf{m}_0, \mathsf{sig}_0)$ and $(\mathsf{m}_1, \mathsf{sig}_1)$ in random order. For nonce blindness, the adversary is given just one recipient public key. In both cases, we do not consider aborts, i.e., if one of the signatures cannot be finalized, we give $(\perp, \perp)$ to the adversary. Our blindness definitions are defined in the malicious key model, where the adversary generates the signing key. We also define an honest key model notion. For unforgeability, we consider the standard one-more definition. The adversary gets access to an oracle returning presignatures for the adversary-specified nonce and public key. In the end, the adversary must return more valid signatures than queries, similar to one-more unforgeability of standard blind signatures.

NIBS are distinct from two-move blind signatures [38] and standard blind signatures in general. The former supports recipient-specified messages, while in NIBS, the message is an output of the unblinding process and is unpredictable for the recipient and the signer. NIBS can be issued without interaction, given the recipient's public key. Standard blind signatures are inherently interactive and require the recipient to keep a state to unblind the signer's response successfully. There are two ways of using NIBS. The first way is to use an existing PKI for recipients' public keys. A signer can issue presignatures to a set of users without interacting with them and publish the corresponding presignatures. Recipient blindness ensures that a given final message-signature pair cannot be linked to any particular recipient in the set. Nonce blindness allows the signer to repeat the process and issue more than one presignature per user. Alternatively, NIBS can be part of a two-move protocol. The first message is a freshly generated recipient public key, and the signer's response is the presignature. The result is not a standard two-move blind signature since the recipient will receive a signature under a random message. However, as we already discussed, this is acceptable in many applications. The main advantage compared to standard blind signature is that we can reuse the first message of the two-move protocol with NIBS in consecutive runs. The same is not possible for standard blind signatures. Nonce blindness ensures that reusing the recipient's public key does not allow an adversary to link final message-signature pairs to a recipient.

The main disadvantage of non-interactive blind signatures is that there is no simple way of including information about the freshness of the signature. Consider the following scenario. The user must provide a fresh signature that she adheres to the service's policies to access it, which might include a per-day limit. The signature is implemented via a blind signature scheme to protect the

user's privacy. Of course, the user can hoard signatures and reuse them later. So the service requires that a challenge be signed instead of a random message, and NIBS cannot be used here. However, note that we already know how to date blind signatures [2] using partial blindness. We can formalize a similar definition in the non-interactive setting. Our next contribution is the definition of tagged non-interactive blind signatures TNIBS.

Definitions themself are not attractive if one cannot instantiate them with existing cryptography. Therefore we show how to build NIBS and TNIBS efficiently in the random oracle model. The central primitive we use are signatures on equivalence classes (SPS-EQ) [24,27] and their tagged version (TBEQ) [28]. Interestingly, both schemes support recipient public keys in the form of standard discrete logarithm public keys $\mathsf{pk} = g^{\mathsf{sk}}$. Thus, our construction supports keys from various schemes like El-Gamal encryption, Diffie-Hellman key exchange, Schnorr, and DSA/ECDSA signatures. One caveat is that the underlying group (generated by g) must be a source group for which an admissible bilinear pairing function exists. Fortunately, we know how to construct such pairing-friendly groups [9] and use them to support the above algorithms. As our last contribution, we propose a generic construction of NIBS and TNIBS, which we can construct by generalizing ideas from the equivalence class construction. Both generic constructions are setup-free but rely on the random oracle model. We targetted a setup-free setting since it, by definition, allows to reuse recipients keys from different schemes, i.e., the recipients are not required to use a common reference string (CRS) to generate their keys. However, one can easily translate both schemes to work with a CRS instead of the random oracle model. For completeness, we will now summarize our contribution. In this paper, we improved the state-of-the-art of blind signatures as follows.

1. We introduce the notion and security model for (tagged) non-interactive blind signatures.
2. We show a very efficient construction in the random oracle model from signatures on equivalence classes that works well with preexisting public keys.
3. We provide a generic construction of NIBS and TNIBS in the random oracle model using verifiable random functions, digital signatures, and non-interactive proofs. Depending on the requirements, one can easily replace the random oracle model with the common reference string model.

Our Techniques

One of the main contributions of our paper is the efficient instantiation of non-interactive blind signatures in the random oracle model. Our construction is based on signature on equivalence classes that were already used as a building block to construct round-optimal blind signatures [22,23]. A SPS-EQ signature on (g, g^x) can be transformed to a signature under (g^r, g^{rx}) without the secret signing key. Moreover, the transformed signature on (g^r, g^{rx}) is indistinguishable from a fresh signature on this message. This property is called the perfect adaptation of signatures. Together with the fact that messages (g, g^x) and (g^r, g^{rx})

are indistinguishable, under the decisional Diffie-Hellman assumption, form the main privacy property used in the design of [22,23].

The idea of our construction is as follows. The signer uses the SPS-EQ scheme to sign $(g^{\mathsf{sk}_R}, \mathsf{H}(\mathsf{nonce}))$, where g^{sk_R} is the recipients public key. It is worth noting that this key can be a preexisting public key of the recipient, as already mentioned in the introduction. The equivalence class signature is the NIBS presignature the user receives from the signer. The user can easily check the presignature using the SPS-EQ verification function. The interesting part follows. The actual NIBS signature is a SPS-EQ signature on the message $(g, \mathsf{H}(\mathsf{nonce})^{\mathsf{sk}_R^{-1}})$, also called the canonical representative of the equivalence class [5]. Note that we fix the first component of the message vector to g for the user to be able to compute exactly one blind signature from a presignature.

The unforgeability of the scheme directly follows from the unforgeability of the SPS-EQ scheme and the canonical representative notion. A "fresh" blind signature implies a signature on a class that was not signed already, which constitutes a valid forgery for the SPS-EQ scheme. On the other hand, blindness follows from the perfect adaptation of the SPS-EQ scheme and the inverse and strong decisional Diffe-Hellman assumptions. The idea is that anyone but the recipient cannot distinguish $\mathsf{H}(\mathsf{nonce})$ from $\mathsf{H}(\mathsf{nonce})^{\mathsf{sk}_R^{-1}}$. Thanks to careful random oracle programming, we show that the blindness of our construction relies on those assumptions.

We can easily modify this construction to support tags. In other words, with small changes, we can construct a TNIBS scheme using the same ideas. Hanzlik and Slamanig [28] introduced the notion of tag-based equivalence class signatures TBEQ. Contrary to SPS-EQ, they allow the signer to specify a tag τ that remains unchanged even after the user transforms the signature to a different representative of the same class. The idea of our TNIBS scheme is simple. We replace the SPS-EQ scheme in the NIBS construction above with a TBEQ scheme. The resulting construction is a tag-based, non-interactive blind signature scheme. Blindness follows from the same assumptions. The main difference is the unforgeability, which is now based on the unforgeability of the TBEQ scheme.

Surprisingly this construction follows a blueprint that we describe in the form of our generic construction. Firstly, we notice that the message in the construction is $\mathsf{H}(\mathsf{nonce})^{\mathsf{sk}_R^{-1}}$. The key property we use is that this value is unpredictable for the signer. Otherwise, the blindness property cannot hold. Looking at this value more closely, we notice that it can be interpreted as an evaluation of a pseudo-random function (PRF) on input nonce with key sk_R. In other words, the presignature defines the input to the PRF and its key. The blind signature is a signature on the evaluation of the PRF. Note that rather than a PRF, we have to consider a verifiable random function (VRF) since it provides a verification key that can be used as a public key. Finally, we observe that the SPS-EQ signature ensures that the recipient can only receive a valid blind signature if she correctly evaluates the VRF. Thus, the signature on equivalence classes actually acts as

a proof system that binds the recipient and ensures correct evaluation of the random function.

Equipped with all those observations, we define our generic construction as follows. The signer creates the presignature by signing the recipient's VRF verification key pk_R and a nonce nonce. Since the presignature is a standard digital signature, it can be easily verified. The random blind signature message m is the evaluation of the VRF on nonce. The signature is a non-interactive proof that the recipient knows a signature on pk_R and nonce and that m is a proper evaluation of a VRF with key pk_R on input nonce. We can also use this blueprint to construct TNIBS generically. The only difference is that the presignature is additionally a signature under the tag τ. The statement of the proof system also changes a bit and now must include τ as part of the statement.

Applications

Privacy Pass. Privacy Pass [17] is a system designed to make life easier for anonymous network users who frequently solve CAPTCHAs. The idea is to let users first get a single-use token from an issuer via a non-anonymous network connection and let them redeem those tokens instead of solving CAPTCHAs. This provides a more user-friendly experience when using an anonymous network like TOR or a VPN connection. A Privacy-pass token is composed of the input and output of an oblivious pseudo-random function, where the issuer holds the function's secret key. During the issuing process, the user's platform (e.g., browser extension) requests an evaluation of the PRF on a chosen random input, similar to the e-cash scenario. It is worth noting that an oblivious PRF can be seen as a designated-verifier blind signature.

Recently, a rate-limiting version of Privacy Pass [32] called Private Access Token (PAT) was introduced. In this new variant, the number of tokens a user can get depend on a policy enforced by a trusted mediator. Moreover, the RSA blind signature scheme is used instead of the oblivious PRF, and the signed message is chosen as part of the service's challenge. This new version was recently introduced into iOS 16 and is supported by Apple[1]. In this setting, iCloud plays the role of the mediator and enforces the service's access policy. If the policy applies to the user, the issuer finalizes an RSA blind signature query made by the user. For a formal analysis of the RSA blind signature scheme used in PAT, see [35]. In both versions, the user and issuer must repeat the protocol several times to create multiple tokens at once, i.e., batch issuing. Although both parties can execute the protocol concurrently, the user is always required to participate. The same problem arises in the case of issuing more tokens after some time. We can improve this using (tagged) non-interactive blind signatures (see Table 1).

As we mentioned multiple times in the e-cash scenario, we are not always interested in the structure of the signed message and only care about the freshness of the token. This is also the case for the standard version of Privacy Pass, where the user chooses the input of the oblivious PRF. Replacing the oblivious

[1] https://developer.apple.com/videos/play/wwdc2022/10077/.

PRF with NIBS would improve the communication complexity in the case of multi-token issuance. In such a case, the user sends her public key to the issuer and reives n presignatures, which can be finalized into n unique tokens. The communication complexity from the user to the issuer is independent of n. The issuer can also afterward decide on the number of issued tokens. The exciting part is that to create more tokens, the issuer does not need to interact with the user and can make fresh presignatures using the user's public key. This design allows the issuer to issue new tokens periodically without interaction. Users can then later download them at their convenience. It is worth noting that this is impossible with an oblivious PRF or standard blind signatures, which inherently require interaction between both parties.

Unfortunately, standard NIBS cannot replace the RSA blind signature scheme in Private Access Token since the service and not the user chooses the signed message. PAT was introduced to enforce an access policy and only issue tokens for users adhering to the policy. Because the service chooses the blindly signed message, it knows that the user must conform to the latest policy, and the proof is fresh. Otherwise, a malicious user could hoard and use tokens during a given period breaking any per-day (or other) time policies. It is worth noting that the service gains no additional properties by picking a non-random message since it is hidden from both the mediator and the issuer. To get around this, instead of using NIBS, we can use the tagged version TNIBS. The tag remains unchanged after the user transforms the presignature into the final signature. This way, the service can date signatures. Using TNIBS instead of the RSA blind signature would have the same benefits as using NIBS in the case of the standard Privacy Pass. Moreover, our TNIBS solution is DLP based, which would be an alternative to the required RSA assumption.

Whistleblowing System. Ring signatures [42] were introduced as a way for whistleblowers to leak trusted intel without revealing their identity. According to a recent EU directive [1], big companies must implement a whistleblowing system for their employees to leak information anonymously about any misconduct of the employer. Ring signatures would be an ideal candidate to support such a system. The whistleblower combines the public keys of all employees and creates the ring signature. This solution does not work in case no PKI is implemented at the company. An alternative approach would be to build a system supporting Privacy Pass. A company or third-party supported service would issue single-use tokens to verified employees, who can later redeem the token with the intel. Unfortunately, this solution is inherently not private. Employees must first request a token, making them a target, i.e., whistleblowers hide inside the anonymity set of token owners and not in the set of all employees.

