Non-interactive Blind Signatures: Post-quantum and Stronger Security

Foteini Baldimtsi
George Mason University*

Jiaqi Cheng
UW–Madison‡

Rishab Goyal
UW–Madison†

Aayush Yadav
George Mason University§

Abstract

Blind signatures enable a receiver to obtain signatures on messages of its choice without revealing any message to the signer. Round-optimal blind signatures are designed as a two-round interactive protocol between a signer and receiver. Coincidentally, the choice of message is not important in many applications, and is routinely set as a random (unstructured) message by a receiver.

With the goal of designing more efficient blind signatures for such applications, Hanzlik (Eurocrypt '23) introduced a new variant called non-interactive blind signatures (NIBS). These allow a signer to asynchronously generate *partial* signatures for any recipient such that only the intended recipient can extract a *blinded* signature for a *random* message. This bypasses the two-round barrier for traditional blind signatures, yet enables many known applications. Hanzlik provided new practical designs for NIBS from bilinear pairings.

In this work, we propose new enhanced security properties for NIBS as well as provide multiple constructions with varying levels of security and concrete efficiency. We propose a new generic paradigm for NIBS from circuit-private leveled homomorphic encryption achieving optimal-sized signatures (i.e., same as any non-blind signature) at the cost of large public keys. We also investigate concretely efficient NIBS with post-quantum security, satisfying weaker level of privacy as proposed by Hanzlik.

*Email: foteini@gmu.edu. This research is supported by NSF #2143287.

†Email: rishab@cs.wisc.edu. Support for this research was provided by OVCRGE at UW–Madison with funding from the Wisconsin Alumni Research Foundation.

‡Email: jcheng77@wisc.edu.

§Email: ayadav5@gmu.edu. Work done while visiting UW–Madison. Funded by NSF #2143287.

Contents

1 Introduction **4**

2 Technical Overview **7**
- 2.1 Defining non-interactive blind signatures . . . 7
- 2.2 Extending Fischlin's paradigm to NIBS . . . 9
- 2.3 A new template: Circuit-private LHE to NIBS . . . 10
- 2.4 Making Fischlin-based NIBS practical and post-quantum . . . 11
- 2.5 Security and the randomized OM-ISIS assumption . . . 13
- 2.6 Efficiency comparisons for our NIBS schemes . . . 17

3 Preliminaries **17**
- 3.1 Lattice preliminaries . . . 18
- 3.2 Hardness assumptions . . . 19
- 3.3 Randomness extraction . . . 20
- 3.4 Cryptographic building blocks . . . 20

4 A Stronger Model for Non-Interactive Blind Signatures **24**

5 NIBS from Circuit Private LHE **27**
- 5.1 Knowledge of secret keys model . . . 28
- 5.2 Construction . . . 28
- 5.3 One-more unforgeability . . . 30
- 5.4 Strong receiver blindness . . . 33
- 5.5 Strong nonce blindness . . . 37

6 The Randomized One-more ISIS Assumption **39**
- 6.1 Cryptanalysis of the rOM-ISIS assumption . . . 41

7 Lattice-based NIBS **44**
- 7.1 Construction . . . 45
- 7.2 One-more unforgeability . . . 46
- 7.3 Receiver blindness . . . 49
- 7.4 Efficiency of lattice-based NIBS . . . 52

8 Lattice-based NIBS with Nonce Blindness **54**
- 8.1 Construction . . . 55
- 8.2 Nonce blindness . . . 56
- 8.3 Receiver blindness . . . 60
- 8.4 One-more unforgeability . . . 62

A Equivalence of Two NIBS Formalizations **71**

B Knowledge of Secret Key Assumption **73**

C Lattice-based Tagged NIBS **74**
C.1 Tagged NIBS definitions 74
C.2 Construction 75
C.3 One-more unforgeability 77
C.4 Receiver blindness 79

D NIBS from General Purpose NIZKs **81**
D.1 Construction 81
D.2 One-more unforgeability 83
D.3 Strong receiver blindness 84
D.4 Strong nonce blindness 88

1 Introduction

A blind signature scheme is a special class of digital signatures that allows a receiver to obtain a signature on a message of the receiver's choice without revealing this message to the signer. Typically, blind signatures are implemented as an interactive protocol between the *signer* (holding a secret key sk) and the *receiver* (holding the message m to be signed and the signer's public verification key vk). At the end of their interaction, the receiver should obtain a valid signature σ on m. The protocol needs to be correct (i.e., the output signature can be verified using vk and m) and additionally, it needs to satisfy two security properties: *blindness* (i.e., the signer does not obtain any information with respect to m) and *one-more unforgeability* (i.e., the receiver cannot create more valid signatures beyond the ones received after interacting with the signer).

Blind signatures were first introduced by Chaum [Cha83] and have been used as a main building block in numerous privacy-preserving applications. One of the most prominent applications, and the one originally considered by Chaum [Cha83] is private electronic payments (e-cash). The idea is that in an e-cash system, a bank (a.k.a. the signer) creates electronic coins through the use of blind signatures. Each coin is associated with a unique serial number selected by the user, and the bank's signature on it serves as proof of its validity. The blindness property guarantees that nobody (including the bank) can trace the spending of user coins, while the unforgeability property guarantees that users cannot forge coins. Beyond e-cash, blind signatures have found applications in e-voting systems [CGT06], anonymous credentials/tokens [PZ13, BL13, FHS15, DGS+18], direct anonymous attestation [BCC04] and coin tumblers for cryptocurrencies [HBG16, HAB+17].

Two-move barrier. The original blind signature scheme by Chaum [Cha83] requires *two-moves* of interaction: the recipient sends the blinded message to the signer and the signer responds with a signature on the blinded message. The recipient locally un-blinds the signature and outputs the final message/signature pair. It is easy to see that blind signatures are impossible unless the recipient and signer both send at least one message. Thus, *two-move* blind signatures are considered *round-optimal* and are particularly interesting for real-world applications. Beyond their optimal communication efficiency, their unforgeability automatically extends to the concurrent setting [Lin08, HKKL07] (i.e. even when the adversary engages in multiple concurrent sessions with the signer). There has been a long line of research on round-optimal blind signatures under different models and assumptions [BNPS02, Bol03, Fis06, GRS+11, SC12, FHS15, FHKS16, Gha17, KNYY21, AKSY22, BLNS23a].

Non-interactive blind signatures: beyond the two-move barrier. In a two-move blind signature scheme, the first message originates from the receiver and typically includes a blinded version of the message to be signed. Interestingly though, in many applications, the signed message is selected randomly and does not come from a specific distribution or has a specific structure. Consider e-cash as an example, where the signed message is a random value denoting the e-coin ID, arbitrarily selected by a user. This observation was recently made explicit by Hanzlik [Han23] who put forward a new notion called non-interactive blind signatures (NIBS).

In a NIBS system, each receiver is associated with a public-secret key pair $(\mathsf{pk}_R, \mathsf{sk}_R)$ such that any signer can asynchronously create partial signatures psig, called *presignatures*, given only the receiver's public key pk_R. The receiver can extract a pair of *blinded* signature σ and message μ from presignature psig using its corresponding secret key sk_R. Completeness states that σ is a

valid signature for message μ, and can be verified given only signer's verification key vk. The message μ should be an unpredictable message for the receiver and the signer.

In terms of security, a NIBS scheme should satisfy the properties of (one-more) unforgeability and blindness similar to the interactive setting. As pointed out in [Han23], using some secret input from the receiver (i.e. sk_R) during the computation of the final signature is crucial to achieve blindness without incurring two-moves. Additionally, Hanzlik also extended the idea of NIBS to *tagged* non-interactive blind signatures (TNIBS) to resemble the functionality of *partially blind* signatures. That is, signatures that allow for the inclusion of some un-blinded metadata, called the 'tag', along with the blinded signature. For example, the tag could contain application specific public information such as date, time, purpose, etc.

Non-interactive blind signatures could replace traditional blind signatures in any application where the choice of message is not important, and could be set as a random (unstructured) message. This would lead to protocols with minimal communication complexity. Beyond the case of e-cash, (tagged) NIBS can be used to implement anonymous token systems, à la Privacy Pass [DGS+18], and lottery systems—we refer to [Han23] for a longer discussion on applications. Interestingly, if we consider the PKI model, then a recipient never needs to interact with a signer to send its public key. Thus, a signer can issue and publish presignatures for users without ever interacting with them. This property makes NIBS suitable for many modern applications. As an example, consider cryptocurrency airdrops which is a mechanism to gift coins to users (typically used to bootstrap interest in a new coin). A cryptocurrency could distribute coins to registered user public keys by creating and publishing presignatures, and then users could obtain the final signatures (i.e., the actual coins) in a privacy preserving way.

Existing constructions. [Han23] provided a generic template for building (T)NIBS in the random oracle (RO) model from verifiable random functions, digital signatures, and general purpose dual-mode non-interactive witness indistinguishable proofs. They also designed a practically efficient NIBS scheme from signatures on equivalence classes [HS14, FHS19], and a TNIBS from tag-based equivalence class signatures [HS21]. Both schemes crucially rely on the use of bilinear pairings for instantiating the underlying equivalence-class signature. Moreover, their security proofs are carried out in the generic group model.

This work. The goal of this work is two-fold: pushing the barriers both on the definitional framework and on constructions. Since NIBS is a newly introduced primitive, our first focus is to examine the definitional framework proposed in [Han23] and to ask the following natural question:

> *Is the definitional framework proposed by [Han23] correctly capturing the desired properties?*

As it turns out, the blindness properties defined by [Han23] only capture a weaker level of privacy, which is not adequate for all the applications NIBS was proposed for. Thus, in this work, we revisit the definitional framework and provide blindness definitions that better capture the intended level of privacy and which should be considered the basis for future NIBS development.

On the construction side, the current state-of-the-art is that we do not have efficient post-quantum NIBS, *even with only conjectured security*. This was left as an important open problem in [Han23] and brings us to the following question:

Can we design efficient (tagged) NIBS from post-quantum assumptions?

We answer the above question in the affirmative, and believe this will lead to further progress in the emerging area of practical (round-optimal) blind signature schemes with post-quantum security (see [LNP22a, AKSY22] and references therein). We summarize our contributions below.

1. We address the privacy shortcomings of the existing NIBS definition framework, and present new stronger blindness properties for NIBS. Our study of stronger blindness definitions is motivated by scenarios in which a blindness attacker might be able to get a hold of some presignatures along with their corresponding message-signature pairs.[1] We also provide a new feasibility result by extending the folklore Fischlin's paradigm [Fis06] to the non-interactive setting, while proving security in our stronger corruption model.

2. We propose a new generic paradigm for designing (non-interactive) blind signatures with *optimal-sized signatures* from any circuit-private leveled homomorphic encryption (LHE) scheme. We show that LHE can upgrade any regular (non-blind) signature into a NIBS without increasing the size of the final signature. We believe that our LHE-based template could serve as an alternative to the famous Fischlin's paradigm. With great ongoing research in the realm of (somewhat/leveled) homomorphic encryption, we further believe that our proposed template might lead to alternate approaches to efficient (non-interactive) blind signatures in the future. As we discuss later, our LHE-based construction enables a new interesting tradeoff leading to optimal-sized signature with large public keys and pre-signatures; while a general NIZK-based solution enables the opposite tradeoff.

3. We also design a practical lattice-based (T)NIBS scheme that satisfies a weaker level of privacy, as proposed in [Han23]. As we discuss later, this is still adequate for a subset of applications mentioned in [Han23]. The concrete costs and overhead of our NIBS scheme are similar to that for state-of-the-art interactive (round-optimal) blind signatures from lattices. We introduce, and prove security of our system under, a more robust variant of the one-more-ISIS assumption [AKSY22], that we call *randomized one-more-ISIS* assumption which may be of independent interest. We do preliminary cryptanalysis of our assumption, and show that all known attacks on the one-more-ISIS assumption [AKSY22] are unsuccessful in breaking our variant.

Related work on lattice-based blind signatures. Recently, many new round-optimal schemes for lattice-based blind signatures have been proposed [LNP22a, dK22, AKSY22, BLNS23a]. Lyubashevsky et al. [LNP22a] use one-time signatures, and build blind signatures under standard lattice assumptions (MSIS, MLWE). However, their scheme only supports bounded polynomial number of signatures per public key with each signature being around 150 KB in size. This was improved by del Pino and Katsumata [dK22] who adapted Fischlin's paradigm [Fis06] to the lattice setting. The resulting scheme allows for an unbounded number of signatures, of around 102 KB each, and is also secure under standard lattice assumptions. Agrawal et al. [AKSY22] significantly improved both the signature, as well as the transcript size, by also leveraging Fischlin's paradigm, but relying on efficient lattice-based NIZK for linear relations [LNP22b]. Their final signature size is 45 KB and the total transcript is just over 1 KB. The security of their scheme is based on the one-more ISIS assumption [AKSY22]. Most recently, Beullens et al. [BLNS23a] improved the

[1] At a very high level, one could view our definitions providing a stronger CCA-style blindness guarantee, while existing definitions provide only a CPA-style blindness guarantee.

Scheme	Assumption	Communication Complexity		
		$R \to S$	$S \to R$	$\lvert\sigma\rvert$
[AKSY22]	OM-ISIS	0.96 KB	0.56 KB	45 KB
[BLNS23a]	MSIS and MLWE	> 100 KB	~ 60 KB	22 KB
Our Construction (7.1)	rOM-ISIS	0 B	0.96 KB	68 KB

Table 1: Our construction compared with state of the art two-move lattice-based blind signatures. $R \to S$ communication is 0 bytes if a PKI exists, else it is a one-time cost unlike the two-move schemes that have a linear dependence.

Agrawal et al. scheme by off-loading some inefficient computation to the receiver's first message. This has the twin benefit of simultaneously allowing a reduction to standard lattice assumption, as well as reducing the signature size to just 22 KB. However, this came at the cost of the receiver having to prove the validity of cryptographic hash function input-output pair in its first message. This makes the first message significantly less efficient at a few hundred kilobytes. We compare the concrete cost of our lattice-based NIBS construction with the current state-of-the-art round-optimal blind signatures from lattices in Table 1.

2 Technical Overview

In this section, we provide a high-level technical overview and summarize our main contributions. We start by recalling the notion of non-interactive blind signatures.

2.1 Defining non-interactive blind signatures

In a NIBS system, the Setup algorithm generates the system's (global) public parameters pp (viewed as CRS in-the-sky). There are two key generation algorithms, $\mathsf{KeyGen}_S \to (\mathsf{sk}, \mathsf{vk}), \mathsf{KeyGen}_R \to (\mathsf{sk}_R, \mathsf{pk}_R)$, for the signer and receiver, respectively. Given sk, the signer runs a (randomized) Issue algorithm for any receiver's public key pk_R to compute a presignature, psig and nonce. Here nonce roughly denotes the randomness used by the signer. The receiver then runs the Obtain algorithm on the above presignature to compute the final message-signature pair (μ, σ) using secret key sk_R[2].