Implementing a whistleblowing system using NIBS would mitigate some of those problems. Assuming the recipient's public key is the ephemeral Diffie-Hellman key used for establishing a TLS connection, the system could look as follows. Every time an employee connects to some internal system of the company, she gets a token for the whistleblowing system. To redeem the token, the employee must install a plugin that retains the ephemeral TLS credential and

later uses them to finalize the NIBS. It is worth noting that in this design, the company is oblivious to who installed the plugin, and potentially all employees could be the owner of a token showing up in the whistleblowing system. This application shows the power of our non-interactive blind signatures. We can use NIBS in systems where all recipients are potential users and can either use the presignature or just ignore it without the signer knowing about their choice.

Airdropping e-coins. Airdropping is a mechanism that allows sending cryptocurrency to users. This technique is frequently used to bootstrap interest in a currency by gifting cryptocurrency to users. An ideal scheme preserves the privacy of the recipient [43] once she redeems her coins. An airdropping system must also provide means for public accountability so that users can check that the airdropping mechanism will only produce a limited number of unique tokens.

Tublebit [31] is a protocol for anonymous cryptocurrency payments. At its core, the protocol implements the e-cash scenario. A designated party called the tumbler issues blind signatures for cryptocurrency payments. Blind signatures can later be redeemed to finalize the payment. Non-interactive blind signatures can add the airdropping functionality to the tumbler, introducing potential ways to attract new users. The key property of NIBS that allows this is non-interactives. The tumbler can look for publicly available public keys/addresses (e.g., on the blockchain or Github) and blindly drop NIBS to their owners.

Lottery System. NIBS can also be used to implement a fair lottery system. Users can register their public key for a given round by paying the lottery fee. What each user receives from the service is a NIBS presignature. The lottery winner is the user with a valid signature under the smallest/biggest message. This approach requires the service to replace the signing key with each lottery round. However, if we use the tagged version, the service can easily tag each signature with the round for which it was created. The lottery is fair, and the service cannot predict the outcome of the lottery because of the blindness property. On the other hand, because of one-more unforgeability, only users that pay can receive the prize.

Open Problems and Relation to Impossibility Results

The main open problem is to design an efficient NIBS scheme without pairings. Although one can instantiate our generic construction without using them, it will probably not be efficient due to the general-purpose use of proof systems. Efficient instantiations of NIBS from post-quantum assumptions are also desired. Another interesting problem is instantiating NIBS from standard assumptions without the CRS or random oracle model. Fischlin and Schröder [20] showed that constructing a statistical blind three-move blind signature from standard falsifiable assumptions without relying on the random oracle model or the common reference string model is impossible. The results carry over to two-move schemes and computational blind schemes with certain additional constraints.

Fortunately, there exist ways to circumvent those impossibility results, e.g., using complexity leveraging [25].

As already mentioned, one way of using NIBS is to run a two-move protocol. One would think this means that impossibility results also apply to NIBS. However, this is unclear and requires further investigation. Recall that a two-move protocol from NIBS is not a standard blind signature. In the latter, the recipient can arbitrarily choose the message, whereas, in the former protocol, the message depends on the nonce selected by the signer and the recipient's secret key. In other words, despite the NIBS two-move protocol being useful in similar applications as standard blind signatures, the notion is different.

We leave two open questions here. The first would be to verify if one can extend the impossibility results to blind signature schemes, where the message is not chosen by the recipient but is an output of the signing protocol. Note that the two-move protocol based on NIBS is an instantiation of such blind signatures. A positive answer to this question would mean that NIBS cannot be instantiated from standard falsifiable assumptions without ROM or a trusted setup phase. Alternatively, one could try to construct such a NIBS scheme, implying that the impossibility results from [20] do not hold if the message is not chosen by the recipient but as part of the protocol.

2 Preliminaries

2.1 Notation, Bilinear Groups and Assumptions

We denote by $y \leftarrow \mathcal{A}(x)$ the execution of algorithm $\mathcal{A}$ on input x and with output y. By $r \leftarrow_\$ S$ we mean that r is chosen uniformly at random over the set S. We will use $1_\mathbb{G}$ to denote the identity element in group $\mathbb{G}$ and $[n]$ to denote the set $\{1, \ldots, n\}$. Throughout the paper we will use the multiplicative notation and by $\mathcal{A}^{\mathcal{O}}$ we denote an algorithm $\mathcal{A}$ that has access to oracle $\mathcal{O}$.

Definition 1 (Bilinear Groups). *Let us consider cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ of prime order p. Let g_1, g_2 be generators of respectively $\mathbb{G}_1$ and $\mathbb{G}_2$. We call $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ a* bilinear map *(pairing) if it is efficiently computable and the following holds: 1) Bilinearity: $\forall (S, T) \in \mathbb{G}_1 \times \mathbb{G}_2$, $\forall a, b \in \mathbb{Z}_p$, we have $e(S^a, T^b) = e(S, T)^{a \cdot b}$, 2) Non-degeneracy: $e(g_1, g_2) \neq 1$ is a generator of group $\mathbb{G}_T$. We will consider Type-3 pairings, i.e., there is no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.*

Definition 2 (Inverse Decisional Diffie-Hellman Assumption in $\mathbb{G}_1$ [7]). *For all PPT adversaries $\mathcal{A}$ given elements $(g_1^\alpha, g_1^\beta) \in \mathbb{G}_1^2$ it is hard to decide whether $\beta = \alpha^{-1} \mod p$ or $\beta \leftarrow_\$ \mathbb{Z}_p^*$. We will use $\mathbf{Adv}_{\text{invDDH}}(\mathcal{A})$ to denote the advantage of the adversary $\mathcal{A}$ in solving this problem.*

Definition 3 (Strong Decisional Diffie-Hellman Assumption in $\mathbb{G}_1$ [39]). *For all PPT adversaries $\mathcal{A}$ given elements $(g_1^\alpha, g_1^\beta, g_1^{\beta^{-1}}, g_1^\gamma) \in \mathbb{G}_1^4$ it is hard to decide whether $\gamma = \alpha \cdot \beta \mod p$ or $\gamma \leftarrow_\$ \mathbb{Z}_p^*$. We will use $\mathbf{Adv}_{\text{sDDH}}(\mathcal{A})$ to denote the advantage of the adversary $\mathcal{A}$ in solving this problem.*

2.2 Signature Schemes

Definition 4. *A signature scheme* SIG *consists of three PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *with the following syntax.*

$\mathsf{KeyGen}(\lambda)$**:** *On input a security parameter* λ*, it outputs a public and secret signing key* $(\mathsf{pk}, \mathsf{sk})$*.*

$\mathsf{Sign}(\mathsf{sk}, \mathsf{m})$**:** *On input a key* sk *and a message* m*, it outputs a signature* σ*.*

$\mathsf{Verify}(\mathsf{pk}, \mathsf{m}, \sigma)$**:** *On input a public key* pk*, a message* m *and a signature* σ*, it outputs either* 0 *or* 1*.*

We require the following properties of a signature scheme.

Correctness: *For every security parameter* $\lambda \in \mathbb{N}$ *and every message* m *given that* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{SIG.KeyGen}(\lambda)$*,* $\mathsf{sig} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, \mathsf{m})$ *it holds that*

$$\mathsf{SIG.Verify}(\mathsf{pk}, \mathsf{m}, \mathsf{sig}) = 1.$$

Existential Unforgeability under Chosen Message Attacks: *Every PPT adversary* $\mathcal{A}$ *has at most negligible advantage in the following experiment.*

$\underline{\mathsf{EUF\text{-}CMA}_{\mathcal{A},\mathsf{SIG}}(\lambda)}$

$Q := \emptyset$

$(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{SIG.KeyGen}(\lambda)$

$(\mathsf{m}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{O}_1(\mathsf{sk}, \cdot)}(\mathsf{pk})$

return $\mathsf{m}^* \neq \mathsf{m} \quad \forall \mathsf{m} \in Q \wedge$
$\mathsf{SIG.Verify}(\mathsf{pk}, \mathsf{m}^*, \sigma^*) = 1$

$\underline{\mathcal{O}_1(\mathsf{sk}, \mathsf{m})}$

$\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, \mathsf{m})$

$Q := Q \cup \{\mathsf{m}\}$

return σ

The advantage of $\mathcal{A}$ *is defined by* $\mathbf{Adv}_{\mathsf{SIG}}(\mathcal{A}) = \Pr[\mathsf{EUF\text{-}CMA}_{\mathcal{A},\mathsf{SIG}}(\lambda) = 1]$.

2.3 Dual-Mode Witness Indistinguishable Proofs

In our generic construction, we will use non-interactive proofs. To this end, we will use the dual-mode non-interactive witness indistinguishable proof system proposed by Groth-Sahai (GS) [26]. The main property of this system is that there exists a common reference string (crs) that can be either in the "binding" or "hiding" modes. Depending on the type, the system satisfies perfect soundness and extractability or perfect witness indistinguishability.

An interesting property of GS proofs [26] is that crs is composed of group elements that depending on the mode, fulfill a specific relation, e.g., a DDH tuple can be used as in the binding mode, and a non-DDH tuple in the hiding mode. Thus, instead of generating the common string by a trusted party, we can use the random oracle to output it. The idea is that with high probability, we will end up with a string in the hiding mode by querying the random oracle $\mathsf{H}_{\mathsf{crs}}(1)$, which outputs values of the form of reference strings. On the other hand, the reduction in the proof can program oracle $\mathsf{H}_{\mathsf{crs}}$ to output a string in binding mode.

Definition 5 (Dual-Mode Witness Indistinguishable Proofs). *A dual-mode witness indistinguishable proof system for language $\mathcal{L}_{\mathcal{R}}$ consists of algorithms* $\mathsf{DMWI} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Extract})$ *with the following syntax.*

$\mathsf{Setup}(\lambda, \mathsf{binding})$**:** *On input of security parameter, it outputs a common reference string* crs *which we call binding. It additionally outputs a trapdoor* $\mathsf{td}_{\mathsf{Ext}}$.

$\mathsf{Setup}(\lambda, \mathsf{hiding})$**:** *On input of security parameter, it outputs a common reference string* crs*, which we call a hiding reference string.*

$\mathsf{Prove}(\mathsf{crs}, x, w)$**:** *On input a common reference string* crs*, a statement x and a witness w, it outputs a proof π.*

$\mathsf{Verify}(\mathsf{crs}, x, \pi)$**:** *On input the common reference string* crs*, a statement x, a proof π, it outputs either 0 or 1.*

$\mathsf{Extract}(\mathsf{td}_{\mathsf{Ext}}, x, \pi)$**:** *On input the extraction trapdoor* $\mathsf{td}_{\mathsf{Ext}}$*, a statement x and a proof π, it outputs a witness w.*

We require that DMWI *meets the following properties.*

Mode Indistinguishability: *For all λ we define the advantage of $\mathcal{A}$ against mode indistinguishability as follows:* $\mathbf{Adv}_{\mathsf{modeIND},\mathcal{A}}(\lambda) =$

$$\left| \Pr\left[\mathsf{mode} = \mathsf{mode}^* : \begin{array}{c} \mathsf{mode} \leftarrow_{\$} \{\mathsf{binding}, \mathsf{hiding}\}; \\ (\mathsf{crs}) \leftarrow \mathsf{Setup}(\lambda, \mathsf{mode}); \\ \mathsf{mode}^* \leftarrow \mathcal{A}(\lambda, \mathsf{crs}) \end{array} \right] - \frac{1}{2} \right|,$$

where the probability is taken over the random choice of mode *and the random coins of* Setup*. We say that the proof system is mode indistinguishable if for all PPT adversaries, $\mathcal{A}$ the advantage is negligible.*

Perfect Completeness in both Modes: *For all security parameters $\lambda \in \mathbb{N}$, all statements $x \in \mathcal{L}_{\mathcal{R}}$ and all witnesses w for which $\mathcal{R}(x, w) = 1$,* $\mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, \mathsf{binding})$*, and* $\pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)$ *it holds that* $\mathsf{Verify}(\mathsf{crs}, x, \pi) = 1$*. The same holds for* $\mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, \mathsf{hiding})$.

Perfect Soundness in Binding Mode: *For all adversaries $\mathcal{A}$ we have*

$$\Pr\left[\begin{array}{c} (\mathsf{crs}, \mathsf{td}_{\mathsf{Ext}}) \leftarrow \mathsf{Setup}(\lambda, \mathsf{binding}) \\ (x, \pi) \leftarrow \mathcal{A}(\mathsf{crs}) \end{array} : \begin{array}{c} \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \\ \wedge\ x \notin \mathcal{L}_{\mathcal{R}} \end{array} \right] = 0$$

Extractability in Binding Mode: *For any (x, π), it holds:*

$$\Pr\left[\begin{array}{c} (\mathsf{crs}, \mathsf{td}_{\mathsf{Ext}}) \leftarrow \mathsf{Setup}(\lambda, \mathsf{binding}) \\ w \leftarrow \mathsf{Extract}(\mathsf{td}_{\mathsf{Ext}}, x, \pi) \end{array} : \begin{array}{c} \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \\ \implies\ \mathcal{R}(x, w) = 1 \end{array} \right] = 1$$

Perfect Witness-Indistinguishability in Hiding Mode: *We say that proof system for language $\mathcal{L}_{\mathcal{R}}$ is perfectly witness indistinguishable if all adversaries $\mathcal{A}$ the following is 0:*

$$\left| \Pr\left[\begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(\lambda, \mathsf{hiding}) \\ (x, w_0, w_1) \leftarrow \mathcal{A}(\mathsf{crs}) \\ b \leftarrow_{\$} \{0, 1\} \\ \pi^* \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w_b) \end{array} : \hat{b} \leftarrow \mathcal{A}(\mathsf{crs}, \pi^*) \right] - \frac{1}{2} \right|,$$

where $\mathcal{A}$ is restricted to outputs such that $\mathcal{R}(x, w_0) = \mathcal{R}(x, w_1) = 1$.

2.4 Verifiable Random Function [36]

Definition 6 (Verifiable Random Function VRF). *A verifiable random function* $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{P}, \mathsf{V})$ *with input length* $n(\lambda)$ *and output length* $m(\lambda)$ *consists of the following PPT algorithms:*

$\mathsf{Gen}(\lambda)$**:** *On input of security parameter, outputs secret key* $\mathsf{sk}_{\mathsf{VRF}}$ *and public verification key* $\mathsf{pk}_{\mathsf{VRF}}$.

$\mathsf{Eval}(\mathsf{sk}_{\mathsf{VRF}}, x)$**:** *On input a secret key* $\mathsf{sk}_{\mathsf{VRF}}$ *and input value* $x \in \{0,1\}^{n(\lambda)}$ *it returns the output value* $y \in \{0,1\}^{m(\lambda)}$

$\mathsf{P}(\mathsf{sk}_{\mathsf{VRF}}, x)$**:** *On input a secret key* $\mathsf{sk}_{\mathsf{VRF}}$ *and* x *this prover algorithm outputs a proof* π_{VRF} *that* y *is consistent with the verification key* $\mathsf{pk}_{\mathsf{VRF}}$.

$\mathsf{V}(\mathsf{pk}_{\mathsf{VRF}}, \pi_{\mathsf{VRF}}, x, y)$**:** *On input a verification key* $\mathsf{pk}_{\mathsf{VRF}}$*, proof* π_{VRF}*,* x*,* y *this algorithm outputs* 1 *or* 0.

We require that VRF *meets the following properties.*

Completness: *For every security parameter* λ *and input* $x \in \{0,1\}^{n(\lambda)}$

$$\Pr\left[\begin{array}{l}(\mathsf{sk}_{\mathsf{VRF}}, \mathsf{pk}_{\mathsf{VRF}}) \leftarrow \mathsf{Gen}(\lambda) \\ \quad y \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{VRF}}, x) \quad : \mathsf{V}(\mathsf{pk}_{\mathsf{VRF}}, \pi_{\mathsf{VRF}}, x, y) = 1 \\ \quad \pi_{\mathsf{VRF}} \leftarrow \mathsf{P}(\mathsf{sk}_{\mathsf{VRF}}, x)\end{array}\right] = 1.$$

Uniqueness: *For every security parameter* λ *and input* $x \in \{0,1\}^{n(\lambda)}$*, arbitrary verification key* $\mathsf{pk}_{\mathsf{VRF}}$*, there exists at most a single* $y \in \{0,1\}^{m(\lambda)}$ *for which there exists an accepting proof* π_{VRF}*. That is, if*

$$\mathsf{V}(\mathsf{pk}_{\mathsf{VRF}}, \pi_{\mathsf{VRF}}, x, y) = \mathsf{V}(\mathsf{pk}_{\mathsf{VRF}}, \pi'_{\mathsf{VRF}}, x, y') = 1$$

then $y = y'$.

Adaptive Indistinguishability: *Every PPT adversary* $\mathcal{A}$ *has at most negligible advantage in the following experiment.*

$\underline{\mathsf{Exp}_{\mathcal{A},\mathsf{VRF}}(\lambda)}$

$Q := \emptyset$
$b \leftarrow_{\$} \{0,1\}$
$(\mathsf{sk}_{\mathsf{VRF}}, \mathsf{pk}_{\mathsf{VRF}}) \leftarrow \mathsf{Gen}(\lambda)$
$(\mathsf{st}, x^*) \leftarrow \mathcal{A}^{\mathcal{O}_1(\mathsf{sk}_{\mathsf{VRF}}, \cdot)}(\mathsf{pk}_{\mathsf{VRF}})$
$y_0 \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{VRF}}, x^*)$
$y_1 \leftarrow_{\$} \{0,1\}^{m(\lambda)}$
$\bar{b} \leftarrow \mathcal{A}^{\mathcal{O}_1(\mathsf{sk}_{\mathsf{VRF}}, \cdot)}(\mathsf{st}, y_b)$
return $b = \bar{b} \ \wedge \ x^* \notin Q$

$\underline{\mathcal{O}_1(\mathsf{sk}_{\mathsf{VRF}}, x)}$

$y \leftarrow \mathsf{Eval}(\mathsf{sk}_{\mathsf{VRF}}, x)$
$\pi_{\mathsf{VRF}} \leftarrow \mathsf{P}(\mathsf{sk}_{\mathsf{VRF}}, x)$
$Q := Q \cup \{x\}$
return (y, π_{VRF})

The advantage of $\mathcal{A}$ *is defined by* $\mathbf{Adv}_{\mathsf{VRF}}(\mathcal{A}) = \Pr[\mathsf{Exp}_{\mathcal{A},\mathsf{VRF}}(\lambda) = 1]$.

2.5 Structure-Preserving Signatures of Equivalence Classes

Structure-preserving signatures on equivalence classes (SPS-EQ) [24,27] can be used to sign equivalence classes $[M]$ of vectors $M \in (\mathbb{G}_i^*)^\ell$ for $\ell > 1$ and with equivalence relation: $M, N \in \mathbb{G}_i^\ell : M \sim_{\mathcal{R}} N \Leftrightarrow \exists\, s \in \mathbb{Z}_p^* : M = N^s$.

Definition 7 (SPS-EQ). *An SPS-EQ scheme* $\mathsf{SPS\text{-}EQ}$ *on message space* $(\mathbb{G}_i^*)$ *for* $i \in \{1,2\}$ *consists of the following PPT algorithms.*

$\mathsf{KeyGen}_{\mathsf{EQ}}(\lambda, \ell)$**:** *On input of security parameter* λ *and input message vector length* $\ell > 1$*, it outputs a key pair* $(\mathsf{sk}_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}})$.

$\mathsf{Sign}_{\mathsf{EQ}}(\mathsf{sk}_{\mathsf{EQ}}, M)$**:** *On input of input a secret key* $\mathsf{sk}_{\mathsf{EQ}}$ *and representative* $M \in (\mathbb{G}_i^*)^\ell$*, outputs a signature* σ_{EQ} *for equivalence class* $[M]$.

$\mathsf{ChgRep}_{\mathsf{EQ}}(M, \sigma_{\mathsf{EQ}}, \mu, \mathsf{pk})$**:** *On input of input representative* $M \in (\mathbb{G}_i^*)^\ell$ *of equivalence class* $[M]$*, a signature* σ_{EQ} *on* M*, a value* μ *and a public key* $\mathsf{pk}_{\mathsf{EQ}}$*, returns an updated message-signature pair* (M', σ')*, where the new representative is* $M' = M^\mu$ *and* σ'_{EQ} *its corresponding (or, updated) signature.*

$\mathsf{Verify}_{\mathsf{EQ}}(\mathsf{pk}_{\mathsf{EQ}}, M, \sigma_{\mathsf{EQ}})$**:** *On input of a public key* $\mathsf{pk}_{\mathsf{EQ}}$*, a representative* $M \in (\mathbb{G}_i^*)^\ell$*, and a signature* σ_{EQ} *it deterministically outputs a bit* $b \in \{0,1\}$.

$\mathsf{VKey}_{\mathsf{EQ}}(\mathsf{sk}_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}})$**:** *On input of secret key* $\mathsf{sk}_{\mathsf{EQ}}$ *and a public key* $\mathsf{pk}_{\mathsf{EQ}}$*, it deterministically checks if it represents a valid key pair and outputs a bit* b.

Definition 8 (Correctness). *An* $\mathsf{SPS\text{-}EQ}$ *scheme on* $(\mathbb{G}_i^*)^\ell$ *is called correct if for all security parameters* $\lambda \in \mathbb{N}$*,* $\ell > 1$*,* $(\mathsf{sk}_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}}) \leftarrow \mathsf{KeyGen}_{\mathsf{EQ}}(\lambda, \ell)$*,* $M \in (\mathbb{G}_i^*)^\ell$ *and* $\mu \in \mathbb{Z}_p^*$*:*

$$\mathsf{VKey}_{\mathsf{EQ}}(\mathsf{sk}_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}}) = 1 \;\wedge\; \Pr\left[\mathsf{Verify}_{\mathsf{EQ}}(M, \mathsf{Sign}_{\mathsf{EQ}}(M, \mathsf{sk}_{\mathsf{EQ}}), \mathsf{pk}_{\mathsf{EQ}}) = 1\right] = 1$$
$$\wedge \Pr\left[\mathsf{Verify}_{\mathsf{EQ}}(\mathsf{ChgRep}_{\mathsf{EQ}}(M, \mathsf{Sign}_{\mathsf{EQ}}(M, \mathsf{sk}_{\mathsf{EQ}}), \mu, \mathsf{pk}_{\mathsf{EQ}}), \mathsf{pk}_{\mathsf{EQ}}) = 1\right] = 1.$$

Definition 9 (EUF-CMA). *For scheme* $\mathsf{SPS\text{-}EQ}$ *and adversary* $\mathcal{A}$ *we define the following experiment:*

$\mathsf{EUF\text{-}CMA}_{\mathcal{A},\mathsf{SPS\text{-}EQ}}(\lambda, \ell)$

$Q := \emptyset$
$(\mathsf{sk}_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}}) \leftarrow \mathsf{KeyGen}_{\mathsf{EQ}}(\lambda, \ell)$
$(M^*, \sigma^*_{\mathsf{EQ}}) \leftarrow \mathcal{A}^{\mathcal{O}_1(\mathsf{sk}_{\mathsf{EQ}}, \cdot)}(\mathsf{pk}_{\mathsf{EQ}})$
return $[M^*] \neq [M] \;\; \forall M \in Q \wedge$
$\mathsf{Verify}_{\mathsf{EQ}}(\mathsf{pk}, M^*, \sigma^*_{\mathsf{EQ}}) = 1$

$\mathcal{O}_1(\mathsf{sk}_{\mathsf{EQ}}, M)$

$\sigma \leftarrow \mathsf{Sign}_{\mathsf{EQ}}(\mathsf{sk}_{\mathsf{EQ}}, M)$
$Q := Q \cup \{M\}$
return σ_{EQ}

A $\mathsf{SPS\text{-}EQ}$ *over* $(\mathbb{G}_i^*)^\ell$ *is unforgeable if for all PPT adversaries* $\mathcal{A}$*, their advantage defined as* $\mathbf{Adv}_{\mathsf{SPS\text{-}EQ}}(\mathcal{A}) = \Pr[\mathsf{EUF\text{-}CMA}_{\mathcal{A},\mathsf{SPS\text{-}EQ}}(\lambda, \ell) = 1]$ *is negligible.*

Definition 10 (Perfect adaption of signatures under malicious keys [23]). *Let* $\ell > 1$*. A* $\mathsf{SPS\text{-}EQ}$ *scheme on* $(\mathbb{G}_i^*)^\ell$ *perfectly adapts signatures under malicious keys if for all tuples* $(\mathsf{pk}_{\mathsf{EQ}}, M, \sigma_{\mathsf{EQ}}, \mu)$ *with*

$$M \in (\mathbb{G}_i^*)^\ell \quad \wedge \quad \mathsf{Verify}_{\mathsf{EQ}}(M, \sigma_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}}) = 1 \quad \wedge \quad \mu \in \mathbb{Z}_p^*$$

we have that the output of $\mathsf{ChgRep}_{\mathsf{EQ}}(M, \sigma_{\mathsf{EQ}}, \mu, \mathsf{pk}_{\mathsf{EQ}})$ *is a uniformly random element in the space of signatures, conditioned on* $\mathsf{Verify}_{\mathsf{EQ}}(M^\mu, \sigma'_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}}) = 1$.