[Han23] also extended the above basic notion to *tagged* NIBS (TNIBS). The goal was to capture the notion of partially blind signatures [AF96] which allow a signer and recipient to jointly agree on a public metadata/value to be included as part of the signed message. This is captured by allowing such a common metadata/value, called a *tag*, to be chosen explicitly by the signer, and shared with the receiver along with the presignature. Syntactically, TNIBS is defined identically to NIBS, except Issue, Obtain, and Verify algorithms take tag τ as an additional input.

The need for reusability. An essential property of NIBS is *reusability*, which says that a signer can issue multiple distinct presignatures (leading to multiple distinct message-signature pairs) for the *same* receiver public key pk_R. Hanzlik [Han23] attempted to capture this property by providing a random nonce value as input to the Issue algorithm. Unfortunately, as we explain in Section 4, the existing formulation is insufficient in capturing the desired reusability property. The issue is that, under the current formulation, there can exist trivial NIBS scheme where Issue and Obtain algorithms simply ignore nonce, thus lead to a limited one-use system. To fix this, we

[2]Our syntax is nearly identical to that proposed in [Han23]. We deviate slightly in the handling of nonce as we discuss in this section, and also later in Section 4.

formalize reusability as its own property. Informally, it says that any receiver should obtain two distinct message-signature pairs for two distinct presignature-nonce pairs for any given receiver. Fortunately, the bilinear-based NIBS schemes [Han23] already satisfy reusability, but just did not prove/define it formally.

NIBS Security. For security, we want NIBS to satisfy one-more unforgeability and blindness. One-more unforgeability for NIBS can be defined as a natural extension of the one-more unforgeability for traditional blind signatures [PS96]: the adversary gets access to a presignature oracle, and after receiving ℓ presignatures for recipient public keys specified by the adversary, the adversary must return $\ell + 1$ valid signatures.

On the other hand, defining blindness for NIBS is much more nuanced than for traditional blind signatures[3]. Intuitively, it is captured by defining unlinkability between presignatures and final signatures. More formally, Hanzlik [Han23] proposes the following experiment: the adversary receives two recipient public keys $\mathsf{pk}_{R_0}, \mathsf{pk}_{R_1}$, and outputs two presignatures $\mathsf{psig}_0, \mathsf{psig}_1$ (one for each). The challenger extracts the final signature-message pairs $(\mu_0, \sigma_0), (\mu_1, \sigma_1)$ from these, and the scheme is said to satisfy *receiver blindness* if the attacker cannot link the final signature-message pairs to the presignatures. Abstractly, receiver blindness could be thought of as an 'inter-receiver' blindness property, since it guarantees that a malicious signer cannot figure out the recipient of a final (blinded) signature between two possible options.

Hanzlik [Han23] also proposed a secondary blindness notion, called *nonce blindness*. This can be viewed as an *intra-recipient* blindness property, where what we want is that a signer issuing more than one presignature to a *specific* user should not be able to link the final message-signature pairs to the corresponding presignatures. Essentially, this provides some flavor of "ordering" unlinkability. This was formalized by Hanzlik [Han23] via an experiment similar to that of receiver blindness with the difference that the adversary is only given one recipient public key pk_R, while it still outputs two presignatures $\mathsf{psig}_0, \mathsf{psig}_1$. The challenger extracts the final signature-message pairs $(\mu_0, \sigma_0), (\mu_1, \sigma_1)$ from these, and the scheme is said to satisfy *nonce blindness* if the attacker cannot link the final signature-message pairs to the original presignatures.

The issues with [Han23] blindness. Unfortunately, the two blindness definitions given by Hanzlik [Han23] do not really capture a sufficient level of privacy that would be required in most NIBS applications.

The core issue is that both definitions only guarantee unlinkability of two presignatures with their corresponding message-signature pairs. That is, a NIBS scheme secure under the existing nonce blindness might not satisfy unlinkability when given three (or more) presignatures to different users. To showcase the problem consider a concrete scenario wherein an adversary issues two presignatures to a user with pk_{R_0} and one presignature to a user with pk_{R_1}. Once the adversary learns two signatures of a user (through a verifier), they will know with certainty[4] that this must have been the user with pk_{R_0}.

New stronger blindness definitions. To capture such practical attacks described above, we propose a new stronger security framework for NIBS blindness properties. The motivation is to cap-

[3] Essentially, this is because the receiver does not choose the message in the non-interactive setting.

[4] We note that this is not a problem in regular blind signatures, where blindness is indeed defined as an indistinguishability game. However the creation of user assigned presignatures in NIBS complicates the definition of blindness.

ture scenarios where an adversary is able to learn some correlation between presignature-nonce pairs and their corresponding blind signature-message pairs. We formulate this by providing an adversary oracle access to the receiver's secret key in the form of Obtain queries. To ensure this is not trivially impossible, we restrict the adversary to not make an Obtain query on any of the two challenge presignatures $\mathsf{psig}_0, \mathsf{psig}_1$.

We call these the strong receiver (resp. nonce) blindness properties. At a high level, these can be viewed as CCA-version of blindness (since adversary gets query access to the secret key like in IND-CCA). These definitions guarantee blindness to hold even when a malicious signer is able to bypass blindness of signatures from previous sessions. Now any NIBS scheme that is secure under the above stronger definitions protects against the attacks described earlier. This is because we are letting an adversary break blindness of all non-challenge presignatures, and still ask that the blindness of the challenge presignatures is unaffected. We discuss our definitions in detail later in Section 4.

When is basic-blindness [Han23] sufficient? There are many applications, as discussed in the introduction as well as [Han23], where the basic-blindness receiver and nonce blindness properties as defined by [Han23] are not sufficient. Thus, one of our main assertions and contributions is that for typical applications of NIBS, where multiple presignatures are issued to multiple recipients, the strong blindness definitions should be used.

We also remark that there are some applications where the notion of basic blindness will be sufficient. As a concrete application, consider one-per user anonymous airdrops or e-voting tokens, where only a single presignature will be generated for each recipient. In such applications the notion of (basic) receiver-blindness alone, as defined by [Han23], is sufficient. More broadly, any NIBS scheme satisfying only basic receiver blindness implies a round-optimal (interactive) blind signature scheme for random messages. Thus, any NIBS scheme with basic blindness already covers all known applications of traditional blind signatures where messages are selected randomly. Therefore, we believe that a NIBS scheme satisfying only receiver-blindness is interesting for some applications. Since achieving basic-blindness is relatively easier than achieving strong-blindness, a NIBS scheme secure as per the basic blindness can lead to schemes with better concrete efficiency.

In this work, we provide new constructions and templates for designing NIBS that are secure as per our stronger blindness definitions. We also design new practically efficient NIBS scheme secure as per the basic blindness definitions [Han23]. This leads to first post-quantum NIBS scheme. We leave the problem of designing practically efficient NIBS with strong blindness as an interesting open problem.

2.2 Extending Fischlin's paradigm to NIBS

In this section, we start describing our main technical ideas. Our starting point is the well-known Fischlin's paradigm [Fis06] for constructing *traditional* blind signatures. At a high level, Fischlin's scheme assumes a common reference string (CRS), a standard signature scheme S, an encryption scheme $\mathcal{E}$, a commitment scheme COM and general NIZK proofs.

The CRS contains an encryption public key pk_E, while each signer simply generates its key pair $(\mathsf{sk}, \mathsf{vk})$ by running the setup of the signature scheme. The user computes $c = \mathsf{COM}(m)$ and sends c to the signer. The signer runs $\mathsf{Sign}(\mathsf{sk}, c)$ to produce σ' and sends σ' to the user. If the signature

verifies, the user computes $\mathsf{ct} = \mathsf{Enc}_{\mathsf{pk}_E}(c \,\|\, \sigma')$ and outputs ct, m and a NIZK π proving knowledge of signature σ' and message m such that σ' is a signature on a commitment of m. Intuitively, the one-more unforgeability follows from binding of the commitment scheme and unforgeability of the signature scheme, via straight-line extraction enabled by the trapdoor key sk_E corresponding to pk_E. Further, blindness follows from the semantic security of the encryption scheme, zero-knowledge of the NIZK, and hiding property of the commitment. Consider the following natural extension of Fischlin's paradigm to the non-interactive setting.

Adapting Fischlin's paradigm to NIBS. Given that there is no interaction between the signer and the user, the receiver's public key will, in a sense, replace the commitment submitted in the first move. The challenge now is that the commitment can no longer represent a commitment of a single message but has to be a succinct commitment of an exponential number of messages at once. The hope is that signer can sign the commitment in a way such that the signature obliviously binds to exactly one of those messages at random. This binding to one message from an exponential set is important for one-more unforgeability as otherwise the receiver could cheat. Lastly, the receiver can use NIZKs to prove knowledge of the signature and reveals the choice of message that was obliviously selected.

Thus, to extend the above template, we need to overcome two challenges: (a) how to commit to an exponential number of messages efficiently, and (b) how can the signer obliviously select one of those messages. Our approach is to set the receiver's public key pk_R as a commitment to a PRF key K, and the key K (along with its opening) as the secret key. Note that this implicitly defines the exponential set of messages as all the outputs of the PRF. Then, during the issuance of a presignature, the signer can obliviously sample one of the messages by selecting a random input r and signing it along with pk_R. The user obtains a final signature on the message $m = F_K(r)$ and along with m it outputs σ to be a NIZK proof of knowledge of a (pre)signature on pk_R and r corresponding to the message m.

Given that the PRF is a deterministic function and commitment is binding, our approach guarantees that the receiver can only obtain a single final signature from a presignature which allows us to prove one-more unforgeability. Further, blindness properties can be reduced to the zero-knowledge property of the underlying NIZK and hiding of the commitment scheme. Moreover, as we discuss in detail later in Appendix D, we can prove our scheme to be secure under our stronger blindness definitions. The core intuition is that the NIZK proof systems are multi-theorem zero-knowledge, thus they can be used to simulate responses to all Obtain queries including the challenge queries, thus each blinded signature is completely unlinkable to its presignature, since the blinded signature is simulated without any information about the presignature.

Remark 2.1. Hanzlik [Han23] proposed a similar generic template for NIBS using verifiable random functions and general-purpose dual mode witness indistinguishable proofs, but only proved security under his restricted blindness notions. Our construction is notably simpler as it only requires a PRF and general-purpose NIZK, and provides stronger blindness security. For completeness, we provide the full construction in Appendix D satisfying our strong blindness definitions.

2.3 A new template: Circuit-private LHE to NIBS

We now describe a new template for designing NIBS with stronger blindness based on circuit-private LHE. We already know that homomorphic encryption with special properties [AJL+12,

MW16] are useful in designing round-optimal MPC protocols. Moreover, such protocols can be client-optimized where the communication complexity of a client is extremely low. Our main observation is that we can instantiate a NIBS scheme as a specialized two-round MPC protocol where the first round message can be reused [IKO+11, AMPR14, MR17, BJOV18, CDI+19], and by using FHE to instantiate the protocol, we can optimize the communication complexity for the receiver to nearly optimal. Let us elaborate on our main ideas below.

Our key idea is that rather than using NIZKs to hide the receiver's secrets, we can leverage the fact that leveled homomorphic encryption (LHE) enables arbitrary homomorphic operations on the receiver's commitment. Specifically, the receiver commits to a PRF secret key K by encrypting it under LHE. Using this commitment, a signer issues a presignature as follows: it first homomorphically evaluates the PRF $F_K(\cdot)$ on some randomness r of its choice and then homomorphically generates a signature on $F_K(r)$ treated as a message. The resulting ciphertext $\widehat{\mathsf{ct}}$ is, therefore, an encryption of the signature on $F_K(r)$. Under the mild assumption that the LHE is circuit-private [OPP14], the signer can simply send $\widehat{\mathsf{ct}}$ as the presignature along with an argument of knowledge that $\widehat{\mathsf{ct}}$ was evaluated using the signing key as input to the circuit, and randomness r as the nonce. The receiver sets its message to $F_K(r)$ and obtains the corresponding signature by decrypting $\widehat{\mathsf{ct}}$. Notably, the final signature is just a regular signature and is thus optimal in size. Moreover, beyond optimality, this construction satisfies our stronger notion of blindness: strong receiver and nonce blindness.

At a high level, both strong-receiver/nonce blindness follow from the IND-CPA security of the encryption scheme which ensures that ct does not reveal anything about K, and the pseudorandomness of F, which ensures that two messages on different nonces look random. The one-more unforgeability of the protocol follows from the unforgeability of the underlying signature scheme and the circuit-privacy of the LHE. For the proof to go through, we additionally require that both parties also prove knowledge of their secret keys. That is, our proof is in the knowledge of secret key (KOSK) model [MOR01, Bol03]. As a supplementary contribution, we also prove that any NIBS scheme secure in the KOSK model is also secure in the standard model, assuming existence of NIZKs.

Our LHE-based NIBS construction is described in Section 5, and the generic compiler to upgrade security in KOSK model to standard model is provided in Appendix B. We highlight that our construction explores a new interesting trade-off that yields optimal-sized signatures while paying in terms of higher setup and computation costs. We believe this could lead to alternate approaches to practical NIBS (and round-optimal interactive blind signatures) in the future.

2.4 Making Fischlin-based NIBS practical and post-quantum

The next contribution of our work is a practically efficient NIBS scheme that satisfies basic-blindness definitions [Han23]. Moreover, we provide a proof-of-concept estimate of all the parameters sizes of our NIBS construction. Let us start by revisiting the Fischlin-based NIBS construction that we explained earlier in Section 2.2. We show how to adapt our construction by combining with new ideas such that it makes the design concretely efficient, yet it satisfies (basic) receiver-blindness.

With the above goal, we look back to the recent work by Agrawal et al. [AKSY22] on desgining practical rount-optimal (interactive) lattice-based blind signatures. Agrawal et al. suggested that the usage of NIZKs is somewhat unavoidable when designing round-optimal blind signatures

from lattices[5]. Thus, their main insight was that, by optimizing the NP language for which NIZKs are needed, one could instantiate Fischlin's paradigm efficiently. In particular, they showed that NIZKs for linear relations, which are already known to be efficiently implementable from lattice assumptions [LNP22b], are sufficient for building practically efficient round-optimal blind signatures.