In this work, we will use the scheme presented by Fuchsbauer, Hanser, and Slamanig in [24]. A signature on a message $(M_1, \ldots, M_\ell) \in (\mathbb{G}_1^*)^\ell$ is of the form (Z, Y_1, Y_2) where $Z = \left(\prod_{i=1}^{\ell} (M_i)^{x_i}\right)^y$, $Y_1 = g_1^{1/y}$, $Y_2 = g_2^{1/y}$ and $x_1, \ldots, x_\ell$ is the secret key of the signer. The signatures can be adapted to a signature on message $(M_1^b, \ldots, M_\ell^b)$ using random coins $r, b \leftarrow \mathbb{Z}_p^*$ and computing $(Z^{r \cdot b}, Y_1^{1/r}, Y_2^{1/r})$.

2.6 Tag-Based Equivalence Class Signatures

Hanzlik and Slamanig [28] introduced the notion of tag-based equivalence class signatures (TBEQ). Additionally, to the message M being a representative of class $[M]$, the signature scheme support an auxiliary tag $\tau \in \{0,1\}^*$. The key idea is that the tag remains the same for a given signature and does not change with the change of the representation. They also propose an efficient instantiation of their scheme, which is an adaptation of scheme from [24] with an additional component $H(\tau)^{\frac{1}{y}}$ in σ_{EQ}. We define TBEQ more formally below.

Definition 11 (TBEQ). *Tag-Based Equivalence Class Signature* TBEQ *consists of the following PPT algorithms.*

$\mathsf{KeyGen}_{\mathsf{TEQ}}(\lambda, \ell)$**:** *On input of security parameters* λ *and message vector length* $\ell > 1$*, it outputs a key pair* $(\mathsf{sk}_{\mathsf{TEQ}}, \mathsf{pk}_{\mathsf{TEQ}})$.

$\mathsf{Sign}_{\mathsf{TEQ}}(\mathsf{sk}_{\mathsf{TEQ}}, M, \tau)$**:** *On input of a secret key* $\mathsf{sk}_{\mathsf{TEQ}}$*, representative* $M \in (\mathbb{G}_i^*)^\ell$*, and tag* $\tau \in \{0,1\}^*$*, outputs a signature* σ_{TEQ} *for equivalence class* $[M]$.

$\mathsf{ChgRep}_{\mathsf{TEQ}}(M, \sigma_{\mathsf{TEQ}}, \mu, \mathsf{pk})$**:** *On input of representative* $M \in (\mathbb{G}_i^*)^\ell$ *of equivalence class* $[M]$*, a signature* σ_{TEQ} *on* M*, a value* μ *and a public key* $\mathsf{pk}_{\mathsf{TEQ}}$*, returns an updated message-signature pair* (M', σ')*, where the new representative is* $M' = M^\mu$ *and* σ'_{TEQ} *its corresponding (or, updated) signature.*

$\mathsf{Verify}_{\mathsf{TEQ}}(\mathsf{pk}_{\mathsf{TEQ}}, M, \tau, \sigma_{\mathsf{TEQ}})$**:** *On input of a public key* $\mathsf{pk}_{\mathsf{TEQ}}$*, a representative* $M \in (\mathbb{G}_i^*)^\ell$*, tag* $\tau \in \{0,1\}^*$ *and a signature* σ_{TEQ} *it deterministically outputs a bit* $b \in \{0,1\}$.

$\mathsf{VKey}_{\mathsf{TEQ}}(\mathsf{sk}_{\mathsf{TEQ}}, \mathsf{pk}_{\mathsf{TEQ}})$**:** *On input secret key* $\mathsf{sk}_{\mathsf{TEQ}}$ *and a public key* $\mathsf{pk}_{\mathsf{TEQ}}$*, it deterministically checks if it is a valid key pair and outputs a bit* $b \in \{0,1\}$.

Definition 12 (EUF-CMA). *For scheme* TBEQ *and adversary* $\mathcal{A}$ *we define the following experiment:*

$\mathsf{EUF\text{-}CMA}_{\mathcal{A},\mathsf{TBEQ}}(\lambda, \ell)$

$Q := \emptyset$

$(\mathsf{sk}_{\mathsf{TEQ}}, \mathsf{pk}_{\mathsf{TEQ}}) \leftarrow \mathsf{KeyGen}(\lambda, \ell)$

$(M^*, \sigma^*_{\mathsf{TEQ}}, \tau^*) \leftarrow \mathcal{A}^{\mathcal{O}_1(\mathsf{sk}_{\mathsf{TEQ}}, \cdot, \cdot)}(\mathsf{pk}_{\mathsf{TEQ}})$

return $\mathsf{Verify}(\mathsf{pk}_{\mathsf{TEQ}}, M^*, \tau^*, \sigma^*_{\mathsf{TEQ}}) = 1 \wedge$
$([M^*], \tau^*) \neq ([M], \tau) \quad \forall (M, \tau) \in Q$

$\mathcal{O}_1(\mathsf{sk}, M, \tau)$

$\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, M, \tau)$

$Q := Q \cup \{(M, \tau)\}$

return σ

A TBEQ *is EUF-CMA, secure if for all PPT adversaries* $\mathcal{A}$*, their advantage defined as* $\mathbf{Adv}_{\mathsf{TBEQ}}(\mathcal{A}) = \Pr[\mathsf{EUF\text{-}CMA}_{\mathcal{A},\mathsf{TBEQ}}(\lambda, \ell) = 1]$ *is negligible.*

3 Non-interactive Blind Signatures (NIBS)

We will now discuss the syntax and security of non-interactive blind signatures NIBS. The signer uses the recipient's public key pk_R to generate a presignature psig. To do so, the signer first creates a signing keypair $(\mathsf{sk}, \mathsf{pk})$ using the KeyGen algorithm. The idea is for the recipient's public key to be a key for a scheme that is independent of NIBS. However, to model the security definition, we need to introduce a key generation algorithm RKeyGen that outputs a keypair $(\mathsf{sk}_R, \mathsf{pk}_R)$.

To issue a presignature, the signer uses the Issue algorithm that takes as input the secret key sk, the recipient's public key pk_R, and a nonce. The nonce allows the signer to issue multiple signatures for the same public key. We made the nonce an explicit parameter to model what we call *nonce blindness* that captures the unlinkability of NIBS issued to the same public key. After receiving a presignature the user can execute the Obtain algorithm and compute the final signature or output $\perp$ in case the presignature is invalid (e.g., issued for a different public key or nonce). We provide the syntax more formally in Definition 13.

Definition 13 (Non-interactive Blind Signature). *A non-interactive blind signature* NIBS *scheme consists of the following PPT algorithms.*

$\mathsf{KeyGen}(\lambda)$**:** *On input security parameter* λ*, outputs a key pair* $(\mathsf{sk}, \mathsf{pk})$*.*

$\mathsf{RKeyGen}(\lambda)$**:** *On input security parameter* λ*, outputs a key pair* $(\mathsf{sk}_R, \mathsf{pk}_R)$*.*

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce})$**:** *On input a secret key* sk*, user public key* pk_R *and nonce* $\mathsf{nonce} \in \mathcal{N}$*, outputs a pre-signature* psig*.*

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}, \mathsf{nonce})$**:** *On input a user secret key* sk_R*, signer's public key* pk*, pre-signature* psig *and nonce* $\mathsf{nonce} \in N$*, outputs a message-signature pair* $(\mathsf{m}, \mathsf{sig})$ *or* $\perp$*.*

$\mathsf{Verify}(\mathsf{pk}, (\mathsf{m}, \mathsf{sig}))$**:** *On input a public key* pk*, a message-signature pair* (m, σ) *deterministically outputs a bit* $b \in \{0, 1\}$*.*

Similar to standard blind signatures, one can define NIBS *with respect to a common reference string. In such a case, we would define a* $\mathsf{crs}_{\mathsf{NIBS}} \leftarrow\!\!\$\ \mathsf{Setup}(\lambda)$ *setup algorithm, where* $\mathsf{crs}_{\mathsf{NIBS}}$ *becomes an implicit input to all other algorithms.*

Definition 14 (Correctness). *A* NIBS *scheme is called correct if for all security parameters* λ*,* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\lambda)$*,* $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{RKeyGen}(\lambda)$*,* nonce*:*

$$\Pr\left[\mathsf{Verify}(\mathsf{pk}, \mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce}), \mathsf{nonce})) = 1\right] = 1.$$

We model unforgeability using a standard one-more definition. The adversary is allowed to make any number k of signing queries but, in the end, must return at least $\ell = k + 1$ valid message-signature pairs for unique messages. The main difference in our definition is that we allow the adversary to specify the recipient's public key and the nonce. We discuss one-more unforgeability more formally in Definition 15.

Definition 15 (One-More Unforgeability). *For scheme* NIBS *and adversary* $\mathcal{A}$ *we define the following experiment:*

$\mathsf{OM\text{-}UNF}_{\mathcal{A},\mathsf{NIBS}}(\lambda)$

$(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(\lambda)$

$((\mathsf{m}_1,\mathsf{sig}_1),\ldots,(\mathsf{m}_\ell,\mathsf{sig}_\ell)) \leftarrow \mathcal{A}^{\mathcal{O}_1(\mathsf{sk},\cdot,\cdot)}(\mathsf{pk})$

return $\mathsf{m}_i \neq \mathsf{m}_j$ *for* $1 \leq i < j \leq \ell \wedge$
$\mathsf{Verify}(\mathsf{pk},\mathsf{m}_i,\mathsf{sig}_i) = 1$ *for* $1 \leq i \leq \ell \wedge$
$k < \ell$

$\mathcal{O}_1(\mathsf{sk},\mathsf{pk}_R,\mathsf{nonce})$

if k *not initialized* **then**
$k := 0$

$\mathsf{psig} \leftarrow \mathsf{Issue}(\mathsf{sk},\mathsf{pk}_R,\mathsf{nonce})$

$k := k + 1$

return psig

A NIBS *scheme is one-more unforgeable, if for all PPT adversaries* $\mathcal{A}$*, their advantage defined as* $\mathbf{Adv}^{\mathsf{OM\text{-}UNF}}_{\mathcal{A},\mathsf{NIBS}} = \Pr[\mathsf{OM\text{-}UNF}_{\mathcal{A},\mathsf{NIBS}}(\lambda) = 1]$ *is negligible.*

We will now discuss the blindness properties of our non-interactive blind signatures. On a high level, we want presignatures and signatures to be unlinkable for the signer, independent of which public keys and nonces were used. We introduce two security definitions, recipient blindness and nonce blindness, which capture this intuition formally. We will also use the notion of *full blindness* to define a non-interactive blind signature scheme that is recipient and nonce blind.

In the case of nonce blindness, we consider the scenario of a malicious signer trying to distinguish who was the original recipient of a signature it sees. To do so formally, we create an experiment where the adversary is given two unique public keys and issues two presignatures (psig_0 and psig_1) for potentially different nonces and a public key that it can choose maliciously. We will also consider a variant called the honest key model, where the adversary must additionally return the secret signing key that matches the returned public key. In the experiment, the challenger finalizes both presignatures and gives the finalized signatures $(\mathsf{sig}_b,\mathsf{sig}_{1-b})$ to the adversary. The order of the signatures provided to the adversary depends on a bit b, which the adversary must guess. If the Obtain algorithm outputs $\perp$ for one of the presignatures, the challenger sends $(\perp,\perp)$ to omit simple distinguishing attacks. The experiment is defined more formally in Fig. 1 and recipient blindness in Definition 23.

Recipient blindness considers only single signatures issued to a particular public key. To create more signatures for the same public key, we introduced the explicit parameter nonce. We will now look at a scenario where the signer issues several presignatures to the same public key under different nonces and later wants to link the presignatures to the final signatures. We formalize it with the notion of nonce blindness. We create an experiment similar to the above one. The adversary is given one public key and issues two presignatures for two unique nonces. Again, the challenger finalizes both presignatures and gives signatures $(\mathsf{sig}_b,\mathsf{sig}_{1-b})$ to the adversary. The adversary must guess bit b. The experiment is defined more formally in Fig. 1 and nonce blindness in Definition 24

Finally, we define full blindness as the combination of both definitions. In other words, if a scheme is recipient and nonce blind, then it is fully blind. The intuition behind that follows from a hybrid argument. Recipient blindness ensures that signatures issued to different public keys are unlinkable, independent of the nonce used. On the other hand, nonce blindness ensures that multiple signatures for the same public key are unlinkable.

$\mathsf{RBnd}_{\mathcal{A},\mathsf{NIBS}}(\lambda)$

$(\mathsf{sk}_{R_0}, \mathsf{pk}_{R_0}) \leftarrow \mathsf{RKeyGen}(\lambda)$
$(\mathsf{sk}_{R_1}, \mathsf{pk}_{R_1}) \leftarrow \mathsf{RKeyGen}(\lambda)$
$(\mathsf{psig}_0, \mathsf{nonce}_0, \mathsf{psig}_1, \mathsf{nonce}_1, \mathsf{pk}) \leftarrow \mathcal{A}(\mathsf{pk}_{R_0}, \mathsf{pk}_{R_1})$
$(\mathsf{m}_0, \mathsf{sig}_0) \leftarrow \mathsf{Obtain}(\mathsf{sk}_{R_0}, \mathsf{pk}, \mathsf{psig}_0, \mathsf{nonce}_0)$
$(\mathsf{m}_1, \mathsf{sig}_1) \leftarrow \mathsf{Obtain}(\mathsf{sk}_{R_1}, \mathsf{pk}, \mathsf{psig}_1, \mathsf{nonce}_1)$
if $\mathsf{sig}_0 = \perp$ or $\mathsf{sig}_1 = \perp$ **then**
  $(\mathsf{m}_0, \mathsf{sig}_0) := \perp; (\mathsf{m}_1, \mathsf{sig}_1) := \perp$
$b \leftarrow_\$ \{0,1\}$
$\hat{b} \leftarrow \mathcal{A}((\mathsf{m}_b, \mathsf{sig}_b), (\mathsf{m}_{1-b}, \mathsf{sig}_{1-b}))$
return $b = \hat{b}$

$\mathsf{NBnd}_{\mathcal{A},\mathsf{NIBS}}(\lambda)$

$(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{RKeyGen}(\lambda)$
$(\mathsf{psig}_0, \mathsf{nonce}_0, \mathsf{psig}_1, \mathsf{nonce}_1, \mathsf{pk}) \leftarrow \mathcal{A}(\mathsf{pk}_R)$
$(\mathsf{m}_0, \mathsf{sig}_0) \leftarrow \mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}_0, \mathsf{nonce}_0)$
$(\mathsf{m}_1, \mathsf{sig}_1) \leftarrow \mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}_1, \mathsf{nonce}_1)$
if $\mathsf{sig}_0 = \perp$ or $\mathsf{sig}_1 = \perp$ **then**
  $(\mathsf{m}_0, \mathsf{sig}_0) := \perp; (\mathsf{m}_1, \mathsf{sig}_1) := \perp$
$b \leftarrow_\$ \{0,1\}$
$\hat{b} \leftarrow \mathcal{A}((\mathsf{m}_b, \mathsf{sig}_b), (\mathsf{m}_{1-b}, \mathsf{sig}_{1-b}))$
return $b = \hat{b}$

Fig. 1. Blindness Experiments for Non-interactive Blind Signatures

Definition 16 (Recipient Blindness). *A* NIBS *scheme is recipeint blind, if for all PPT adversaries $\mathcal{A}$, their advantage is negligible:*

$$\mathbf{Adv}^{\mathsf{RBnd}}_{\mathcal{A},\mathsf{NIBS}} = |\Pr[\mathsf{RBnd}_{\mathcal{A},\mathsf{NIBS}}(\lambda) = 1] - 1/2|.$$

Definition 17 (Nonce Blindness). *A* NIBS *scheme is nonce blind, if for all PPT adversaries $\mathcal{A}$, their advantage is negligible:*

$$\mathbf{Adv}^{\mathsf{NBnd}}_{\mathcal{A},\mathsf{NIBS}} = |\Pr[\mathsf{NBnd}_{\mathcal{A},\mathsf{NIBS}}(\lambda) = 1] - 1/2|.$$

Definition 18 (Full Blindness). *A* NIBS *scheme is fully blind if it is recipient and nonce blind.*

In both cases, we define blindness in a way that the adversary returns just the public key pk. The definitions do not assume any particular structure of the public key. Moreover, they allow the adversary to choose the public key so that a corresponding secret key sk might not even exist. We call this notion *malicious key model*. We will define a weaker version of blindness, where we will require the adversary to output sk additionally. This notion is called *honest key model* and is known for the case of standard blind signatures. Below we will define it more formally.

Definition 19 (Honest Key Model). *A* NIBS *scheme is recipient blind in the honest key model, respectively nonce blind in the honest key model if the adversary outputs* $(\mathsf{psig}_0, \mathsf{nonce}_0, \mathsf{psig}_1, \mathsf{nonce}_1, \mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{A}(\mathsf{pk}_{R_0}, \mathsf{pk}_{R_1})$ *in experiment* $\mathsf{RBnd}_{\mathcal{A},\mathsf{NIBS}}$*, respectively outputs* $(\mathsf{psig}_0, \mathsf{nonce}_0, \mathsf{psig}_1, \mathsf{nonce}_1, \mathsf{sk}, \mathsf{pk})$ *in experiment* $\mathsf{NBnd}_{\mathcal{A},\mathsf{NIBS}}$

Remark 1. Any honest key can be transformed into a malicious key blind non-interactive blind signature using a zero-knowledge proof of knowledge of the secret key sk. In this transformation, the pubic key pk' for the malicious key

blind scheme is composed of the old public key pk and the proof of possession of the secret key sk in the form of a proof of knowledge. The security reduction follows by extracting the secret key from the adversary's proof of possession and running the reduction for honest key blindness.

4 Tagged NIBS

Partially blind signatures [2] allow the signer and recipient to agree on some common information that is included as part of the signed message. The signer knows that the user cannot change this information. At the same time, the recipient is assured that blindness holds with respect to this information. Since both parties agree on the message, partially blind signatures are, in some sense, interactive by definition.

We will show how to adapt the partially blind notion to the non-interactive case. The common information will be only chosen by the signer, which might limit the application compared to partially blind signatures. However, we show that this is enough for protocols that require some kind of freshness nonce to be included in the signature. To distinguish that in case of NIBS only the signer chooses the common information, we will call our primitive tagged NIBS. The main changes in the syntax in comparison to standard NIBS are that the Issue, Obtain, and Verify take an additional input in the form of the tag τ.

Definition 20 (Tagged Non-interactive Blind Signature). *A tagged non-interactive blind signature scheme* TNIBS *consists of the following PPT algorithms.*

$\mathsf{KeyGen}(\lambda)$**:** *On input security parameter* λ*, outputs a key pair* $(\mathsf{sk}, \mathsf{pk})$.
$\mathsf{RKeyGen}(\lambda)$**:** *On input security parameter* λ*, outputs a key pair* $(\mathsf{sk}_R, \mathsf{pk}_R)$.
$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce}, \tau)$**:** *On input a secret key* sk*, user public key* pk_R*, nonce* $\mathsf{nonce} \in \mathcal{N}$*, and tag* $\tau \in \mathcal{T}$*, outputs a pre-signature* psig.
$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}, \mathsf{nonce}, \tau)$**:** *On input a user secret key* sk_R*, signer's public key* pk*, pre-signature* psig *, nonce* $\mathsf{nonce} \in \mathcal{N}$ *and tag* $\tau \in \mathcal{T}$*, outputs the tuple* $(\mathsf{m}, \tau, \mathsf{sig})$ *or* $\perp$.
$\mathsf{Verify}(\mathsf{pk}, (\mathsf{m}, \tau, \mathsf{sig}))$**:** *On input a public key* pk*, a message* m*, tag* $\tau \in \mathcal{T}$ *and signature* σ *deterministically outputs a bit* $b \in \{0, 1\}$.

Definition 21 (Correctness). *A* TNIBS *scheme on is called correct if for all security parameters* λ*,* $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{KeyGen}(\lambda)$*,* $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{RKeyGen}(\lambda)$*,* $\mathsf{nonce} \in \mathcal{N}$*,* $\tau \in \mathcal{T}$*:*

$$\Pr\left[\mathsf{Verify}(\mathsf{pk}, \mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce}, \tau), \mathsf{nonce}, \tau)) = 1\right] = 1.$$

To define one-more unforgeability for our tagged NIBS we need to change the signing oracle. We now allow the adversary to query it with the recipient's public key pk_R, a nonce nonce, and a tag τ. We say that an adversary succeeded in breaking unforgeability if it returns at least $\ell = k_\tau + 1$ valid signatures on unique messages and only queried the signing oracle k_τ times for a given tag τ. More details are given in Definition 22.

Definition 22 (One-More Unforgeability). *For scheme tagged* TNIBS *and adversary* $\mathcal{A}$ *we define the following experiment:*

$\mathsf{OM\text{-}UNF}_{\mathcal{A},\mathsf{TNIBS}}(\lambda)$

$(\mathsf{sk},\mathsf{pk}) \leftarrow \mathsf{KeyGen}(\lambda)$

$(\tau,(\mathsf{m}_1,\mathsf{sig}_1),\ldots,(\mathsf{m}_\ell,\mathsf{sig}_\ell)) \leftarrow \mathcal{A}^{\mathcal{O}_1(\mathsf{sk},\cdot,\cdot,\cdot)}(\mathsf{pk})$

return $\mathsf{m}_i \neq \mathsf{m}_j$ *for* $1 \leq i < j \leq \ell \wedge$
$\mathsf{Verify}(\mathsf{pk},(\mathsf{m}_i,\tau,\mathsf{sig}_i)) = 1$ *for* $1 \leq i \leq \ell \wedge$
$k_\tau < \ell$

$\mathcal{O}_1(\mathsf{sk},\mathsf{pk}_R,\mathsf{nonce},\tau)$

if k_τ *not initialized* **then**
$k_\tau := 0$
$\mathsf{psig} \leftarrow \mathsf{Issue}(\mathsf{sk},\mathsf{pk}_R,\mathsf{nonce},\tau)$
$k_\tau := k_\tau + 1$
return psig

A TNIBS *scheme is one-more unforgeable, if for all PPT adversaries* $\mathcal{A}$*, their advantage defined as* $\mathbf{Adv}^{\mathsf{OM\text{-}UNF}}_{\mathcal{A},\mathsf{TNIBS}} = \Pr[\mathsf{OM\text{-}UNF}_{\mathcal{A},\mathsf{TNIBS}}(\lambda) = 1]$ *is negligible.*

We will now move our attention to the blindness definitions of tagged non-interactive blind signatures. As we already mentioned, blindness can only hold with respect to the same tag. Since the tag is chosen by the signer and cannot be changed, it is additional information that can be used to distinguish if two signatures were signed under the same tag or not. The experiments for the blindness definitions are defined formally in Fig. 2.