Our initial attempt is to follow a similar approach. We start by optimizing our NIZK-based template for designing NIBS, similar to how [AKSY22] optimized Fischlin's paradigm. The challenge here is to remove all inefficient generic cryptographic components currently used in our NIZK relation, and implement them via just linear relations. As a starting point, consider the standard hash-then-sign paradigm. A signer samples lattice trapdoor as $(\mathbf{C}, \mathbf{T_C})$ and outputs $\mathbf{C}$ as verification key vk. Receiver starts by randomly selecting a PRF key K, and sets its public key $\mathsf{pk}_R = \mathsf{Com}(K; s)$ as a commitment of K with its secret key $\mathsf{sk}_R = (K, s)$, for some random string s. Signer then assigns presignatures to receiver: it uniformly samples randomness r as nonce, and provides a short preimage of $\mathsf{H}(\mathsf{pk}_R \| r)$ as the presignature, where $\mathbf{C} \cdot \mathbf{z} = \mathsf{H}(\mathsf{pk}_R \| r)$ and $\mathsf{psig} = \mathbf{z}$, $\mathsf{nonce} = r$. Finally, receiver generates message μ as $F_K(r)$, and signature π as a NIZK proof establishing that $\mathbf{z} = \mathbf{C}^{-1}(\mathsf{H}(\mathsf{pk}_R \| r))$, $\mu = F_K(r)$ and $\mathbf{z}$ is short.

Such a design essentially instantiates our NIBS template with the lattice-based signature scheme in [GPV08]. Observe that the major source of inefficiency in the NIZK arises from the hash evaluation and the PRF evaluation. These evaluations are extremely heavy and thus not very practical to prove in a NIZK. Our strategy is to entirely exclude both computations from the NIZK relation. Following from the blueprint of [AKSY22], it is possible to remove the hash evaluation (from the NIZK relation) if we fix the receiver's public key as $\mathsf{pk}_R = \mathbf{A} \cdot \mathbf{x} + \mathsf{H}(\delta)$ for some randomly sampled $\mathbf{x}$ and hash input δ, where $\mathbf{A}$ is a random matrix part of the CRS. Evaluations of $\mathsf{H}(\delta)$ could then be performed outside of the NIZK proof system, if the signature contains δ in the clear. This removes the need to prove costly hash evaluations using the NIZK. The reason this does not break blindness is that even if the signer learns δ, it still cannot link δ with pk_R as $\mathbf{x}$ contains enough entropy so that $\mathbf{A} \cdot \mathbf{x}$ statistically hides all information about $H(\delta)$.

However, at this point, the parallels between our design and the interactive blind signature scheme in [AKSY22] start to diminish. This is because, in the interactive setting, a receiver can simply select δ as a fresh message in each new session. But, in NIBS, we need to create multiple signatures for the same receiver key pk_R. And, the issue is pk_R binds to a fixed value for the lifetime of the system. Thus, unlike [AKSY22], we cannot set δ as the final signed message. This is because this will violate reusability, and make the NIBS scheme only single-use. Therefore, it will not be any more advantageous than a regular two-round blind signature scheme. Basically, this suggests that we cannot simply replace the usage of a PRF within our generic template as easily! Moreover, if we keep on using the PRF to generate a fresh message from each distinct presignature computed by the signer, then the receiver will have to prove it using the NIZK which will be extremely inefficient.

In summary, this means we can no longer use δ (inside each receiver's key), or the PRF trick to generate a fresh message from each distinct presignature computed by the signer. To this end, we propose an entirely different approach to generate the final messages from presignatures. Our insight here is that the extracted final messages need *not* be 'truly random messages', but it is enough to have (1) the messages be just uncorrelated amongst different receivers, and (2) be

[5]This is due to the fact that all existing practical lattice-based signature schemes do not support simple algebraic homomorphisms in a practically efficient way, unlike signatures from pairings or factoring-based assumptions.

distinct for any two distinct valid presignatures for the same receiver.

Our idea is to set the final message as $\mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_\perp \\ \mathbf{z}_\perp \end{bmatrix}$ instead of $F_K(r)$, where $\mathbf{C} \cdot \mathbf{z} = \mathsf{pk}_R$. Here we write $\mathbf{x}_\top$ and $\mathbf{x}_\perp$ to be the top and bottom halves of $\mathbf{x}$ (respectively), and the same notation applies to $\mathbf{z}$. The intuition is that, while the receiver's public key contains the entire $\mathbf{x}$ vector, if we set the parameters appropriately, then we can argue by the leftover hash lemma that $\mathbf{x}_\perp$ is statistically hidden from the signer's view. We point out that it is important in our design that the signer implicitly assigns nonce to be $\mathbf{z}_\perp$ (with presignature $\mathbf{z}_\top$) where $\mathbf{C} \cdot \mathbf{z} = \mathsf{pk}_R$, instead of sampling r as randomness and $\mathbf{z}$ as preimage of $\mathsf{pk}_R \| r$. This ensures that NIZKs for linear relations are still sufficient as well as makes the design even more optimal. For ensuring this property from preimage sampling, we rely on the Bonsai trick [CHKP10, ABB10b] for lattice trapdoors. Using the standard Bonsai trick, one can generate a preimage $\mathbf{z}$ with the above special property, i.e. $\mathbf{z}_\perp$ is sampled as a random Gaussian vector with appropriate norm.

The above serves as the core skeleton of our efficient construction in Section 7 however, as we discuss next, we need to make a few more slight modifications to ensure that we can prove desired unforgeability and blindness security properties for our NIBS scheme.

2.5 Security and the randomized OM-ISIS assumption

Since the lattice-based core of our optimized NIBS scheme shares many similarities with interactive blind signature scheme of [AKSY22], a natural first attempt is to prove unforgeability of our NIBS scheme using the same proof strategy as used in their work. Now, to prove security of their round-optimal blind signature scheme, Agrawal et al. [AKSY22] proposed a new ISIS-like assumption, called the one-more ISIS assumption (OM-ISIS)[6]. Below we briefly recall the assumption[7].

1. The challenger samples a challenge matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ along with a large set of random target vectors $T \subset \mathbb{Z}_q^n$. It provides the attacker with $\mathbf{A}$ and T.

2. $\mathcal{A}$ can make preimage queries for any target vector $\widehat{\mathbf{t}} \in \mathbb{Z}_q^n$ to which the challenger replies with a short[8] vector $\widehat{\mathbf{x}}$ such that $\mathbf{A} \cdot \widehat{\mathbf{x}} = \widehat{\mathbf{t}}$.

3. OM-ISIS assumption says that $\mathcal{A}$, having made at most ℓ preimage queries, cannot output $\ell + 1$ distinct vector pairs $\left\{(\mathbf{x}_j, \mathbf{t}_j)\right\}_{j \in [\ell+1]}$ such that $\mathbf{A} \cdot \mathbf{x}_j = \mathbf{t}_j$, $\mathbf{t}_j \in T$, $\mathbf{x}_j$ is sufficiently short.

Intuitively, the assumption says that an adversary cannot find *good* (i.e., reasonably short) preimages for a set of $\ell + 1$ randomly selected vectors, even when it has access to a preimage sampling algorithm that can generate at most ℓ preimages for adversarially selected target vectors.

Agrawal et al. also performed some preliminary cryptanalysis of their assumption. While they were able to design practical attacks [AKSY22, § 4.5] for certain parameter settings, they suggested that, by carefully selecting the parameters in the above assumption, all known attacks fail. Moreover, they proved that their two-round blind signature scheme can be proven secure under the above assumption with those parameters.

[6] The "one-more" name inspired from one-more-RSA assumption [BNPS03].

[7] For ease of exposition, we present a simplified assumption here. In the original assumption, the set of target vectors T can be chosen adaptively by the challenger.

[8] As in typical lattice settings, by short vectors we mean vectors with small norms.

However, the presence of efficient practical attacks on a wide range of parameters on the OM-ISIS assumption suggests that this assumption is not very stable and robust. On a technical level, the assumption appears quite strong since an attacker can ask for short preimages for "*any*" target vector of its choice. In more detail, an attacker can simply ask for multiple short preimages for the all-zeros vector. Given such preimage vectors, an attacker can combine them to create an approximate lattice trapdoor. As Agrawal et al. discussed, the quality of such a trapdoor computed is worse than the actual trapdoor, thus such a trapdoor would be useless if the $\ell + 1$ preimage vectors that the attacker must compute have to be as short as the preimage vectors it received.

Unfortunately, setting the parameters carefully sidesteps just one limitation of the OM-ISIS assumption, but does not make the assumption truly robust. Simply put, we believe that providing an unrestricted preimage query access to the attacker is too strong. To further illustrate this, consider an even simpler attack that finds short preimage vectors without creating a lattice trapdoor. The attacker just makes two preimage queries (which correspond to preimage queries) — one on vector $\mathbf{0}$ and other on any non-zero vector $\mathbf{t} \in T$ (T is the target set). Let $\mathbf{z}_1$ and $\mathbf{z}_2$ be the respective preimage vectors. That is, $\mathbf{A} \cdot \mathbf{z}_1 = \mathbf{0}$ and $\mathbf{A} \cdot \mathbf{z}_2 = \mathbf{t}$. Given $\mathbf{z}_1$ and $\mathbf{z}_2$, an attacker simply output three distinct vectors $\mathbf{z}_2$, $\mathbf{z}_2 - \mathbf{z}_1$, and $\mathbf{z}_2 + \mathbf{z}_1$ as the preimages for the same target vector $\mathbf{t} \in T$.

Abstractly, the issue is that an adversary can perform simple linear combinations on preimage vectors, and such linear combinations map to appropriate linear combination of the corresponding target vectors. While this does not qualify as an attack on the two-round blind signature scheme of Agrawal et al. [AKSY22] (since any admissible attacker in their system must be generating preimages on ℓ+1 distinct vectors), this highlights that the OM-ISIS assumption is susceptible to arbitrary linear combination attacks.

Existence of such linear combination attack strategies are a big barrier to designing reusable non-interactive blind signatures. Our initial attempt to build NIBS as a generalization of the Agrawal et al. scheme turns out to be insecure. The attack is basically the same as the one described above. To ensure reusability, we set the final message as $\mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_\perp \\ \mathbf{z}_\perp \end{bmatrix}$ rather than just δ (where δ is was used in the receiver's public key). An attacker simply makes one presignature query for some receiver public key pk_R, and one presignature query for the all-zeros vector (as the public key), and combines them linearly to obtain > 2 valid presignatures for pk_R. One can easily show that the resulting final messages for all these preimage vectors will also be different, thereby constituting an efficient attack on one-more unforgeability of our basic NIBS scheme.

Randomized OM-ISIS assumption. To thwart the aforementioned linear-combination style attacks on the OM-ISIS assumption, we propose a new variant that we call *randomized one-more ISIS* assumption rOM-ISIS. Our goal here is twofold— (a) we want to turn OM-ISIS assumption into a more robust assumption such that there do not exist any efficient/practical attacks (irrespective of how the parameters are set), (b) we can design a non-interactive (as well as two-round) blind signature scheme which can be proven under the new assumption.

Our strategy is to prevent an attacker from learning preimages on arbitrary target vectors of its choice. This was the central property that was exploited by Agrawal et al. [AKSY22] in their practical attacks/cryptanalytic efforts. To this end, we make the challenger "re-randomize" each target vector (independently for each query) before computing its preimage. In turn, this takes away the attacker's prior advantage from obtaining distinct short preimages for the same target

vector (or any target of its choice more generally). Moroever, by carefully selecting how the per-query re-randomization happens, we can also avoid all known affine attacks that we discussed. Below we summarize our new randomized one-more ISIS assumption rOM-ISIS.

1. The challenger samples a challenge matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a randomization matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$ along with a large set of random target vectors $T \subset \mathbb{Z}_q^n$. It provides the attacker with $\mathbf{A}$, $\mathbf{B}$ and the vector set T.
2. $\mathcal{A}$ can make preimage queries for any target vector $\widehat{\mathbf{t}} \in \mathbb{Z}_q^n$ such that the challenger replies with a short vector $\widehat{\mathbf{x}}$ and a ± 1 vector $\widehat{\mathbf{y}} \in \{\pm 1\}^m$ such that $\mathbf{A} \cdot \widehat{\mathbf{x}} + \mathbf{B} \cdot \widehat{\mathbf{y}} = \widehat{\mathbf{t}}$.
3. rOM-ISIS assumption says that $\mathcal{A}$ cannot output $\ell + 1$ distinct vector tuples $\left\{(\mathbf{x}_j, \mathbf{y}_j, \mathbf{t}_j)\right\}_{j \in [\ell+1]}$ such that $\mathbf{A} \cdot \mathbf{x}_j + \mathbf{B} \cdot \mathbf{y}_j = \mathbf{t}_j$, $\mathbf{t}_j \in T$, $\mathbf{x}_j$ is sufficiently short, $\mathbf{y}_j$ is a ± 1 vector, *and* $\mathcal{A}$ made at most ℓ preimage queries.

Intuitively, the attacker now cannot truly select the preimage vector arbitrarily since the challenger randomizes the *actual* target vector as $(\mathbf{t} - \mathbf{B} \cdot \mathbf{y})$, where $\mathbf{y}$ is a random ± 1 vector. Since the attacker receives the vector $\widehat{\mathbf{y}}$ used for randomization, it is unclear whether we can reduce it to the standard ISIS assumption.[9] However, our preliminary cryptanalysis (cf. § 6.1) shows that it is more robust when compared with the OM-ISIS assumption. We believe that this new formulation could serve as a better lattice analogue of the one-more RSA assumption [BNPS03]. For example, we can also prove that a mild adaptation of the Agrawal et al. [AKSY22] two-round blind signature scheme is still secure under rOM-ISIS assumption, and now we no longer have set the parameters as carefully to avoid simple attacks as was done in [AKSY22]. This further illustrates the flexibility of our new assumption. Later, in Section 6, we describe the assumption in full detail and also provide some preliminary cryptanalysis.