$\mathsf{RBnd}_{\mathcal{A},\mathsf{TNIBS}}(\lambda)$

$(\mathsf{sk}_{R_0},\mathsf{pk}_{R_0}) \leftarrow \mathsf{RKeyGen}(\lambda)$
$(\mathsf{sk}_{R_1},\mathsf{pk}_{R_1}) \leftarrow \mathsf{RKeyGen}(\lambda)$
$(\mathsf{psig}_0,\mathsf{nonce}_0,\mathsf{psig}_1,\mathsf{nonce}_1,\mathsf{pk},\tau) \leftarrow \mathcal{A}(\mathsf{pk}_{R_0},\mathsf{pk}_{R_1})$
$(\mathsf{m}_0,\mathsf{sig}_0) \leftarrow \mathsf{Obtain}(\mathsf{sk}_{R_0},\mathsf{pk},\mathsf{psig}_0,\mathsf{nonce}_0,\tau)$
$(\mathsf{m}_1,\mathsf{sig}_1) \leftarrow \mathsf{Obtain}(\mathsf{sk}_{R_1},\mathsf{pk},\mathsf{psig}_1,\mathsf{nonce}_1,\tau)$
if $\mathsf{sig}_0 = \perp$ or $\mathsf{sig}_1 = \perp$ **then**
$(\mathsf{m}_0,\mathsf{sig}_0) := \perp; (\mathsf{m}_1,\mathsf{sig}_1) := \perp$
$b \leftarrow_\$ \{0,1\}$
$\hat{b} \leftarrow \mathcal{A}((\mathsf{m}_b,\mathsf{sig}_b),(\mathsf{m}_{1-b},\mathsf{sig}_{1-b}))$
return $b = \hat{b}$

$\mathsf{NBnd}_{\mathcal{A},\mathsf{TNIBS}}(\lambda)$

$(\mathsf{sk}_R,\mathsf{pk}_R) \leftarrow \mathsf{RKeyGen}(\lambda)$
$(\mathsf{psig}_0,\mathsf{nonce}_0,\mathsf{psig}_1,\mathsf{nonce}_1,\mathsf{pk},\tau) \leftarrow \mathcal{A}(\mathsf{pk}_R)$
$(\mathsf{m}_0,\mathsf{sig}_0) \leftarrow \mathsf{Obtain}(\mathsf{sk}_R,\mathsf{pk},\mathsf{psig}_0,\mathsf{nonce}_0,\tau)$
$(\mathsf{m}_1,\mathsf{sig}_1) \leftarrow \mathsf{Obtain}(\mathsf{sk}_R,\mathsf{pk},\mathsf{psig}_1,\mathsf{nonce}_1,\tau)$
if $\mathsf{sig}_0 = \perp$ or $\mathsf{sig}_1 = \perp$ **then**
$(\mathsf{m}_0,\mathsf{sig}_0) := \perp; (\mathsf{m}_1,\mathsf{sig}_1) := \perp$
$b \leftarrow_\$ \{0,1\}$
$\hat{b} \leftarrow \mathcal{A}((\mathsf{m}_b,\mathsf{sig}_b),(\mathsf{m}_{1-b},\mathsf{sig}_{1-b}))$
return $b = \hat{b}$

Fig. 2. Blindness Experiments for Tagged Non-interactive Blind Signatures

Definition 23 (Recipient Blindness). *A* TNIBS *scheme is recipeint blind, if for all PPT adversaries* $\mathcal{A}$*, their advantage is negligible:*

$$\mathbf{Adv}^{\mathsf{RBnd}}_{\mathcal{A},\mathsf{TNIBS}} = |\Pr[\mathsf{RBnd}_{\mathcal{A},\mathsf{TNIBS}}(\lambda) = 1] - 1/2|.$$

Definition 24 (Nonce Blindness). *A* TNIBS *scheme is nonce blind, if for all PPT adversaries* $\mathcal{A}$*, their advantage is negligible:*

$$\mathbf{Adv}^{\mathsf{NBnd}}_{\mathcal{A},\mathsf{TNIBS}} = |\Pr[\mathsf{NBnd}_{\mathcal{A},\mathsf{TNIBS}}(\lambda) = 1] - 1/2|.$$

Definition 25 (Full Blindness). *A* TNIBS *scheme is fully blind if it is recipient and nonce blind.*

5 SPS-EQ Construction of NIBS

In this section, we present an efficient construction of non-interactive blind signatures from signatures on equivalence classes. The main advantage of our solution is that it admits recipients' public keys of the form used by many discrete logarithm schemes. The idea of the construction is as follows. The signer uses a signature on equivalence classes to create a presignature psig on the vector $(\mathsf{pk}_R = g_1^{\mathsf{sk}_R}, \mathsf{H}(\mathsf{nonce}))$, where H is a hash function modeled as a random oracle. The recipient knowing the secret key sk_R can change the presignatures representation to a SPS-EQ signature on the vector $(g_1, \mathsf{H}(\mathsf{nonce})^{\mathsf{sk}_R^{-1}})$ which is returned as the final signature sig. In the end, a valid blind signature is SPS-EQ on a vector of messages where the first element is g_1 and the second element is $\mathsf{H}(\mathsf{nonce})^{\mathsf{sk}_R^{-1}}$. Full blindness of the construction relies on the fact that the value $\mathsf{H}(\mathsf{nonce})^{\mathsf{sk}_R^{-1}}$ is indistinguishable from a random element under the inverse decisional Diffie-Hellman assumption. One more unforgeability follows directly from the unforgeability of SPS-EQ. The construction is presented in detail in Scheme 1. Note that the scheme is only blind in the honest key model since we rely on the perfect adaptation of signatures, i.e., to prove security, the reduction will use the SPS-EQ signing key to resign messages. We could use the DMWI proof system and its extraction property, which, based on Remark 1, would allow us to transform the scheme into one secure in the malicious key model. We opted to present it this way because of two reasons. Firstly, it simplifies the presentation and shows the essence of our construction. Secondly, we want an efficient scheme, and using a proof system for NP languages would be impractical. We will later show in the discussion section that for the SPS-EQ from [24], proof of knowledge of the signing key can be done via ℓ proofs of knowledge of discrete logarithms in $\mathbb{G}_2$. Note that in our scheme $\ell = 2$, which shows that our instantiation can be easily transformed into the malicious key model.

Security

Theorem 1 (One-more Unforgeability). *Scheme 1 is one-more unforgeable in the random oracle model assuming the* SPS-EQ *scheme is existentially unforgeable under adaptively chosen-message attacks.*

Proof (Sketch). The proof follows by a straightforward reduction to the security of the SPS-EQ. The reduction uses the provided signing oracle to generate

$\underline{\mathsf{KeyGen}(\lambda)}$**:** generate SPS-EQ keypair $(\mathsf{pk}_{\mathsf{EQ}}, \mathsf{sk}_{\mathsf{EQ}}) \leftarrow \mathsf{KeyGen}_{\mathsf{EQ}}(\lambda, 2)$ and set $(\mathsf{sk}, \mathsf{pk}) := (\mathsf{sk}_{\mathsf{EQ}}, \mathsf{pk}_{\mathsf{EQ}})$.

$\underline{\mathsf{RKeyGen}(\lambda)}$**:** choose $x \leftarrow_{\$} \mathbb{Z}_p^*$. Set $\mathsf{sk}_R := x$ and $\mathsf{pk}_R := g_1^x$.

$\underline{\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce})}$**:** generate SPS-EQ signature $\mathsf{psig} \leftarrow \mathsf{Sign}_{\mathsf{EQ}}(\mathsf{sk}, (\mathsf{pk}_R, \mathsf{H}(\mathsf{nonce})))$.

$\underline{\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}, \mathsf{nonce})}$**:** output $\perp$ if $\mathsf{Verify}_{\mathsf{EQ}}(\mathsf{pk}, (\mathsf{pk}_R, \mathsf{H}(\mathsf{nonce})), \mathsf{psig}) = 0$, otherwise adapt presignature $\mathsf{sig} \leftarrow \mathsf{ChgRep}_{\mathsf{EQ}}((\mathsf{pk}_R, \mathsf{H}(\mathsf{nonce})), \mathsf{psig}, \mathsf{sk}_R^{-1}, \mathsf{pk})$ output message-signature pair $(\mathsf{m} = \mathsf{H}(\mathsf{nonce})^{\mathsf{sk}_{\mathcal{R}}^{-1}}, \mathsf{sig})$.

$\underline{\mathsf{Verify}(\mathsf{pk}, (\mathsf{m}, \mathsf{sig}))}$**:** output $\mathsf{Verify}_{\mathsf{EQ}}(\mathsf{pk}, (g_1, \mathsf{m}), \mathsf{sig})$.

Scheme 1: SPS-EQ Construction of NIBS

presignatures for the adversary's $\mathcal{A}$ queries. Finally, the adversary outputs ℓ valid message-signature pairs for the NIBS scheme, simultaneously making only q_s signing queries. Without loss of generality, we can assume that $\ell = q_s + 1$ (otherwise, the reduction omits the additional message-signature pairs). Note that since all ℓ pairs are signatures under unique messages, it follows that they also belong to separate equivalence classes due to the notion of canonical representative. Thus, the adversary returned SPS-EQ message-signature pair for ℓ different classes, while the reduction only queries the SPS-EQ signing oracle $\ell - 1$ times. However, because of the hiding property, the reduction cannot guess the class for which it did not query the SPS-EQ signing oracle, i.e., the forgery. So the only way to win the unforgeability experiment is for the reduction to choose one message-signature pair at random. With probability $1/\ell$, this guess will be correct, and the pair will be a valid forgery against the SPS-EQ scheme. The complete proof can be found in the full version of the paper.

Theorem 2 (Recipient Blindness). *Scheme 1 is recipient blind (in the honest key model) in the random oracle model assuming the inverse decision Diffie-Hellman assumption holds in* $\mathbb{G}_1$ *and that the* SPS-EQ *scheme perfectly adapts signatures under a malicious signer.*

Proof (Sketch). The idea behind the proof is to make the challenged messages $\mathsf{m}_0, \mathsf{m}_1$ and corresponding signatures $\mathsf{sig}_0, \mathsf{sig}_1$ independent of the public keys $\mathsf{pk}_{R_0}, \mathsf{pk}_{R_1}$. We achieve this by making indistinguishable changes to how the recipient blindness experiment generates them. Firstly, the reduction program the random oracle H so that for all queried nonces nonce the reductions know r_{nonce}, such that $\mathsf{H}(\mathsf{nonce}) = g_1^{r_{\mathsf{nonce}}}$. It replaces the public key of one of the recipients with g_1^{α}, where $(g_1^{\alpha}, g_1^{\beta})$ is an instance of the inverse decisional Diffie-Hellman problem. Thanks to the programming of the oracle, the reduction can compute the messages as $(g_1, (g_1^{\beta})^{r_{\mathsf{nonce}}})$, without knowing the recipient's secret key. The reduction uses the known SPS-EQ signing key and perfect adaptation

of signatures to resign the presignature. If $(g_1^{\alpha}, g_1^{\beta})$ is an inverse decisional Diffie-Hellman tuple, then the reductions simulation is perfect. This way, the reduction can change the messages signed by the challenged signatures $\mathsf{sig}_0, \mathsf{sig}_1$ to be independent of the public keys $\mathsf{pk}_{R_0}, \mathsf{pk}_{R_1}$. Thus, the best an adversary can do is to guess the bit b in the experiment. The complete proof can be found in the full version of the paper.

Theorem 3 (Nonce Blindness). *Scheme 1 is nonce blind (in the honest key model) in the random oracle model assuming the strong decision Diffe-Hellman assumption holds in* $\mathbb{G}_1$ *and that the* SPS-EQ *scheme perfectly adapts signatures under a malicious signer.*

Proof (Sketch). The proof follows a blueprint similar to the above one. The main difference is that now given a strong decisional Diffie-Hellman instance $(g_1^{\alpha}, g_1^{\beta}, g_1^{\beta^{-1}}, g_1^{\gamma})$, we set the recipient's public key to $g_1^{\beta^{-1}}$. The reduction programs the oracle H similarly but tries to guess the query $\mathsf{H}(\mathsf{nonce}_0)$ to program it to g_1^{α}. The programming allows the reduction to compute the message m_0 as (g_1, g_1^{γ}). Note that if $\gamma = \alpha \cdot \beta$, the simulation is perfect, and the reduction can use an adversary noticing that m_0 is computed incorrectly to solve the strong decisional Diffie-Hellman problem. The reduction can make message m_0 and signature sig_0 independent of nonce_0. It can use the same strategy to make m_1 and sig_1 independent of nonce_1. Finally, the adversary is given only messages and signatures independent of nonce_0, nonce_1. The best it can do is guess the bit $\bar{b}$. The complete proof can be found in the full version of the paper.

5.1 Tagged NIBS from TBEQ

Scheme 1 can be easily transformed into a TNIBS. The only change is to replace the standard structure-preserving signature scheme with the tagged version TBEQ. For completeness, we present Scheme 2.

Security

All proofs follow the same strategy as the corresponding ones for Scheme 1. The complete proofs can be found in the full version of the paper.

Theorem 4 (One-more Unforgeability). *Scheme 2 is one-more unforgeable in the random oracle model assuming the* TBEQ *scheme is existentially unforgeable under adaptively chosen-message attacks.*

Theorem 5 (Recipient Blindness). *Scheme 2 is recipient blind (in the honest key model) in the random oracle model assuming the inverse decision Diffe-Hellman assumption holds in* $\mathbb{G}_1$ *and that the* SPS-EQ *scheme perfectly adapts signatures under a malicious signer.*

Theorem 6 (Nonce Blindness). *Scheme 2 is nonce blind (in the honest key model) in the random oracle model assuming the strong decision Diffe-Hellman assumption holds in* $\mathbb{G}_1$ *and that the* SPS-EQ *scheme perfectly adapts signatures under a malicious signer.*

$\underline{\mathsf{KeyGen}(\lambda)}$: generate TBEQ keypair $(\mathsf{pk}_{\mathsf{TEQ}}, \mathsf{sk}_{\mathsf{TEQ}}) \leftarrow \mathsf{KeyGen}_{\mathsf{TEQ}}(\lambda, 2)$ and set $(\mathsf{sk}, \mathsf{pk}) := (\mathsf{sk}_{\mathsf{TEQ}}, \mathsf{pk}_{\mathsf{TEQ}})$.

$\underline{\mathsf{RKeyGen}(\lambda)}$: choose $x \leftarrow_{\$} \mathbb{Z}_p^*$. Set $\mathsf{sk}_R := x$ and $\mathsf{pk}_R := g_1^x$.

$\underline{\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce}, \tau)}$: return presignature $\mathsf{psig} \leftarrow \mathsf{Sign}_{\mathsf{TEQ}}(\mathsf{sk}, (\mathsf{pk}_R, \mathsf{H}(\mathsf{nonce})), \tau)$.

$\underline{\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}, \mathsf{nonce}, \tau)}$: output $\perp$ if $\mathsf{Verify}_{\mathsf{TEQ}}(\mathsf{pk}, (\mathsf{pk}_R, \mathsf{H}(\mathsf{nonce})), \tau, \mathsf{psig}) = 0$, otherwise adapt pre-signature $\mathsf{sig} \leftarrow \mathsf{ChgRep}_{\mathsf{TEQ}}((\mathsf{pk}_R, \mathsf{H}(\mathsf{nonce})), \mathsf{psig}, \mathsf{sk}_R^{-1}, \mathsf{pk})$ output message-signature pair $(\mathsf{m} = \mathsf{H}(\mathsf{nonce})^{x^{-1}}, \mathsf{sig})$.

$\underline{\mathsf{Verify}(\mathsf{pk}, (\mathsf{m}, \mathsf{sig}))}$: output $\mathsf{Verify}_{\mathsf{TEQ}}(\mathsf{pk}, (g_1, \mathsf{m}), \tau, \mathsf{sig})$.

Scheme 2: TBEQ Construction of TNIBS

5.2 Discussion

Instantiating NIBS *and* TNIBS. We already mentioned at the beginning of this section that depending on how the SPS-EQ scheme is instantiated in our construction, we can end up with schemes with different properties. An efficient instantiation follows if we use the SPS-EQ scheme from [24]. The used equivalence class signature requires type 3 pairings groups. Barreto, Lynn, and Scott [8] introduced the BLS family of pairing-friendly groups that can be used in this case. We instantiate it with the popular BLS12-381 parameters [11]. In this setting the groups are defines as $\mathbb{G}_1 = E(\mathbb{F}_q)$, $\mathbb{G}_2 = E'(\mathbb{F}_{q^2})$ and $\mathbb{G}_T = \mathbb{F}_{q^{12}}$ for a 381-bit prime q. Consequently, the recipient's public key and the message space are in $\mathbb{G}_1$. The blind signature comprises two elements in $\mathbb{G}_1$ and one in $\mathbb{G}_2$. The signer's public key is two group elements in $\mathbb{G}_2$. Assuming we use the BLS12-381 groups, this constitutes a *signature size of* 1527*-bits*, where the message is an element of $\mathbb{G}_1$ and size 382-bit.

Recipient Public Key. With the above instantiation, the recipient's public key space is set to $\mathbb{G}_1$, where $\mathbb{G}_1$ is a standard elliptic curve. Thus, preexisting public keys for other schemes can be used as the recipient's public key. In particular, we can use public keys for the ECDSA, Schnorr signature scheme, ephemeral keys for the Diffie-Hellman protocol, and keys for the El-Gamal encryption scheme defined over the group $\mathbb{G}_1$ in our instantiation.

Composing the above schemes with NIBS does not seem to introduce security issues. Still, it will require providing proof for a composed primitive. We leave the formal proofs for future work. However, we can reduce the security of those schemes to the strong decisional Diffie-Hellman assumption, allowing the security of NIBS to hold independent of the use of the secret key in the other scheme.

Interestingly, we cannot use a BLS signature scheme public key. Recall that for BLS (for type-3 pairings), the public key g_1^x is in $\mathbb{G}_1$, then signatures are in $\mathbb{G}_2$ and of the form $\mathsf{H}_{\mathbb{G}_2}(m)^x$. The message in our construction for such a

public key would be $\mathsf{m} = \mathsf{H}(\mathsf{nonce})^{x^{-1}}$. A malicious signer could then compute $e(\mathsf{m}, \mathsf{H}_{\mathbb{G}_2}(m)^x)$, for some known BLS signature of the recipient under m, and compare it with $e(\mathsf{H}(\mathsf{nonce}), \mathsf{H}(m))$. How the BLS signature scheme uses the recipient's secret key breaks the blindness properties of NIBS. The intuition is that the assumption used in unforgeability proof of BLS signatures cannot hold simultaneously with the strong decisional Diffie-Hellman assumption.

6 Generic Construction

In the previous section, we presented one NIBS and one TNIBS scheme that can efficiently be instantiated using signatures on equivalence classes and its variant. An interesting observation we make here is that the resulting random message in both of those schemes is $\mathsf{H}(\mathsf{m})^{\mathsf{sk}^{-1}}$. Blindness then follows from the inverse and strong decisional Diffie-Hellman assumptions. However, looking at it more closely, we notice that this is actually a valid evaluation of a pseudo-random function with key sk. Note that $\mathsf{PRF}_{\mathsf{sk}}(\mathsf{m}) := \mathsf{H}(\mathsf{m})^{\mathsf{sk}^{-1}}$ is a known construction. This observation is key to why blindness holds for those schemes. Even though the signer is choosing the input to the function, its evaluation is indistinguishable from random. Let us use this intuition to derive a generic construction.

The main problem is ensuring that given a presignature on the input to the function, the recipient will evaluate it correctly and preserve the signer's signature. In Schemes 1 and 2, this was possible because the relation defined by the equivalence class signatures worked well with the PRF. We can achieve something similar using non-interactive proofs and a verifiable random function VRF. Unfortunately, we will require proofs for NP languages. Additionally, we will use a trapdoor witness that will allow us to simulate this proof, i.e., we will allow for a trapdoor witness that we will be able to use in the proof by programming the random oracle $\mathsf{H}_{\mathsf{crs}}$. Alternatively, instead of using a trapdoor witness to simulate proofs, we can use a DMWI proof system with a trusted setup. Thus, this allows us to rely on the common reference string instead of the random oracle model. The idea of the scheme is as follows. The presignature is a standard digital signature psig on the recipients VRF public key $\mathsf{pk}_{\mathsf{VRF}}$ and the nonce. To obtain a valid signature, the recipient first evaluates the VRF on input nonce to receive message m. Later the recipient creates proof that it knows a signature psig under a key $\mathsf{pk}_{\mathsf{VRF}}$ and nonce nonce and that m is the result of the VRF's evaluation. The actual non-interactive blind signature is then this proof. More details are given in Scheme 3.

Remark 2 (Generic TNIBS*).* We can easily transform Scheme 3 to a tagged version. To make it work, the signer must include the tag τ as the message in the presignature, i.e., we replace $\mathsf{psig} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, (\mathsf{nonce}, \mathsf{pk}_R))$ with $\mathsf{psig} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, (\mathsf{nonce}, \mathsf{pk}_R, \tau))$ and modify relation $\mathcal{R}$ accordingly. In the generic TNIBS version, the tag τ is part of the statement x.

Let $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{P}, \mathsf{V})$ be a verifiable random function, $\mathsf{DMWI} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{Extract})$ be a dual-mode witness indistinguishable proof system for the language $\mathcal{L}_{\mathcal{R}}$ and $\mathsf{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a standard digital signature scheme. Moreover, let $\mathsf{H}_{\mathsf{crs}}$ be a random oracle that, on inputs from $\{0,1\}^*$, outputs elements from the space of reference strings for the DMWI system. Finally, let us define the following relation $\mathcal{R}$:

$$((\mathsf{m}, \mathsf{pk}), (\mathsf{nonce}, \mathsf{psig}, \mathsf{pk}_R, \pi_{\mathsf{VRF}}, r)) \in \mathcal{R} \iff \mathsf{H}_{\mathsf{crs}}(1) = \mathsf{DMWI.Setup}(\lambda, \mathsf{binding}; r) \ \vee \ \mathsf{VRF.V}(\mathsf{pk}_R, \pi_{\mathsf{VRF}}, \mathsf{nonce}, \mathsf{m}) = 1 \ \wedge \ \mathsf{SIG.Verify}(\mathsf{pk}, (\mathsf{pk}_R, \mathsf{nonce}), \mathsf{psig}) = 1$$

$\underline{\mathsf{KeyGen}(\lambda)}$**:** generate keypair $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathsf{SIG.KeyGen}(\lambda)$.

$\underline{\mathsf{RKeyGen}(\lambda)}$**:** generate keypair $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{VRF.Gen}(\lambda)$.

$\underline{\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce})}$**:** create presignature $\mathsf{psig} \leftarrow \mathsf{SIG.Sign}(\mathsf{sk}, (\mathsf{nonce}, \mathsf{pk}_R))$.

$\underline{\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{pk}, \mathsf{psig}, \mathsf{nonce})}$**:** output $\perp$ if $\mathsf{SIG.Verify}(\mathsf{pk}, (\mathsf{nonce}, \mathsf{pk}_R)), \mathsf{psig}) = 0$, otherwise compute $\mathsf{m} \leftarrow \mathsf{VRF.Eval}(\mathsf{sk}_R, \mathsf{nonce})$, compute $\pi_{\mathsf{VRF}} \leftarrow \mathsf{VRF.P}(\mathsf{sk}_R, \mathsf{nonce})$, set statement $x = (\mathsf{m}, \mathsf{pk})$ and witness $w = (\mathsf{nonce}, \mathsf{psig}, \mathsf{pk}_R, \pi_{\mathsf{VRF}}, \cdot)$. Compute blind signature $\mathsf{sig} \leftarrow \mathsf{DMWI.Prove}(\mathsf{H}_{\mathsf{crs}}(0), x, w)$. Output message-signature pair $(\mathsf{m}, \mathsf{sig})$.

$\underline{\mathsf{Verify}(\mathsf{pk}, (\mathsf{m}, \mathsf{sig}))}$**:** Set statement $x = (\mathsf{m}, \mathsf{pk})$ and output $\mathsf{DMWI.Verify}(\mathsf{H}_{\mathsf{crs}}(0), x, \mathsf{sig})$.