Our final NIBS construction. With the above strengthening of the one-more ISIS assumption, we make some slight changes to our core design. Each signer additionally samples a random ± 1 vector $\mathbf{y}$, and computes the preimage for the syndrome for $(\mathsf{pk}_R - \mathbf{B} \cdot \mathbf{y})$, instead of just pk_R, i.e., it samples $\mathbf{z}$ such that $\mathbf{C} \cdot \mathbf{z} = \mathsf{pk}_R - \mathbf{B} \cdot \mathbf{y}$. The signer then explicitly sets $\mathbf{z}$ as the presignature and $\mathbf{y}$ as the nonce. Given this, the receiver creates a NIZK proof π stating that, given $\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{w}, \delta$, there exist vectors $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$ such that the following relation holds:

$$\mathbf{C} \cdot \mathbf{z} + \mathbf{B} \cdot \mathbf{y} = \mathbf{A} \cdot \mathbf{x} + \mathsf{H}(\delta) \ \wedge \ \mathbf{w} = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_\perp \\ \mathbf{z}_\perp \end{bmatrix} \wedge$$

$$\mathbf{y} = \begin{bmatrix} y_1 \ y_2 \cdots \end{bmatrix}^\mathsf{T}, \ \forall i : y_i \in \{\pm 1\} \ \wedge \ \|\mathbf{x}\|, \|\mathbf{z}\| \text{ are short.}$$

We describe our NIBS construction in detail in Section 7.1. We prove one-more unforgeability of our NIBS protocol under the rOM-ISIS assumption. Here, we run into a slight technical issue. Namely, when proving one-more unforgeability, the reduction will need to extract the adversary's $\ell + 1$ forgeries from the NIZK proof, but rewinding $\ell + 1$ times results in an exponential (in ℓ) soundness loss. Fortunately, one can easily apply the so-called "encryption-to-the-sky" trick

[9] Interestingly, one can easily show by a simple application of the leftover hash lemma [HILL99, DRS04, DORS08], that if the attacker does not receive $\widehat{\mathbf{y}}$, then this is as hard as the standard ISIS assumption.

[AKSY22, BLNS23a] to get straight-line extraction. More concretely, we modify the proof system to also require a proof of encryption to the witness $(\mathbf{x}, \mathbf{y}, \mathbf{z})$. Therefore the precise relation that the receiver must prove also includes $\mathsf{ct} = \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}\|\mathbf{z}; r)$, where the ciphertext ct and the PKE public key $\mathsf{pke.pk}$ also form part of the instance, and the encryption randomness r is included in the witness. As done is prior works, this can be efficiently proved using a linear relation. Finally, the receiver sets $(\mathbf{w}, \delta)$ as its message and (π, ct) as the corresponding signature. To verify a signature, one simply runs the verification algorithm for the NIZK.

In the security proof, the reduction now retains access to the PKE secret key. This allows the reduction to extract $\mathbf{x}_i$, $\mathbf{y}_i$ and $\mathbf{z}_i$ for all of the adversary's $\ell + 1$ forgeries. If the hash function is modeled as a random oracle, the reduction can simply answer all hash queries for $H(\delta)$ from the challenge set T; if it further sets the public matrix $\mathbf{A}$ as a matrix-matrix product of the rOM-ISIS challenge matrix $\mathbf{C}$, it can break rOM-ISIS. Observe that the matrix $\mathbf{A}$, statistically hides the receiver's secret $\mathbf{x}$ (by the leftover hash lemma). In the security proof , we combine this fact with the zero-knowledge property of the NIZK proof system, and the semantice security of the encryption scheme to prove receiver blindness.

Remark 2.2. Independently, Bootle et al. [BLNS23b] proposed the ISIS_f assumption. Under this assumption, the adversary is given access to a preimage oracle that outputs a randomly chosen $\widehat{y} \in D$ (D some domain) and a short vector $\widehat{\mathbf{x}} \in \mathbb{Z}_q^n$ such that for (public) matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ sampled from some distribution and (public) function $f : D \to \mathbb{Z}_q^n$, $\mathbf{A} \cdot \widehat{\mathbf{x}} = f(\widehat{y})$. The assumption then requires that it is hard for an adversary to output a value $y \in D$ and a short vector $\mathbf{x}$ ($\neq \widehat{\mathbf{x}}$) satisfying $\mathbf{A} \cdot \mathbf{x} = f(y)$. Importantly, this assumption is dependent on the choice of f. For instance, if f is the linear map $\mathbf{y} \mapsto \mathbf{B} \cdot \mathbf{y}$ for $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{y} \leftarrow\!\!\$\, \mathbb{Z}_q^m$, then the ISIS_f is trivially broken by linearly combining the query responses. [BLNS23b] also define an interactive version (that is reducible to the non-interactive ISIS_f) where the adversary is allowed to query for preimages under specific targets $\widehat{\mathbf{t}} \in \mathbb{Z}_q^l$ and the oracle outputs $(\widehat{y}, \widehat{\mathbf{x}})$ such that $\mathbf{A} \cdot \widehat{\mathbf{x}} = f(\widehat{y}) + \mathbf{C} \cdot \widehat{\mathbf{t}}$ for matrix $\mathbf{C} \in \mathbb{Z}_q^{n \times l}$ of the challenger's choice. Under this characterization, one might hope to abstractly view the rOM-ISIS assumption as an instantiation of interactive-ISIS_f where the function f linearly maps $\widehat{\mathbf{y}} \in \{\pm 1\}^m$ to $-\mathbf{B} \cdot \widehat{\mathbf{y}}$. However, if $\mathbf{C}$ is the $n \times n$ identity matrix as in rOM-ISIS, there is a non-negligible probability that outputs under this map can be efficiently linearly combined to give an *arbitrary* valid solution, so the ISIS_f problem is actually not hard for this function. On the other hand, the rOM-ISIS assumption remains hard as the adversary is also restricted to providing its (one-more) forgeries on a set T of target vectors chosen by the challenger.

Tagging our NIBS construction. We also discuss a simple modification to our NIBS construction to make it tagged. Recall that in tagged NIBS, the signer and receiver jointly agree on a value that will be treated as a public part of the signed message. To add such a public value τ to each blind signature, the signer computes a short preimage $\mathbf{z}$ such that $\mathbf{C} \cdot \mathbf{z} = \mathsf{pk}_R - \mathbf{B} \cdot \mathbf{y} + \mathsf{H}(\tau)$ using a secret trapdoor for $\mathbf{C}$. It then sends τ along with the preimage $\mathbf{z}$ and nonce $\mathbf{y}$ to the receiver. The receiver now includes τ in the instance of the NIZK relation, and generates an appropriate NIZK proof. The one-more unforgeability of this protocol (described in Section C) follows by a similar reduction to rOM-ISIS (in the random oracle model). Of course, here we have the additional $\mathsf{H}(\tau)$ term, but notice that the one-more unforgeability reduction, both models the hash function to the adversary and chooses the tag τ. Thus the challenger can choose $\mathsf{H}(\tau)$ in a way that later allows it to extract a short preimage for its rOM-ISIS challenge using the (one-more) forgery. The argument for receiver blindness follows directly from receiver blindness of the NIBS counterpart.

Nonce blindness. We also provide a NIBS scheme that satisfies basic nonce blindness in Section 8. Thus, this construction satisfies all the definitions originally provided in [Han23]. To make our NIBS scheme into a nonce-blind NIBS scheme, we make two important technical changes. First, the receiver's key generation algorithm samples and commits a random bit θ, so $\mathsf{pk}_R = \mathbf{A}\cdot\begin{bmatrix}\mathbf{x}\\ \theta\end{bmatrix}+\mathsf{H}(\delta)$. The second change is in Obtain, which now computes the message $\mathbf{w}$ as $(1-2\theta)(\mathbf{z}_\perp - \mathbf{x}_\perp)$.

The intuition behind the $(1-2\theta)$ term is that it prevents simple affine attacks on nonce blindness that can be carried our by subtracting two messages. Crucially, $\mathbf{w}$ does not leak any information about the nonce if $\mathbf{x}_\perp$ is chosen such that it "smudges-out" $\mathbf{z}_\perp$. As we mentioned earlier, this construction is secure under the basic nonce blindness definition, and we can easily upgrade it to k-nonce blindness by sampling $(k-1)$ smudging vectors as well as commit $(k-1)$ bits instead of a single θ bits. At a high level, we are using smudging along with a one-time pad style argument for proving basic nonce blindness, and this can be extended to k-nonce blindness by having more smudging terms.

2.6 Efficiency comparisons for our NIBS schemes

As a proof-of-concept, we provide estimates for the various parameter sizes in our scheme. Just as [AKSY22], we instantiate our construction 7.1 with Falcon [FHK+17] for signatures, [LPS10] for Regev-style encryption and [LNP22b] for the NIZK for linear relations. In table 2, we provide the public key, transcript and signature sizes for all our NIBS constructions. By weak blindness, we mean basic blindness [Han23], and by strong blindness, we mean our newly introduced definitions.

Construction	$\lvert\mathsf{pk}_R\rvert$	$\lvert\mathsf{psig}\rvert+$ $\lvert\mathsf{nonce}\rvert$	$\lvert\sigma\rvert$	**Blindness**
Lattice-based (7.1)	1.6 KB	0.96 KB	68 KB	Weak
Lattice-based (8.1)	$\lambda\cdot$1.6 KB	$\lambda\cdot$0.96 KB	$\lambda\cdot$68 KB	Weak
Circuit-private LHE (5.2)	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$	$\sim$ 0.5 KB	Strong
General-purpose NIZKs (D.1)	$\sim$ 2 KB	$\sim$ 0.5 KB	$\mathsf{poly}(\lambda)$	Strong

Table 2: Public key, transcript and signature sizes of our constructions.

3 Preliminaries

Notation. Let λ denote the security parameter, and PPT denote probabilistic polynomial-time. We denote the set of real numbers by $\mathbb{R}$ and the integers by $\mathbb{Z}$. We denote the set of all positive integers up to n as $[n] := \{1,\ldots,n\}$ and the set of all non-negative integers up to n as $[0,n] := \{0\}\cup[n]$.

For a vector $\mathbf{x}$ of even length, we write $\mathbf{x}_\top$ and $\mathbf{x}_\perp$ to be the top and bottom halves of $\mathbf{x}$ respectively, i.e. $\mathbf{x} = \begin{bmatrix}\mathbf{x}_\top\\ \mathbf{x}_\perp\end{bmatrix}$. Similarly, for any matrix $\mathbf{A} = [\mathbf{A}_\mathsf{L} \mid \mathbf{A}_\mathsf{R}] \in \mathbb{Z}_q^{n\times 2m}$, we denote its left and right halves as $\mathbf{A}_\mathsf{L}$ and $\mathbf{A}_\mathsf{R}$, respectively.

For a vector $\mathbf{x}$, we write its ℓ_2 norm as $\|\mathbf{x}\|_2$, often dropping the subscript and writing it simply as $\|\mathbf{x}\|$. We write the ℓ_∞ norm of $\mathbf{x}$ as $\|\mathbf{x}\|_\infty$.

3.1 Lattice preliminaries

An m-dimensional lattice $\mathcal{L}$ is a discrete additive subgroup of $\mathbb{R}^m$. Given positive integers n, m, q and a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we let $\Lambda_q^{\perp}(\mathbf{A})$ denote the lattice $\{\mathbf{x} \in \mathbb{Z}^m \ : \ \mathbf{A} \cdot \mathbf{x} = \mathbf{0} \mod q\}$. For $\mathbf{u} \in \mathbb{Z}_q^n$, we let $\Lambda_q^{\mathbf{u}}(\mathbf{A})$ denote the coset $\{\mathbf{x} \in \mathbb{Z}^m \ : \ \mathbf{A} \cdot \mathbf{x} = \mathbf{u} \mod q\}$.

Discrete Gaussians. Let ς be any positive real number. The Gaussian distribution $\mathcal{D}_{\varsigma}$ with parameter ς is defined by the probability distribution function $\rho_{\varsigma}(\mathbf{x}) = \exp(-\pi \|\mathbf{x}\|^2 / \varsigma^2)$. For any set $\mathcal{L} \subset \mathbb{R}^m$, define $\rho_{\varsigma}(\mathcal{L}) = \sum_{\mathbf{x} \in \mathcal{L}} \rho_{\varsigma}(\mathbf{x})$. The discrete Gaussian distribution $\mathcal{D}_{\mathcal{L},\varsigma}$ over $\mathcal{L}$ with parameter ς is defined by the probability distribution function $\rho_{\mathcal{L},\varsigma}(\mathbf{x}) = \rho_{\varsigma}(\mathbf{x}) / \rho_{\varsigma}(\mathcal{L})$ for all $\mathbf{x} \in \mathcal{L}$.

The following lemma (Lemma 4.4 of [MR04, GPV08]) shows that if the parameter ς of a discrete Gaussian distribution is small, then any vector drawn from this distribution will be short (with high probability).

Lemma 3.1. Let m, n, q be positive integers with $m > n$, $q \geq 2$. Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be a matrix of dimensions $n \times m$, $\varsigma = \tilde{\Omega}(n)$ and $\mathcal{L} = \Lambda_q^{\perp}(\mathbf{A})$. Then

$$\Pr[\|\mathbf{x}\| > \sqrt{m} \cdot \varsigma : \mathbf{x} \leftarrow\!\!\$\ \mathcal{D}_{\mathcal{L},\varsigma}] \leq \mathsf{negl}(n).$$

We will also require the following lemma (Lemma 4.4 of [Lyu12]) concerning the minimum-entropy of the discrete Gaussian distribution.

Lemma 3.2. Let $\mathcal{D}_{\mathbb{Z}^m,\varsigma}$ be the discrete Gaussian distribution over $\mathbb{Z}^m$ for any $m > 1$, with variance ς. Then, for any $\varsigma \geq 3/\sqrt{2\pi}$ we have that $\mathbf{H}_{\infty}\left(\mathcal{D}_{\mathbb{Z}^m,\varsigma}\right) \geq m$.

Lattice trapdoors. Lattices with trapdoors are lattices that are statistically indistinguishable from randomly chosen lattices, but have certain 'trapdoors' that allow efficient solutions to hard lattice problems.

Definition 3.3 ([Ajt96, GPV08])**.** For lattice parameters n, m, q with $m \geq \mathcal{O}(n \log q)$, a trapdoor lattice sampler consists of algorithms $\mathsf{TrapGen}$ and $\mathsf{SamplePre}$ with the following syntax:

- $\mathsf{TrapGen}(1^n, 1^m, q) \rightarrow (\mathbf{A}, T_{\mathbf{A}})$: The lattice generation algorithm is a randomized algorithm that takes as input the matrix dimensions n, m, modulus q, and outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a trapdoor $T_{\mathbf{A}}$.
- $\mathsf{SamplePre}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{u}, \varsigma) \rightarrow \mathbf{s}$: The presampling algorithm takes as input a matrix $\mathbf{A}$, trapdoor $T_{\mathbf{A}}$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$ and a parameter $\varsigma \in \mathbb{R}$ (which determines the length of the output vectors). It outputs a vector $\mathbf{s} \in \mathbb{Z}_q^m$ such that $\mathbf{A} \cdot \mathbf{s} = \mathbf{u}$ and $\|\mathbf{s}\| \leq \sqrt{m} \cdot \varsigma$.