Scheme 3: Generic Construction of NIBS

Security

Theorem 7 (One-more Unforgeability). *Scheme 3 is one-more unforgeable in the random oracle model assuming the signature scheme* SIG *is existentially unforgeable under adaptively chosen-message attacks, the dual-mode proof system* DMWI *is mode indistinguishable and extractable in binding mode, and the* VRF *meets the uniqueness property.*

Proof (Sketch). The proof works as follows. We first program the random oracle $\mathsf{H}_{\mathsf{crs}}$ in a way that we can extract the witness used by the adversary, i.e., we set $\mathsf{H}_{\mathsf{crs}}(0)$ to output a string in binding mode. Additionally, we program the oracle so the adversary cannot use the trapdoor witness for $\mathsf{H}_{\mathsf{crs}}(1)$, i.e., we set $\mathsf{H}_{\mathsf{crs}}(1)$ to output a string in hiding mode. Now a reduction can extract ℓ valid digital signatures for the SIG scheme while at the same time only querying $\ell - 1$ time. Moreover, the adversary can easily identify the $(\mathsf{pk}^*, \mathsf{nonce}^*, \mathsf{psig}^*)$ which is valid $\mathsf{SIG.Verify}(\mathsf{pk}, (\mathsf{pk}^*, \mathsf{nonce}^*), \mathsf{psig}^*) = 1$ while at the same time was not queried to the signing oracle of the signature scheme SIG. It is possible thanks to the uniqueness property of the VRF and the fact that the adversary must output distinct messages. Thus, $(\mathsf{pk}^*, \mathsf{nonce}^*, \mathsf{psig}^*)$ is a valid forgery for the SIG

unforgeability experiment. The complete proof can be found in the full version of the paper.

Theorem 8 (Recipient Blindness). *Scheme 3 is recipient blind in the random oracle model assuming* DMWI *is mode indistinguishable and perfect witness indistinguishable in the hiding mode, and the* VRF *is indistinguishable.*

Proof (Sketch). The proof works as follows. The reduction programs the random oracle $\mathsf{H}_{\mathsf{crs}}$ in a way that it can simulate the DMWI proof using the trapdoor witness $(\cdot, \cdot, \cdot, \cdot, r)$, where $\mathsf{H}_{\mathsf{crs}}(1) = \mathsf{crs}$ and $(\mathsf{crs}, \cdot) \leftarrow \mathsf{Setup}(\lambda, \mathsf{binding}; r)$. Now the reduction does not need the VRF to generate signatures. Thus, we can replace the messages $\mathsf{m}_0, \mathsf{m}_1$ with random values. Since we do not query for proofs of correct evaluation, this follows from the indistinguishability of the VRF. The complete proof can be found in the full version of the paper.

Theorem 9 (Nonce Blindness). *Scheme 3 is nonce blind in the random oracle model assuming* DMWI *is mode indistinguishable and perfect witness indistinguishable in the hiding mode, and the* VRF *is indistinguishable.*

Proof (Sketch). The proof sketch follows the same strategy as above. The complete proof can be found in the full version of the paper.

7 Conclusions

In this paper, we looked at blind signatures from a practical perspective. We noticed that in many use cases, the distribution of the signed message does not have to be chosen by the recipient. In other words, the application will work even if the message is random but eventually known. By formalizing this idea, we introduced the notion of non-interactive blind signatures for random messages. The key property is that no online interaction between the signer and the recipient is required. It allows us to use blind signature in new applications, including distributing e-coins similarly to cryptocurrency airdropping. We also showed two constructions. One is efficient and admits preexisting public keys from other schemes. The other scheme generically captures the concept of NIBS and is constructed from well-known primitives. We also show how to date non-interactive signatures by introducing the notion of tagged NIBS. We also proposed open problems.

References

1. Abazi, V.: The European union whistleblower directive: a 'Game Changer' for whistleblowing protection? Ind. Law J. **49**(4), 640–656 (2020). https://doi.org/10.1093/indlaw/dwaa023
2. Abe, M., Fujisaki, E.: How to date blind signatures. In: Kim, K., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 244–251. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0034851

3. Agrawal, S., Kirshanova, E., Stehlé, D., Yadav, A.: Practical, round-optimal lattice-based blind signatures. In: Yin, H., Stavrou, A., Cremers, C., Shi, E. (eds.) ACM CCS 2022, pp. 39–53. ACM Press (2022). https://doi.org/10.1145/3548606.3560650
4. Au, M.H., Susilo, W., Mu, Y.: Practical compact E-cash. In: Pieprzyk, J., Ghodosi, H., Dawson, E. (eds.) ACISP 2007. LNCS, vol. 4586, pp. 431–445. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73458-1_31
5. Backes, M., Hanzlik, L., Schneider-Bensch, J.: Membership privacy for fully dynamic group signatures. In: Cavallaro, L., Kinder, J., Wang, X., Katz, J. (eds.) ACM CCS 2019, pp. 2181–2198. ACM Press (2019). https://doi.org/10.1145/3319535.3354257
6. Baldimtsi, F., Lysyanskaya, A.: Anonymous credentials light. In: Sadeghi, A.R., Gligor, V.D., Yung, M. (eds.) ACM CCS 2013, pp. 1087–1098. ACM Press (2013). https://doi.org/10.1145/2508859.2516687
7. Bao, F., Deng, R.H., Zhu, H.F.: Variations of Diffie-Hellman problem. In: Qing, S., Gollmann, D., Zhou, J. (eds.) ICICS 2003. LNCS, vol. 2836, pp. 301–312. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-39927-8_28
8. Barreto, P.S.L.M., Lynn, B., Scott, M.: Constructing elliptic curves with prescribed embedding degrees. In: Cimato, S., Persiano, G., Galdi, C. (eds.) SCN 2002. LNCS, vol. 2576, pp. 257–267. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36413-7_19
9. Barreto, P.S.L.M., Naehrig, M.: Pairing-friendly elliptic curves of prime order. In: Preneel, B., Tavares, S. (eds.) SAC 2005. LNCS, vol. 3897, pp. 319–331. Springer, Heidelberg (2006). https://doi.org/10.1007/11693383_22
10. Boldyreva, A.: Threshold signatures, multisignatures and blind signatures based on the Gap-Diffie-Hellman-Group signature scheme. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 31–46. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36288-6_3
11. Bowe, S.: BLS12-381: New zk-SNARK elliptic curve construction (2017). https://electriccoin.co/blog/new-snark-curve/
12. Camenisch, J., Hohenberger, S., Lysyanskaya, A.: Compact E-cash. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 302–321. Springer, Heidelberg (2005). https://doi.org/10.1007/11426639_18
13. Camenisch, J., Neven, G., shelat: Simulatable adaptive oblivious transfer. In: Naor, M. (ed.) EUROCRYPT 2007. LNCS, vol. 4515, pp. 573–590. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72540-4_33
14. Canard, S., Gaud, M., Traoré, J.: Defeating malicious servers in a blind signatures based voting system. In: Di Crescenzo, G., Rubin, A. (eds.) FC 2006. LNCS, vol. 4107, pp. 148–153. Springer, Heidelberg (2006). https://doi.org/10.1007/11889663_11
15. Chairattana-Apirom, R., Hanzlik, L., Loss, J., Lysyanskaya, A., Wagner, B.: PI-Cut-Choo and friends: compact blind signatures via parallel instance cut-and-choose and more. In: Dodis, Y., Shrimpton, T. (eds.) CRYPTO 2022, Part III. LNCS, vol. 13509, pp. 3–31. Springer, Heidelberg (2022). https://doi.org/10.1007/978-3-031-15982-4_1
16. Chaum, D.: Blind signatures for untraceable payments. In: Chaum, D., Rivest, R.L., Sherman, A.T. (eds.) Advances in Cryptology, pp. 199–203. Springer, Boston (1983). https://doi.org/10.1007/978-1-4757-0602-4_18
17. Davidson, A., Goldberg, I., Sullivan, N., Tankersley, G., Valsorda, F.: Privacy pass: bypassing internet challenges anonymously. PoPETs **2018**(3), 164–180 (2018). https://doi.org/10.1515/popets-2018-0026

18. Fischlin, M.: Round-optimal composable blind signatures in the common reference string model. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 60–77. Springer, Heidelberg (2006). https://doi.org/10.1007/11818175_4
19. Fischlin, M., Schröder, D.: Security of blind signatures under aborts. In: Jarecki, S., Tsudik, G. (eds.) PKC 2009. LNCS, vol. 5443, pp. 297–316. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00468-1_17
20. Fischlin, M., Schröder, D.: On the impossibility of three-move blind signature schemes. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 197–215. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_10
21. Frankel, Y., Tsiounis, Y., Yung, M.: Fair off-line E-cash made easy. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 257–270. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-49649-1_21
22. Fuchsbauer, G., Hanser, C., Kamath, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model from weaker assumptions. In: Zikas, V., De Prisco, R. (eds.) SCN 2016. LNCS, vol. 9841, pp. 391–408. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44618-9_21
23. Fuchsbauer, G., Hanser, C., Slamanig, D.: Practical round-optimal blind signatures in the standard model. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9216, pp. 233–253. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48000-7_12
24. Fuchsbauer, G., Hanser, C., Slamanig, D.: Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. J. Cryptol. **32**(2), 498–546 (2018). https://doi.org/10.1007/s00145-018-9281-4
25. Garg, S., Gupta, D.: Efficient round optimal blind signatures. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 477–495. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-55220-5_27
26. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24
27. Hanser, C., Slamanig, D.: Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In: Sarkar, P., Iwata, T. (eds.) ASIACRYPT 2014. LNCS, vol. 8873, pp. 491–511. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45611-8_26
28. Hanzlik, L., Slamanig, D.: With a little help from my friends: constructing practical anonymous credentials. In: Vigna, G., Shi, E. (eds.) ACM CCS 2021, pp. 2004–2023. ACM Press (2021). https://doi.org/10.1145/3460120.3484582
29. Hauck, E., Kiltz, E., Loss, J.: A modular treatment of blind signatures from identification schemes. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019. LNCS, vol. 11478, p. 375. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17659-4_12
30. Hauck, E., Kiltz, E., Loss, J., Nguyen, N.K.: Lattice-based blind signatures, revisited. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020. LNCS, vol. 12171, pp. 500–529. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-56880-1_18
31. Heilman, E., Alshenibr, L., Baldimtsi, F., Scafuro, A., Goldberg, S.: TumbleBit: an untrusted bitcoin-compatible anonymous payment hub. In: NDSS 2017. The Internet Society (2017)
32. Hendrickson, S., Iyengar, J., Pauly, T., Valdez, S., Wood, C.A.: Rate-limited token issuance protocol. Internet-Draft draft-privacypass-rate-limit-tokens-03, IETF Secretariat (2022)

33. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 150–164. Springer, Heidelberg (1997). https://doi.org/10.1007/BFb0052233
34. Katsumata, S., Nishimaki, R., Yamada, S., Yamakawa, T.: Round-optimal blind signatures in the plain model from classical and quantum standard assumptions. In: Canteaut, A., Standaert, F.-X. (eds.) EUROCRYPT 2021. LNCS, vol. 12696, pp. 404–434. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77870-5_15
35. Lysyanskaya, A.: Security analysis of RSA-BSSA. Cryptology ePrint Archive, Report 2022/895 (2022). https://eprint.iacr.org/2022/895
36. Micali, S., Rabin, M.O., Vadhan, S.P.: Verifiable random functions. In: 40th FOCS, pp. 120–130. IEEE Computer Society Press (1999). https://doi.org/10.1109/SFFCS.1999.814584
37. Miyazaki, S., Sakurai, K.: A more efficient untraceable e-cash system with partially blind signatures based on the discrete logarithm problem. In: Hirchfeld, R. (ed.) FC 1998. LNCS, vol. 1465, pp. 296–308. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055490
38. Pass, R.: Limits of provable security from standard assumptions. In: Fortnow, L., Vadhan, S.P. (eds.) 43rd ACM STOC, pp. 109–118. ACM Press (2011). https://doi.org/10.1145/1993636.1993652
39. Pfitzmann, B.P., Sadeghi, A.-R.: Anonymous fingerprinting with direct non-repudiation. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 401–414. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44448-3_31
40. Pointcheval, D., Stern, J.: Provably secure blind signature schemes. In: Kim, K., Matsumoto, T. (eds.) ASIACRYPT 1996. LNCS, vol. 1163, pp. 252–265. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0034852
41. Pointcheval, D., Stern, J.: Security arguments for digital signatures and blind signatures. J. Cryptol. **13**(3), 361–396 (2000). https://doi.org/10.1007/s001450010003
42. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45682-1_32
43. Wahby, R.S., Boneh, D., Jeffrey, C., Poon, J.: An airdrop that preserves recipient privacy. In: Bonneau, J., Heninger, N. (eds.) FC 2020. LNCS, vol. 12059, pp. 444–463. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-51280-4_24