These algorithms must satisfy the following properties:

1. *Well-Distributedness of Matrix.* The following distributions are statistically indistinguishable:
$$\{\mathbf{A} : (\mathbf{A}, T_{\mathbf{A}}) \leftarrow\!\!\$\ \mathsf{TrapGen}(1^n, 1^m, q)\} \approx_s \{\mathbf{A} : \mathbf{A} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times m}\}.$$
2. *Preimage Sampling:* For all $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow\!\!\$\ \mathsf{TrapGen}(1^n, 1^m, q)$, if $\varsigma = \omega(\sqrt{n \cdot \log q \cdot \log m})$, then the following distributions are statistically indistinguishable:
$$\{\mathbf{s} : \mathbf{u} \leftarrow\!\!\$\ \mathbb{Z}_q^n, \mathbf{s} \leftarrow\!\!\$\ \mathsf{SamplePre}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{u}, \varsigma)\} \approx_s \mathcal{D}_{\mathbb{Z}^m,\varsigma}.$$

These properties are satisfied by the gadget-based trapdoor lattice sampler of [MP12] for parameters m such that $m = \Omega(n \cdot \log q)$.

Bonsai lattice trapdoors. In this work, we will rely on the bonsai trick for lattice trapdoors [ABB10a, CHKP10]. Briefly, using the standard Bonsai trick, one can sample preimage vectors with a special property that a portion of the preimage vector will be sampled as a random Gaussian vector with appropriate norm, rather than from a pre-defined lattice coset. Throughout the sequel, we will routinely sample matrices of dimensions $n \times 2m$ such as $\mathbf{A} \in \mathbb{Z}_q^{n \times 2m}$, where

$$(\mathbf{A}_{\mathsf{L}}, T_{\mathbf{A}_{\mathsf{L}}}) \leftarrow\!\$\ \mathsf{TrapGen}(1^n, 1^m, q), \quad \mathbf{A}_{\mathsf{R}} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}.$$

To create a preimage $\mathbf{s} \in \mathbb{Z}_q^{2m}$, for any vector $\mathbf{u} \in \mathbb{Z}_q^n$, such that $\mathbf{A} \cdot \mathbf{s} = \mathbf{u}$, we simply sample $\mathbf{s}_\perp \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}^m, \varsigma}$, and $\mathbf{s}_\top \leftarrow\!\$\ \mathsf{SamplePre}(\mathbf{A}_{\mathsf{L}}, T_{\mathbf{A}_{\mathsf{L}}}, \mathbf{u} - \mathbf{A}_{\mathsf{R}} \cdot \mathbf{s}_\perp, \varsigma)$.

We refer to the above bonsai-based lattice trapdoors as $\mathsf{bLT} = (\mathsf{bLT.TrapGen}, \mathsf{bLT.SamplePre})$, where the trapdoor generation and preimage sampling algorithms are defined as above. Clearly, the algorithms for bLT satisfy the standard well-distributedness property, as well as the preimage sampling property. Moreover, it satisfies the following stronger *half-preimage* well distribution property:

Half-Preimage Well Distributedness. For all $(\mathbf{A}, T_{\mathbf{A}}) \leftarrow\!\$\ \mathsf{bLT.TrapGen}(1^n, 1^{2m}, q)$, every vector $\mathbf{u} \in \mathbb{Z}_q^n$, every ς, the following distributions are identical:

$$\{\mathbf{s}_\perp : \mathbf{s} \leftarrow\!\$\ \mathsf{SamplePre}(\mathbf{A}, T_{\mathbf{A}}, \mathbf{u}, \varsigma)\} \equiv \mathcal{D}_{\mathbb{Z}^m, \varsigma}.$$

3.2 Hardness assumptions

Definition 3.4 (SIS)**.** Let q, n, m, β be functions of security parameter λ. An instance of the $\mathsf{SIS}_{q,n,m,\beta}$ problem is a matrix $\mathbf{C} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}$, and a solution to the problem is a vector $\mathbf{z} \in \mathbb{Z}^m$ such that $\|\mathbf{z}\|_2 \leq \beta$ and $\mathbf{C} \cdot \mathbf{z} = \mathbf{0} \pmod q$.

Definition 3.5 (Inhomogeneous-SIS)**.** Let q, n, m, β be functions of security parameter λ. An instance of the $\mathsf{ISIS}_{q,n,m,\beta}$ problem is a matrix $\mathbf{C} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{y} \leftarrow\!\$\ \mathbb{Z}_q^m$, and a solution to the problem is a vector $\mathbf{z} \in \mathbb{Z}^m$ such that $\|\mathbf{z}\|_2 \leq \beta$ and $\mathbf{C} \cdot \mathbf{z} = \mathbf{y} \pmod q$.

For suitably chosen parameters, the ISIS problem is at least as hard as certain worst-case lattice problems [Ajt96, MR04, GPV08].

We also recall the one-more ISIS assumption (OM-ISIS), as defined by Agrawal et al. [AKSY22].

Definition 3.6 (One-more ISIS)**.** Let $q, n, m, \varsigma, \beta$ be functions of security parameter λ. Consider the following experiment:

1. The challenger uniformly samples a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, and sends $\mathbf{A}$ to adversary $\mathcal{A}$.
2. $\mathcal{A}$ adaptively makes queries of the following types to the challenger, in any order.

   Syndrome queries. $\mathcal{A}$ requests for a challenge vector, to which the challenger replies with a uniformly sampled vector $\mathbf{t} \leftarrow\!\$\ \mathbb{Z}_q^n$. We denote the set of received vectors by S.

   Preimage queries. $\mathcal{A}$ queries a vector $\widehat{\mathbf{t}} \in \mathbb{Z}_q^n$., to which the challenger replies with a short vector $\widehat{\mathbf{x}} \in \mathbb{Z}_q^m$ such that $\mathbf{A} \cdot \widehat{\mathbf{x}} = \widehat{\mathbf{t}}$ and $\|\widehat{\mathbf{x}}\| \leq \varsigma\sqrt{m}$. Let ℓ denote the total number of preimage queries.

3. Finally, $\mathcal{A}$ outputs $\ell+1$ pairs of the form $\left\{(\mathbf{x}_j, \mathbf{t}_j)\right\}_{j\in[\ell+1]}$. And, $\mathcal{A}$ wins if

$$\forall j \in [\ell+1], \qquad \mathbf{A}\cdot\mathbf{x}_j = \mathbf{t}_j, \text{ and } \left\|\mathbf{x}_j\right\| \leq \beta \text{ and } \mathbf{t}_j \in S.$$

The $\mathsf{OM\text{-}ISIS}_{q,n,m,\varsigma,\beta}$ assumption states that for every PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins is $\mathsf{negl}(\lambda)$.

In words, the assumption states that it is hard to find $\ell+1$ short preimages from a set $\mathcal{S}$ of syndromes, even when given up to ℓ inversions of arbitrary syndromes not in $\mathcal{S}$, for any polynomially bounded ℓ. Equivalently, it is hard to forge $\ell+1$ GPV signatures [GPV08] given up to ℓ inversions of arbitrary syndromes. However, the main reason this is not known to be reducible to standard SIS (unlike security of GPV signatures) is that $\mathcal{A}$ *can ask preimage queries on any arbitrary vector of its choice*.

3.3 Randomness extraction

The min-entropy of a random variable X is defined as $\mathbf{H}_\infty(X) \stackrel{\text{def}}{=} -\log_2(\max_x \Pr[X = x])$. Let $\mathsf{SD}(X, Y)$ denote the statistical distance between two random variables X and Y. Below we state the Leftover Hash Lemma (LHL) from [HILL99, DRS04, DORS08].

Theorem 3.7. Let $\mathcal{H} = \{h : X \to Y\}_{h\in\mathcal{H}}$ be a universal hash family, then for any random variable W taking values in X, the following holds

$$\mathsf{SD}((h, h(W)), (h, U_Y)) \leq \frac{1}{2}\sqrt{2^{-\mathbf{H}_\infty(W)}\cdot|Y|},$$

where U_Y denotes uniform distribution over Y.

We will use the following corollary, which follows from the Leftover Hash Lemma.

Corollary 3.8. Let $\ell > m\cdot n\log_2 q + \omega(\log n)$, q a prime, and m, n are positive integers. Let $\mathbf{R}$ be an $k\times m$ matrix chosen as per distribution $\mathcal{R}$, where $k = k(n)$ is polynomial in n and $\mathbf{H}_\infty(\mathcal{R}) = \ell$. Let $\mathbf{A}$ and $\mathbf{B}$ be matrices chosen uniformly in $\mathbb{Z}_q^{n\times k}$ and $\mathbb{Z}_q^{n\times m}$, respectively. Then the statistical distance between the following distributions is negligible in n.

$$\{(\mathbf{A}, \mathbf{A}\cdot\mathbf{R})\} \approx_s \{(\mathbf{A}, \mathbf{B})\}$$

Lemma 3.9 (Smudging Lemma [AJL+12, Lemma 2.1, paraphrased])**.** Let B_1, B_2 be two polynomials over the integers and let $\mathcal{D} = \{\mathcal{D}(\lambda)\}_\lambda$ be any B_1-bounded distribution family. Let $U = \{U(\lambda)\}_\lambda$ and $U(\lambda)$ denote the uniform distribution over integers $[-B_2(\lambda), B_2(\lambda)]$. The family of distributions $\mathcal{D}$ and U is statistically indistinguishable, $\mathcal{D} + U \approx_s U$, if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $B_1(\lambda)/B_2(\lambda) \leq \mathsf{negl}(\lambda)$.

3.4 Cryptographic building blocks

We also recall the standard cryptographic building blocks that will be essential to this work.

3.4.1 Pseudo Random Functions

A pseudorandom function (PRF) $F : \{0,1\}^\lambda \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ is a keyed function[10], that takes a λ-bit key as input, and on input $x \in \{0,1\}^\lambda$, it outputs a value $y = F_K(x)$. *(Throughout the paper, we use $F_K(x)$ as a shorthand for PRF evaluation.)*

Definition 3.10 (Pseudorandomness)**.** A PRF F is said to be secure if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all λ, the following holds:

$$\Pr\left[\mathcal{A}^{O_{K,b}(\cdot)}(1^\lambda) = b \ : \ K \leftarrow\$ \{0,1\}^\lambda,\ b \leftarrow\$ \{0,1\}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where oracle $O_{K,b}(\cdot)$ is defined as $F_K(\cdot)$ if $b = 0$, otherwise it is defined as a random function from $\{0,1\}^\lambda \to \{0,1\}^\lambda$.

3.4.2 Perfectly binding commitments

A commitment scheme COM consists of the following algorithms.

$\mathsf{Setup}(1^\lambda, 1^n) \to \mathsf{crs}$. The setup algorithm takes as input the security parameter λ, message length n, and outputs a crs crs.

$\mathsf{Com}(\mathsf{crs}, m; r) \to c$. The commit algorithm that takes as input the crs crs, message $m \in \{0,1\}^n$, and randomness $r \in \{0,1\}^\lambda$. It outputs a commitment c. *(We assume for simplicity that r is always a λ-bit string. Note that this follows w.l.o.g., and is not an extra assumption.)*

$\mathsf{Verify}(\mathsf{crs}, m, c, r) \to 0/1$. The verification algorithm takes as input the crs crs, message m, commitment c, and an opening/randomness r. It outputs either 0 or 1 to signal validity of the opening.

Correctness. A commitment scheme COM is correct if for every $\lambda, n \in \mathbb{N}$, crs $\mathsf{crs} \leftarrow\$ \mathsf{Setup}(1^\lambda, 1^n)$, any message $m \in \{0,1\}^n$, the following holds

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, m, c, r) = 1 \ : \ r \leftarrow\$ \{0,1\}^\lambda,\ c = \mathsf{Com}(\mathsf{crs}, m; r)\right] = 1.$$

Perfect binding. A commitment scheme is said to be perfectly binding if no commitment can have valid openings for two different messages.

Definition 3.11 (Perfect binding)**.** A commitment scheme COM is perfectly binding if for all $\lambda, n \in \mathbb{N}$, every crs $\mathsf{crs} \leftarrow\$ \mathsf{Setup}(1^\lambda, 1^n)$, for every (c, m_1, r_1, m_2, r_2) such that $m_1 \neq m_2$, the following holds for at least one $i \in \{1,2\}$:

$$\Pr[\mathsf{Verify}(\mathsf{crs}, m_i, c, r_i) = 1] = 0.$$

Computational hiding. A commitment scheme is said to be computationally hiding if a commitment hides the messages when the openings are hidden.

Definition 3.12 (Hiding)**.** A commitment scheme COM is computationally hiding if for every stateful PPT attacker $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds

$$\Pr\left[\begin{array}{ll} \mathcal{A}(c) = b & 1^n \leftarrow\$ \mathcal{A}(1^\lambda),\ \mathsf{crs} \leftarrow\$ \mathsf{Setup}(1^\lambda, 1^n) \\ \wedge\ m_0, m_1 \in \{0,1\}^n & : \ (m_0, m_1) \leftarrow\$ \mathcal{A}(\mathsf{crs}),\ b \leftarrow\$ \{0,1\} \\ & c \leftarrow\$ \mathsf{Com}(\mathsf{crs}, m_b) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

[10] For simplicity, we fix the key space, input space, and output space to be λ-bit strings.

3.4.3 Public key encryption

A public key encryption (PKE) scheme PKE consists of the following polynomial-time algorithms.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$. The setup algorithm takes as input the security parameter λ, and outputs a public-secret key pair $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{ct}$. The encryption algorithm takes as input a public key pk and a message $m \in \{0,1\}^\lambda$, and outputs a ciphertext ct.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to m$. The decryption algorithm takes as input a secret key sk and a ciphertext ct, and outputs a message m.

The scheme satisfies correctness if for all λ, $m \in \{0,1\}^\lambda$, $(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda)$, and every ciphertext $\mathsf{ct} \leftarrow\!\!\$\ \mathsf{Enc}(\mathsf{pk}, m)$, we have that $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = m$. Further, the standard IND-CPA security notion is defined as follows.

Definition 3.13. A public key encryption scheme PKE is IND-CPA secure if for every stateful PPT adversary $\mathcal{A}$, there exists a negligible functions $\mathsf{negl}(\cdot)$, such that for every $\lambda \in \mathbb{N}$

$$\Pr\left[\mathcal{A}(\mathsf{ct}) = b : \begin{array}{c}(\mathsf{pk}, \mathsf{sk}) \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda); b \leftarrow\!\!\$\ \{0,1\} \\ (m_0, m_1) \leftarrow\!\!\$\ \mathcal{A}(\mathsf{pk}); \mathsf{ct} \leftarrow\!\!\$\ \mathsf{Enc}(\mathsf{pk}, m_b)\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

3.4.4 Signature schemes

A signature scheme $\mathsf{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify})$ with message space $\mathcal{M}$ consists of three algorithms, as follows:

$\mathsf{Setup}(1^\lambda)$ is a randomized algorithm that takes security parameter λ as input and returns a pair of keys $(\mathsf{sk}, \mathsf{vk})$, where sk is the signing key and vk is the verification key.

$\mathsf{Sign}(\mathsf{sk}, m)$ is a possibly randomized algorithm that takes as input the signing key sk, and a message m, and returns a signature σ.

$\mathsf{Verify}(\mathsf{vk}, m, \sigma)$ is a determistic algorithm that takes as input the verification key vk, a message m, and a signature σ, and outputs 1 (accepts) if verification succeeds, and 0 (rejects) otherwise.

A signature scheme satisfies correctness if for all $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$, and every signing-verification key pair $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Setup}(1^\lambda)$, every signature $\sigma \leftarrow\!\!\$\ \mathsf{Sign}(\mathsf{sk}, m)$, $\mathsf{Verify}(\mathsf{vk}, m, \sigma) = 1$.

Definition 3.14. A signature scheme $\mathcal{S} = (\mathsf{Setup}, \mathsf{Sign}, \mathsf{Verify})$ is a secure signature scheme if for every PPT attacker $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds

$$\Pr\left[\mathsf{Verify}(\mathsf{vk}, m^*, \sigma^*) = 1 : \begin{array}{c}(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda) \\ (m^*, \sigma^*) = \mathcal{A}^{\mathsf{Sign}(\mathsf{sk}, \cdot)}(1^\lambda, \mathsf{vk})\end{array}\right] \leq \mathsf{negl}(\lambda),$$

and $\mathcal{A}$ should never have queried m^* to Sign oracle.

3.4.5 Non-interactive zero knowledge (NIZK)

A NIZK proof system for language $\mathcal{L}$ consists of the following polynomial time algorithms:

$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$. The setup algorithm takes as input the security parameter λ, and outputs a crs crs.

$\mathsf{Prove}(\mathsf{crs}, x, \omega) \to \pi$. The prover algorithm takes as input a crs, an instance $x \in \mathcal{L}$, and a witness ω. It outputs a proof π.

$\mathsf{Verify}(\mathsf{crs}, x, \pi) \to 0/1$. The verification algorithm takes as input a crs, an instance x, and a proof π. It outputs a bit to signal whether the proof is valid or not.

A proof system is complete if for every $\lambda \in \mathbb{N}$, crs $\mathsf{crs} \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda)$, any instance $x \in \mathcal{L}$ with corresponding witness ω, the following holds

$$\Pr\left[\mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 \ : \ \pi \leftarrow\!\!\$\ \mathsf{Prove}(\mathsf{crs}, x, \omega)\right] = 1.$$

Furthermore, we require the following properties.

Definition 3.15 (Soundness)**.** A proof system $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ is computationally sound if for every stateful PPT attacker $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds

$$\Pr\left[\begin{array}{l|ll} \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 & : & \mathsf{crs} \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda) \\ \wedge\ x \notin \mathcal{L} & & (x, \pi) \leftarrow\!\!\$\ \mathcal{A}(1^\lambda, \mathsf{crs}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 3.16 ((Multi-theorem) Zero-knowledge)**.** A proof system $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ is computationally zero-knowledge if there exists a stateful PPT simulator $\mathcal{S}$ such that for every stateful PPT attacker $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds

$$\Pr\left[\begin{array}{l|ll} \mathcal{A}(\{\pi_{i,b}\}_i) = b & & b \leftarrow\!\!\$\ \{0,1\} \\ \wedge \left(\begin{array}{l} \forall i \in [\ell],\ \omega_i \text{ is a valid} \\ \text{witness for } x_i \in \mathcal{L} \end{array}\right) & : & \mathsf{crs}_0 \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda) \\ & & \mathsf{crs}_1 \leftarrow\!\!\$\ \mathcal{S}(1^\lambda) \\ & & \{(x_i, \omega_i)\}_{i\in[\ell]} \leftarrow\!\!\$\ \mathcal{A}(1^\lambda, \mathsf{crs}_b) \\ & & \forall i \in [\ell],\ \pi_{i,0} \leftarrow\!\!\$\ \mathsf{Prove}(\mathsf{crs}_0, x_i, \omega_i) \\ & & \{\pi_{i,1}\}_i \leftarrow\!\!\$\ \mathcal{S}(\mathsf{crs}_0, \{x_i\}_i) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

Definition 3.17 (Knowledge extractor)**.** A proof system $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ has a knowledge extractor if there exists a PPT extractor $\mathcal{E}$ such that for every stateful PPT attacker $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, the following holds

$$\Pr\left[\begin{array}{l|ll} \mathsf{Verify}(\mathsf{crs}, x, \pi) = 1 & & \tau \leftarrow\!\!\$\ \{0,1\}^\lambda \\ \wedge \left(\begin{array}{l} \omega = \mathcal{E}(\tau, x, \pi) \text{ is not a} \\ \text{valid witness for } x \in \mathcal{L} \end{array}\right) & : & \mathsf{crs} \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda; \tau) \\ & & (x, \pi) \leftarrow\!\!\$\ \mathcal{A}(1^\lambda, \mathsf{crs}) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where τ denotes the randomness used for running the setup algorithm, and we assume (w.l.o.g.) that $|\tau| = \lambda$.

Remark 3.18. In our security proofs below, we make regular use of straight-line extraction. This can be achieved generically using PKE by encrypting the witness under a shared public encryption key and providing the ciphertext along with the proof. Additionally, the zero-knowledge proof itself must contain a proof of correct computation of the ciphertext. Looking ahead to the security proofs, the challenger will possess a secret decryption key corresponding to the public encryption key enabling straight-line extraction.

3.4.6 Leveled homomorphic encryption

Let $\mathcal{C}_d$ denote the class of boolean valued circuits of depth d. A leveled homomorphic encryption scheme $\mathcal{LHE}$ with message space $\{0,1\}$ for circuit class $\{\mathcal{C}_d\}_{d\in\mathbb{N}}$ consists of the following polynomial time algorithms:

$\mathsf{Setup}(1^\lambda, 1^d) \to (\mathsf{sk}, \mathsf{ek})$ The setup algorithm takes as input the security parameter λ, bound on circuit depth d and outputs a secret key sk and evaluation key ek.

$\mathsf{Enc}(\mathsf{sk}, m \in \{0,1\}) \to \mathsf{ct}$ The encryption algorithm takes as input a secret key sk, message $m \in \{0,1\}$ and outputs a ciphertext ct.

$\mathsf{Eval}(\mathsf{ek}, C \in \mathcal{C}_d, \mathbf{ct}) \to \mathsf{ct}'$ The evaluation algorithm takes as input an evaluation key ek, a circuit $C \in \mathcal{C}_d$, a sequence of ciphertexts $\mathbf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$ for some $\ell > 0$ and outputs a ciphertext $\widehat{\mathsf{ct}}$. Here ℓ denotes the input length of C.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \to x$ The decryption algorithm takes as input a secret key sk and ciphertext ct and outputs $x \in \{0,1\} \cup \{\perp\}$.

Correctness. The scheme $\mathcal{LHE}$ is said to be (perfectly) correct if for all security parameter λ, circuit-depth bound d, $(\mathsf{sk}, \mathsf{ek}) \leftarrow \mathsf{Setup}(1^\lambda, 1^d)$, circuit $C \in \mathcal{C}_d$ and messages $m_1, \ldots, m_\ell \in \{0,1\}$, every ciphertext $\mathsf{ct}_i \leftarrow\!\!\$\, \mathsf{Enc}(\mathsf{sk}, m_i)$ where ℓ denotes input length of C, the following holds:

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{Eval}(\mathsf{ek}, C, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)) = C(m_1, \ldots, m_\ell)\right] = 1.$$

Definition 3.19 (Circuit privacy)**.** An $\mathcal{LHE}$ scheme is said to be circuit private if there exists a PPT algorithm Sim such that for every $d \in \mathbb{N}$ any circuit $C \in \mathcal{C}_d$ with input length $\ell = \mathsf{poly}(\lambda)$, and any sequence of message bits $m_1, \ldots, m_\ell \in \{0,1\}$, the following holds:

$$(\mathsf{ek}, \mathsf{Eval}(\mathsf{ek}, C, (\mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)), \mathsf{ct}_1, \ldots, \mathsf{ct}_\ell) \approx_c \left(\mathsf{ek}, \widehat{\mathsf{ct}}, \mathsf{ct}_1, \ldots, \mathsf{ct}_\ell\right)$$

where $(\mathsf{sk}, \mathsf{ek}) \leftarrow\!\!\$\, \mathsf{Setup}(1^\lambda, 1^d)$, $\mathsf{ct}_i \leftarrow\!\!\$\, \mathsf{Enc}(\mathsf{sk}, m_i)\ \forall i \in [\ell]$, $\widehat{\mathsf{ct}} = \mathsf{Sim}(\mathsf{ek}, C(m_1, \ldots, m_\ell), \mathsf{ct}_1, \ldots, \mathsf{ct}_\ell)$.

4 A Stronger Model for Non-Interactive Blind Signatures

In a NIBS system, a signer issues a random *presignature* psig for any receiver R with public key pk_R, such that the receiver R can extract a blind signature σ for a random message μ using its secret key sk_R. Syntactically, a non-interactive blind signature scheme consists of the following polynomial-time algorithms:

$\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$**.** On input the security parameter λ, the global setup algorithm outputs a set of public parameters pp. All the remaining algorithms take pp as an input, but for notational clarity, we usually omit it as an explicit input.

$\mathsf{KeyGen}_S(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{vk})$. This corresponds to the signer's key generation algorithm. On input pp, it samples a public-secret key pair $(\mathsf{sk}, \mathsf{vk})$.

$\mathsf{KeyGen}_R(\mathsf{pp}) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. This corresponds to the receiver's key generation algorithm. On input pp, it samples a public-secret key pair $(\mathsf{sk}_R, \mathsf{pk}_R)$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \to (\mathsf{psig}, \mathsf{nonce})$. This is a randomized algorithm that is run by the signer. It takes as input the signer's secret key sk as well as a receiver's public key pk_R. It then outputs a presignature psig along with nonce which represents (a portion of) the signer's random coins.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}, \mathsf{nonce})) \to (\mu, \sigma)$. This algorithm corresponds to the receiver's blind signature extraction algorithm. Given the receiver's secret key sk_R as an input, along with a verification key vk and presignature-nonce pair $(\mathsf{psig}, \mathsf{nonce})$, it outputs a message-signature pair (μ, σ) or aborts (in which case it outputs $\perp$).

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0, 1\}$. This is the signature scheme verification algorithm that takes as input a verification key and message-signature pair, and outputs 0/1.

Correctness. A non-interactive blind signature scheme satisfies correctness if for every security parameter $\lambda \in \mathbb{N}$, $\mathsf{pp} \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda)$, $(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\ \mathsf{KeyGen}_S(\mathsf{pp})$, $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow\!\!\$\ \mathsf{KeyGen}_R(\mathsf{pp})$, the following holds:

$$\Pr[\mathsf{Verify}(\mathsf{vk}, \mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, \mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R))) = 1] = 1,$$

where the probability is taken over the random coins of Issue and Obtain.

Remark 4.1 (Comparing with [Han23], and the reusability property)**.** The above formalization of non-interactive blind signatures is nearly identical to the syntax introduced by Hanzlik, except we do not regard nonce as an input supplied to the Issue algorithm but as an output. We essentially simplify the syntax by viewing nonce as a signer's *public (random) coins*. The Obtain algorithm receives both $(\mathsf{psig}, \mathsf{nonce})$ (as in [Han23]).

Paraphrasing [Han23], the purpose of nonce is to ensure reusability of a receiver's public key for obtaining multiple message-signature pairs from a single signer. We emphasize that both formulations are equivalent, and moreover, our formulation seems syntactically cleaner (and closer to syntax for traditional blind signatures) as well as helps defining an important reusability property that is necessary to avoid vacuous solutions. Further, we do not see any advantage in defining nonce as anything other than signer's public randomness since the goal here is to have the signing process be non-interactive, and treating nonce as an extra input is inconsistent with that goal.

Equivalence of both formalizations. It is straightforward to see that NIBS scheme satisfying the above syntax can be generically translated into satisfying the syntax from [Han23]. The idea is to generate the randomness for our Issue algorithm by evaluating a PRF on the nonce value and the public key pk_R provided as input in Hanzlik's version.[11] For completeness, we provide this in Appendix A.

[11] Concretely, $\mathsf{Issue}^{\mathsf{Hanzlik}}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce})$ outputs the presignature as $(\mathsf{psig}', \mathsf{nonce}')$ where $(\mathsf{psig}', \mathsf{nonce}') \leftarrow \mathsf{Issue}^{\mathsf{Ours}}(\mathsf{sk}, \mathsf{pk}_R; F_K(\mathsf{nonce}, \mathsf{pk}_R))$.

A vacuous NIBS scheme. Consider a NIBS scheme where the Issue algorithm is deterministic and it always outputs the same nonce value (or, following Hanzlik's notation, the Issue algorithm ignores the nonce value entirely). Such a NIBS scheme clearly does not satisfy any meaningful notion of reusability, since for each receiver's public key a signer generates at most one presignature. Unfortunately, this is still a valid NIBS scheme as per existing definitions, and furthermore, it satisfies the nonce blindness property [Han23, Definition 17] vacuously, i.e. even if the adversary outputs $(\mathsf{psig}_0, \mathsf{nonce}_0) \neq (\mathsf{psig}_1, \mathsf{nonce}_1)$ it would still not be able to distinguish for (μ_b, σ_b). The main issue is that the existing definitions do not disallow schemes where the Issue and Obtain algorithms ignore the nonce parameter.

A simple and sound approach to capture reusability. Our proposal is to simply define reusability of NIBS schemes directly. Rather than making the nonce parameter explicit, we view it as a portion of the signer's random coins. Thus, we define reusability of a NIBS scheme as the property that any receiver can obtain two distinct messages (along with valid signatures) for two randomly generated presignature-nonce pairs (for the same receiver's public key) with all but negligible probability. Formally:

Definition 4.2 (Reusability)**.** A NIBS scheme $\mathcal{S}$ satisfies the reusability property, if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{l|r} \mathsf{nonce}_0 = \mathsf{nonce}_1 & \mathsf{pp} \leftarrow\!\$\ \mathsf{Setup}(1^\lambda) \\ \vee\ \mu_0 = \mu_1 \ : & (\mathsf{sk}, \mathsf{vk}) \leftarrow\!\$\ \mathsf{KeyGen}_S(\mathsf{pp}), (\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow\!\$\ \mathsf{KeyGen}_R(\mathsf{pp}) \\ & \forall b \in \{0,1\} : (\mathsf{psig}_b, \mathsf{nonce}_b) \leftarrow\!\$\ \mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \\ & \forall b \in \{0,1\} : (\mu_b, \sigma_b) \leftarrow\!\$\ \mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}_b, \mathsf{nonce}_b)) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Next, we provide the standard notion of *one-more* unforgeability for blind signatures, specialized for the NIBS setting [Han23].[12]

Definition 4.3 (One-more unforgeability)**.** A NIBS scheme $\mathcal{S}$ satisfies one-more unforgeability, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{l|r} \bigwedge_{i\in[\ell+1]} \mathsf{Verify}(\mathsf{vk}, \mu_i, \sigma_i) = 1 & \mathsf{pp} \leftarrow\!\$\ \mathsf{Setup}(1^\lambda) \\ \wedge \left(\bigwedge_{i\neq j\in[\ell+1]} \mu_i \neq \mu_j\right) \ : & (\mathsf{sk}, \mathsf{vk}) \leftarrow\!\$\ \mathsf{KeyGen}_S(\mathsf{pp}) \\ & \{(\mu_i, \sigma_i)\}_{i=1}^{\ell+1} \leftarrow\!\$\ \mathcal{A}^{O_{\mathsf{sk}}(\cdot)}(\mathsf{vk}) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $O_{\mathsf{sk}}(\cdot)$ takes as input a receiver's public key pk_{R_i}, and outputs a presignature-nonce pair $(\mathsf{psig}_i, \mathsf{nonce}_i)$ by running $\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_{R_i})$, and $\mathcal{A}$ is an admissible adversary iff $\mathcal{A}$ makes at most ℓ queries to O_{sk}.

In this work, we propose stronger notions of inter/intra-receiver blindness for NIBS schemes. The existing approaches to capture blindness for NIBS do not allow an adversary to learn any correlation between presignature-nonce pairs and their corresponding blind signature-message pairs. Unfortunately, if a server learns the receiver's identity for just one blind signature, then

[12] We want to remark that in [Han23], in the one-more unforgeability security experiment, the adversary can select nonce during each Issue query. In our definition, the challenger samples nonce since it is treated as randomness of Issue. However, both definitions are equivalent since a signer can use a PRF to generate the actual randomness from an input nonce as described in Remark 4.1.

existing definitions are insufficient in providing any notion of blindness from such attacks. In order to protect from such advanced attackers that can bypass the blindness property for receivers on some selected blind signatures, we introduce stronger notions of inter/intra-receiver blindness properties that we refer to as *strong receiver/nonce blindness*.

Definition 4.4 (Strong receiver blindness)**.** A NIBS scheme $\mathcal{S}$ satisfies *strong* receiver blindness, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{r} \mathcal{A}^{O_{\mathsf{sk}_{R_0},\mathsf{sk}_{R_1}}(\cdot,\cdot,\cdot)}(\mu_{\hat{b}},\sigma_{\hat{b}},\mu_{1-\hat{b}},\sigma_{1-\hat{b}}) = \hat{b} : \\ \mathsf{pp} \leftarrow\!\$\, \mathsf{Setup}(1^\lambda),\ \hat{b} \leftarrow\!\$\, \{0,1\}, \\ \forall b \in \{0,1\} :\ (\mathsf{sk}_{R_b},\mathsf{pk}_{R_b}) \leftarrow\!\$\, \mathsf{KeyGen}_R(\mathsf{pp}) \\ (\mathsf{vk},(\mathsf{psig}_b,\mathsf{nonce}_b)_b) \leftarrow\!\$\, \mathcal{A}^{O_{\mathsf{sk}_{R_0},\mathsf{sk}_{R_1}}(\cdot,\cdot,\cdot)}(\mathsf{pk}_{R_0},\mathsf{pk}_{R_1}) \\ \forall b \in \{0,1\} :\ (\mu_b,\sigma_b) \leftarrow\!\$\, \mathsf{Obtain}(\mathsf{sk}_{R_b},\mathsf{vk},(\mathsf{psig}_b,\mathsf{nonce}_b)) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where oracle $O_{\mathsf{sk}_{R_0},\mathsf{sk}_{R_1}}$, on the i-th query $(b^{(i)},\mathsf{vk}^{(i)},(\mathsf{psig}^{(i)},\mathsf{nonce}^{(i)}))$, outputs $\mathsf{Obtain}(\mathsf{sk}_{R_{b^{(i)}}},\mathsf{vk}^{(i)},(\mathsf{psig}^{(i)},\mathsf{nonce}^{(i)}))$. That is, $O_{\mathsf{sk}_{R_0},\mathsf{sk}_{R_1}}$ provides $\mathcal{A}$ oracle access to the Obtain algorithm w.r.t. $\mathsf{sk}_{R_0},\mathsf{sk}_{R_1}$. We say that $\mathcal{A}$ is an admissible adversary iff:

- $\sigma_0,\sigma_1 \neq \perp$ (i.e., Obtain algorithm does not abort), and
- $\mathsf{nonce}_0 \neq \mathsf{nonce}^{(i)}$ and $\mathsf{nonce}_1 \neq \mathsf{nonce}^{(i)}$ for all i. (That is, $\mathcal{A}$ cannot make an Obtain query with nonce value to be either of the challenge nonce values.)

Definition 4.5 (Strong nonce blindness)**.** A NIBS scheme $\mathcal{S}$ satisfies nonce blindness, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{r} \mathcal{A}^{O_{\mathsf{sk}_R}(\cdot,\cdot)}(\mu_{\hat{b}},\sigma_{\hat{b}},\mu_{1-\hat{b}},\sigma_{1-\hat{b}}) = \hat{b} : \\ \mathsf{pp} \leftarrow\!\$\, \mathsf{Setup}(1^\lambda),\ (\mathsf{sk}_R,\mathsf{pk}_R) \leftarrow\!\$\, \mathsf{KeyGen}_R(\mathsf{pp}) \\ (\mathsf{vk},(\mathsf{psig}_b,\mathsf{nonce}_b)_b) \leftarrow\!\$\, \mathcal{A}^{O_{\mathsf{sk}_R}(\cdot,\cdot)}(\mathsf{pk}_R),\ \hat{b} \leftarrow\!\$\, \{0,1\} \\ \forall b \in \{0,1\} :\ (\mu_b,\sigma_b) \leftarrow\!\$\, \mathsf{Obtain}(\mathsf{sk}_R,\mathsf{vk},(\mathsf{psig}_b,\mathsf{nonce}_b)) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where oracle O_{sk_R}, on the i-th query $(\mathsf{vk}^{(i)},(\mathsf{psig}^{(i)},\mathsf{nonce}^{(i)}))$, outputs $\mathsf{Obtain}(\mathsf{sk}_R,\mathsf{vk}^{(i)},(\mathsf{psig}^{(i)},\mathsf{nonce}^{(i)}))$. That is, O_{sk_R} provides $\mathcal{A}$ oracle access to the Obtain algorithm w.r.t. sk_R. We say that $\mathcal{A}$ is an admissible adversary iff:

- $\sigma_0,\sigma_1 \neq \perp$ (i.e., Obtain algorithm does not abort), and
- $\mathsf{nonce}_0 \neq \mathsf{nonce}^{(i)}$ and $\mathsf{nonce}_1 \neq \mathsf{nonce}^{(i)}$ for all i. (That is, $\mathcal{A}$ cannot make an Obtain query with nonce value to be either of the challenge nonce values.)

5 NIBS from Circuit Private LHE

The unforgeability and blindness definitions discussed thus far have been defined in the general chosen-key model where the attacker can specify arbitrary (possibly malformed) receiver and

verification keys. In this section, we will consider definitions under the knowledge of secret keys (KOSK) model [MOR01, Bol03] as define next. We point out that any NIBS scheme satisfying one-more (strong) unforgeability, (strong) receiver blindness, and (strong) nonce blindness in the *KOSK* model can be generically upgraded into another NIBS scheme satisfying the same security properties by using a non-interactive zero-knowledge argument of knowledge (NIZKAoK) scheme (cf. Appendix B).

5.1 Knowledge of secret keys model

In the knowledge of secret keys (KOSK) model for NIBS, we require the attacker to specify the corresponding secret keys along with the respective verification/public keys. That is, the attacker supplies the receiver's secret key in unforgeability experiment and the signer's signing key in blindness experiments, respectively.

Definition 5.1 (strong receiver blindness in KOSK)**.** Recall the strong receiver blindness security experiment from Definition 4.4. Consider another security experiment where $\mathcal{A}$ additionally provides a secret key $\mathsf{sk}^{(i)}$ along with its corresponding $\mathsf{vk}^{(i)}$ at the time of each oracle query. Moreover, $\mathcal{A}$ provides a secret key sk along with vk when declaring the challenge presignature-nonce pairs. We say $\mathcal{A}$ is admissible if $\mathsf{sk}^{(i)}$ (sk) is a valid secret key for $\mathsf{vk}^{(i)}$ (vk, respectively). Note that admissibility can be checked efficiently as the secret key $\mathsf{sk}^{(i)}$ can be regarded as the random coins of the setup algorithm.

A NIBS scheme $\mathcal{S}$ satisfies *strong* receiver blindness *in the KOSK model* if no admissible PPT adversary $\mathcal{A}$ wins in the above adapted security experiment with non-negligible probability.

Definition 5.2 (strong nonce blindness in KOSK)**.** A NIBS scheme $\mathcal{S}$ satisfies *strong* nonce blindness *in the KOSK model* if no admissible PPT adversary $\mathcal{A}$ wins in the adapted security experiment similar to in Definition 5.1 with non-negligible probability. Briefly, the adaptation is that the adversary provides a valid secret key associated with each verification key it outputs.

Definition 5.3 (One-more unforgeability in KOSK)**.** A NIBS scheme $\mathcal{S}$ satisfies one-more unforgeability *in the KOSK model* if no admissible PPT adversary $\mathcal{A}$ wins in the adapted security experiment similar to in Definition 5.1 with non-negligible probability. Briefly, the adaptation is that the adversary provides a valid secret key associated with each receiver key it outputs.

5.2 Construction

We now describe our NIBS scheme from circuit-private LHE. The resulting construction has optimal-size signatures and strong security.

Tools required. The construction relies on a pseudorandom function F, a signature scheme $\mathsf{S} = (\mathsf{S.Setup}, \mathsf{S.Sign}, \mathsf{S.Verify})$, a leveled homomorphic encryption scheme $\mathcal{LHE} = (\mathsf{LHE.Setup}, \mathsf{LHE.Enc}, \mathsf{LHE.Eval}, \mathsf{LHE.Dec})$ and a NIZKAoK proof system $\mathsf{NIZK} = (\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ for the following language:

Language $\mathcal{L}_1$

Instance: Each instance x is interpreted as an LHE evaluation key $\mathsf{lhe.ek}$, LHE ciphertexts ct and $\widehat{\mathsf{ct}}$ and randomness $r \in \{0,1\}^\lambda$.

Witness: Witness ω consists of a secret signing key sk and randomness ρ.

Membership: Let $C_{\mathsf{sk},r}$ encode the circuit $\mathsf{S.Sign}(\mathsf{sk}, F_{\cdot}(r))$ for a public pseudorandom function F. Then ω is a valid witness for x if the following is satisfied:

- $\widehat{\mathsf{ct}}$ is the homomorphic evaluation of the ciphertext ct over the circuit $C_{\mathsf{sk},r}$, ie., $\widehat{\mathsf{ct}} = \mathsf{LHE.Eval}(\mathsf{lhe.ek}, C_{\mathsf{sk},r}, \mathsf{ct}; \rho)$.

Below we describe our NIBS from circuit private LHE.

$\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$. It runs the setup algorithms for NIZK (for language $\mathcal{L}_1$):

$$\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda),$$

and outputs $\mathsf{pp} = \mathsf{nizk.crs}$.

$\mathsf{KeyGen}_S(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{vk})$. The signer's setup algorithm runs the setup algorithm for the signature scheme S. Namely, it generates keys as $(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\ \mathsf{S.Setup}(1^\lambda)$.

$\mathsf{KeyGen}_R(\mathsf{pp}) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. The receiver's setup algorithm first samples a LHE key pair $(\mathsf{lhe.sk}, \mathsf{lhe.ek}) \leftarrow\!\!\$\ \mathsf{LHE.Setup}(1^\lambda, 1^d)$, and random PRF key $K \leftarrow\!\!\$\ \{0,1\}^\lambda$. (Here depth d is defined during the issue algorithm.) Next, it encrypts K as $\mathsf{ct} \leftarrow\!\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.sk}, K)$. Finally, it outputs receiver's secret key and public key as $\mathsf{sk}_R := (\mathsf{lhe.sk}, K)$ and $\mathsf{pk}_R := (\mathsf{lhe.ek}, \mathsf{ct})$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \to (\mathsf{psig}, \mathsf{nonce})$. The issue algorithm first samples a random message $r \leftarrow\!\!\$\ \{0,1\}^\lambda$. Let $C_{\mathsf{sk},r}(\cdot)$ be the following circuit (with key sk and message r hardwired) — $C_{\mathsf{sk},r}(K) \stackrel{\text{def}}{=} \mathsf{S.Sign}(\mathsf{sk}, F_K(r))$. That is, $C_{\mathsf{sk},r}$ runs the PRF function using the circuit input as the PRF key on message r, and then runs the signing algorithm to sign the output of the PRF. Let d denote the depth of the circuit $C_{\mathsf{sk},r}$. (This is the depth we set during the setup of the LHE scheme.)

The algorithm runs the homomorphic evaluation algorithm under uniformly random $\rho \leftarrow\!\!\$\ \{0,1\}^\lambda$[13], and $(\mathsf{lhe.ek}, \mathsf{ct}) := \mathsf{pk}_R$, as follows:

$$\widehat{\mathsf{ct}} \leftarrow \mathsf{LHE.Eval}(\mathsf{lhe.ek}, C_{\mathsf{sk},r}, \mathsf{ct}; \rho),$$

and creates a NIZK proof π for the language $\mathcal{L}_1$ as

$$\pi \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x = (\mathsf{pk}_R, \widehat{\mathsf{ct}}, r), \omega = (\mathsf{sk}, \rho))$$

Finally, it outputs the presignature $\mathsf{psig} := (\widehat{\mathsf{ct}}, \pi)$ and nonce $\mathsf{nonce} := r$.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, \mathsf{psig}, \mathsf{nonce}) \to (\mu, \sigma)$. The receiver parses $(\widehat{\mathsf{ct}}, \pi) := \mathsf{psig}$ and runs the NIZK verifier as $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x = (\mathsf{pk}_R, \widehat{\mathsf{ct}}, \mathsf{nonce}), \pi)$ and aborts if it outputs 0. Otherwise it continues by computing the message as $\mu = F_K(r)$, where $(\mathsf{lhe.sk}, K) := \mathsf{sk}_R$, $\widehat{\mathsf{ct}} := \mathsf{psig}$ and $\mathsf{nonce} :=$

[13] This randomness is needed for circuit privacy.

r. It then runs LHE decryption algorithm to compute the signature $\sigma \leftarrow\!\!\$\ \mathsf{LHE.Dec}(\mathsf{lhe.sk}, \widehat{\mathsf{ct}})$, and it runs the signature verification algorithm to check that σ is a valid signature for μ under vk. That is, $\mathsf{S.Verify}(\mathsf{vk}, \mu, \sigma) = 1$. If the check fails, then it aborts. Otherwise, it outputs μ and σ as the corresponding message-signature pair.

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0,1\}$. The verification algorithm runs the signature scheme verifier and outputs whether $\mathsf{S.Verify}(\mathsf{vk}, \mu, \sigma) = 1$.

Completeness. We have by definition, that circuit $C_{\mathsf{sk},r}(\cdot)$ is a signature on the PRF evaluation of F under key K. Thus, by correctness of the signature scheme and the PRF, $C_{\mathsf{sk},r}(K)$, computes the signature on the $F_K(r)$ and, further, by completeness of the NIZK proof system, $\widehat{\mathsf{ct}}$ is the encryption of the (homomorphically evaluated) signature on $F_K(r)$ under sk, for uniformly random r. Finally, by correctness of LHE, $\widehat{\mathsf{ct}}$ correctly decrypts to a signature σ such that $\mathsf{S.Verify}(\mathsf{vk}, F_K(r), \sigma)$ accepts.

Reusability. Notice that for $b \in \{0,1\}$, if a signer issues presignature-nonce pairs $((\widehat{\mathsf{ct}}_b, \pi_b), r_b)$ to a given receiver with a secret PRF key K then,

$$\begin{aligned}
\Pr[r_0 = r_0 \vee \mu_0 = \mu_1] = \;& \Pr[\mu_0 = \mu_1] + \Pr[r_0 = r_1] - \Pr[\mu_0 = \mu_1 \mid r_0 = r_1] \cdot \Pr[r_0 = r_1] \\
= \;& \Pr[F_K(r_0) = F_K(r_1)] + \Pr[r_0 = r_1] - \Pr[F_K(r_0) = F_K(r_1) \mid r_0 = r_1] \cdot \Pr[r_0 = r_1] \\
= \;& \Pr[F_K(r_0) = F_K(r_1)] + \Pr[r_0 = r_1] - \Pr[r_0 = r_1] = \Pr[F_K(r_0) = F_K(r_1)] \ .
\end{aligned}$$

The probability that for uniformly sampled values $r_0, r_1 \in \{0,1\}^\lambda$, we have that $r_0 = r_1$ is negligible in λ. Additionally, if $r_0 \neq r_1$, then the probability that $F_K(r_0) = F_K(r_1)$ also negligibly small for a secure PRF F, and hence also the probability above is overall negligible. So the construction is reusable.

5.3 One-more unforgeability

We first consider the one-more unforgeability of this protocol.

Theorem 5.4. If NIZK satisfies zero knowledge, $\mathcal{LHE}$ is a circuit private LHE and S is a secure signature scheme, then Construction 5.2 is one-more unforgeable NIBS protocol in the KOSK model.

Proof. We define the following hybrids:

Hybrid_0 This corresponds to the real one-more unforgeability experiment in the KOSK model.

1. After generating the public parameters pp of the protocol, the challenger runs the setup algorithm for the signature scheme S, generates keys as $(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\ \mathsf{S.Setup}(1^\lambda)$ and sends vk to the adversary.

2. In the i^{th} query, the adversary chooses a valid $\mathsf{pk}_{R_i}, \mathsf{sk}_{R_i}$ pair and requests a presignature using it. For each such query, the challenger replies with a presignature psig_i that it computes as follows:

   (a) It parses pk_{R_i} as $(\mathsf{lhe.ek}_i, \mathsf{ct}_i)$ and sk_{R_i} as $(\mathsf{lhe.sk}_i, K_i)$.

(b) It checks whether $K_i = \mathsf{LHE.Dec}(\mathsf{lhe.sk}_i, \mathsf{ct}_i)$. If the check fails, the challenger aborts and continues otherwise.[14]

(c) It samples a random $r_i \leftarrow\!\$\ \{0,1\}^\lambda$.

(d) Then, for circuit $C_{sk,r_i}(K_i) \stackrel{\text{def}}{=} \mathsf{S.Sign}(\mathsf{sk}, F_{K_i}(r_i))$, the challenger runs the homomorphic evaluation algorithm with randomness ρ_i to obtain $\widehat{\mathsf{ct}}_i$ as

$$\widehat{\mathsf{ct}}_i \leftarrow\!\$\ \mathsf{LHE.Eval}(\mathsf{lhe.ek}_i, C_{sk,r_i}, \mathsf{ct}_i; \rho_i) \quad .$$

(e) It creates a NIZK proof for the language $\mathcal{L}_1$ as

$$\pi_i \leftarrow\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_i = (\mathsf{pk}_{R_i}\, \widehat{\mathsf{ct}}_i, r_i), \omega_i = (\mathsf{sk}, C_{\mathsf{sk},r_i}, \rho_i))$$

(f) Sets $\mathsf{psig}_i := (\widehat{\mathsf{ct}}_i, \pi_i)$ and $\mathsf{nonce} := r_i$.

The adversary repeats this step for a total of ℓ queries.

3. Adversary outputs k message and signature pairs $((\mu_1, \sigma_1), \ldots, (\mu_k, \sigma_k))$. Adversary wins if $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, $\mathsf{Verify}(\mathsf{vk}, \mu_i, \sigma_i) = 1$ for all $i \in [k]$, and $k > \ell$.

Hybrid_i For $i \in [\ell]$ each Hybrid_i is the same as Hybrid_{i-1}, except that on i^{th} presignature query, instead of honestly generating the NIZK proof π_i, the challenger simulates it without any witness.

2. (e) It sets statement x_i to be $(\mathsf{pk}_{R_i}\, \widehat{\mathsf{ct}}_i, r_i)$ and, without setting any witness, generates π_i using NIZK simulator.

$\mathsf{Hybrid}_{\ell+i}$ For $i \in [\ell]$ each $\mathsf{Hybrid}_{\ell+i}$ is the same as $\mathsf{Hybrid}_{\ell+i-1}$, except that on the i^{th} presignature query, instead of homomorphically evaluating the circuit, it simulates the ciphertext generation using the LHE simulator.

2. (c) Then, for circuit $C_{sk,r_i}(K_i) \stackrel{\text{def}}{=} \mathsf{S.Sign}(\mathsf{sk}, F_{K_i}(r_i))$, the challenger computes $\sigma_i \leftarrow\!\$\ \mathsf{S.Sign}(\mathsf{sk}, F_{K_i}(r_i))$. It then runs the LHE simulator as $\widehat{\mathsf{ct}}_i \leftarrow\!\$\ \mathsf{Sim}(\mathsf{lhe.ek}_i, \sigma_i)$.

Let $\mathsf{Adv}^j_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in the one-more unforgeability game in Hybrid_j. Then, the following must hold:

Lemma 5.5. Assuming zero-knowledge property of NIZK, it holds that for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^{i-1}_{\mathcal{A}} - \mathsf{Adv}^{i}_{\mathcal{A}}| = \mathsf{negl}(\lambda)$ for $i \in [\ell]$.

Proof. For any $i \in [\ell]$, consider Hybrid_i in which, the challenger, responds to the i^{th} presignature query by simulating the NIZK proof according to the statement $x_i = (\mathsf{pk}_{R_i}\, \widehat{\mathsf{ct}}_i, r_i)$. Concretely, let $\mathcal{A}$ be a PPT attacker such that $|\mathsf{Adv}^{i-1}_{\mathcal{A}} - \mathsf{Adv}^{i}_{\mathcal{A}}|$ is non-negligible, we design a reduction algorithm $\mathcal{B}$ that breaks the zero-knowledge property of NIZK with non-negligible advantage.

The NIZK challenger starts by sampling $b^* \leftarrow\!\$\ \{0,1\}$. If $b^* = 0$, it sets $\mathsf{nizk.crs} \leftarrow\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$. Otherwise it runs the simulator $\mathsf{nizk.crs} \leftarrow\!\$\ \mathcal{S}(1^\lambda)$. It sends nizk.crs to the reduction algorithm

[14] Note that this can be done without loss of generality in the KOSK model as this does not affect any change to the adversary's point of view.

$\mathcal{B}$, which in turn forwards it to $\mathcal{A}$ as pp. After generating the public parameters of the protocol, $\mathcal{B}$ runs the setup algorithm for the signature scheme S to generate the key pair $(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\ \mathsf{S.Setup}(1^\lambda)$. It sends vk to $\mathcal{A}$, who is then allowed to make a series of presignature queries. To answer the i^{th} query, $\mathcal{B}$ computes $x_i = (\mathsf{pk}_{R_i}, \widehat{\mathsf{ct}}_i, r_i), \omega_i = (\mathsf{sk}, C_{\mathsf{sk},r_i}$ as the signer would, and queries the NIZK challenger with (x_i, ω_i) who replies with a proof π_i. $\mathcal{B}$ then sets $\mathsf{psig}_i := (\widehat{\mathsf{ct}}_i, \pi_i)$ and $\mathsf{nonce} := r_i$, and forwards it to $\mathcal{A}$. The attacker $\mathcal{A}$ outputs k message signature pairs $((\mu_1, \sigma_1), \ldots, (\mu_k, \sigma_k))$. If $\mu_{i'} \neq \mu_j$ for $1 \leq i' < j \leq k$, $\mathsf{Verify}(\mathsf{vk}, \mu_{i'}, \sigma_{i'}) = 1$ for $1 \leq i' \leq k$, and $k > \ell$, then $\mathcal{B}$ sends 0 as its guess (i.e., π_i is a simulated proof), otherwise it sends 1 as its guess (i.e., π_i is honestly generated from $\mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_i, \omega_i)$).

Note that if the NIZK challenger honestly generates proof the π_i using $\mathsf{NIZK.Prove}\ (\mathsf{nizk.crs}, x_i, \omega_i)$, then $\mathcal{B}$ perfectly simulates the experiment of Hybrid_{i-1} for adversary $\mathcal{A}$. Otherwise it simulates the experiment of Hybrid_i. As a result, if $|\mathsf{Adv}^{i-1}_{\mathcal{A}} - \mathsf{Adv}^{i}_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks the zero-knowledge property of NIZK with non-negligible advantage. Thus the lemma holds for i which is general, so it holds for all $i \in [\ell]$. □

Lemma 5.6. If $\mathcal{LHE}$ is a circuit private LHE, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^{\ell+i-1}_{\mathcal{A}} - \mathsf{Adv}^{\ell+i}_{\mathcal{A}}| = \mathsf{negl}(\lambda)$ for $j \in [\ell]$.

Proof. For any $i \in [\ell]$, consider the hybrid $\mathsf{Hybrid}_{\ell+i}$ in which, the challenger, responds to the i^{th} presignature query by simulating the ciphertext generation using the LHE simulator instead of homomorphically evaluating the circuit. Concretely, for circuit $C_{sk,r_i}(K_i) \stackrel{\text{def}}{=} \mathsf{S.Sign}(\mathsf{sk}, F_{K_i}(r_i))$, the challenger leverages its knowledge of the secret key to compute $\sigma_i \leftarrow\!\!\$\ \mathsf{S.Sign}(\mathsf{sk}, F_{K_i}(r_i))$. It then runs the LHE simulator as $\widehat{\mathsf{ct}}_i \leftarrow\!\!\$\ \mathsf{Sim}(\mathsf{lhe.ek}_i, \sigma_i)$. By circuit privacy of LHE, we have that for such a PPT algorithm Sim, in particular, the following holds:

$$\left(\mathsf{lhe.ek}, \mathsf{LHE.Eval}(\mathsf{lhe.ek}, C_{\mathsf{sk},r_i}, \mathsf{ct}_i), \mathsf{ct}_i\right) \approx_c \left(\mathsf{lhe.ek}, \mathsf{Sim}(\mathsf{lhe.ek}, C_{\mathsf{sk},r_i}(K_i)), \mathsf{ct}_i\right)$$

It further follows by construction that σ_i is a valid signature on $F_{K_i}(r_i)$. So, $C_{\mathsf{sk},r_i}(K_i) = \sigma_i$ and thus, a PPT adversary has negligible advantage in distinguishing $\widehat{\mathsf{ct}}_i$ produced as the output of $\mathsf{LHE.Eval}(\mathsf{lhe.ek}, C_{\mathsf{sk},r_i}, \mathsf{ct}_i)$ and that produced by the LHE Simulator $\mathsf{Sim}(\mathsf{lhe.ek}, \sigma_i)$. Thus the lemma holds for i which is general, so it holds for all $i \in [\ell]$. □

Lemma 5.7. Assuming the signature scheme S is secure under existential unforgeability (Definition 3.14), adversary $\mathcal{A}$ can have at most negligible advantage in $\mathsf{Hybrid}_{2\ell}$.

Proof. We reduce to the unforgeability of the underlying signature scheme. Suppose there exists a PPT attacker $\mathcal{A}$ that wins the one-more unforgeability game with non-negligible probability $\epsilon = \epsilon(\lambda)$. It outputs k message and signature pairs $((\mu_1, \sigma_1), \ldots, (\mu_k, \sigma_k))$, and wins if and only if $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, $\mathsf{Verify}(\mathsf{vk}, \mu_i, \sigma_i) = 1$ for $1 \leq i \leq k$, and $k > \ell$. Also recall that in the KOSK model, the attacker must output a valid pk_R, sk_R such that $K_i = \mathsf{LHE.Dec}(\mathsf{lhe.sk}_i, \mathsf{ct})$ and $\mathsf{ct}_i = \mathsf{LHE.Enc}(\mathsf{lhe.sk}_i, K_i)$. It then follows by correctness of the underlying scheme that any ciphertext $\widehat{\mathsf{ct}}_i$ must decrypt uniquely under lhe.sk.

Consider, then, a reduction algorithm $\mathcal{B}$ that, given access to $\mathcal{A}$, breaks the security of signature scheme S. To begin, the Unf challenger generates a $(\mathsf{sk}, \mathsf{vk})$ pair using its S.Setup algorithm, and sends vk to $\mathcal{B}$ who then forwards it to $\mathcal{A}$. For every presignature query it receives for R_i, $\mathcal{B}$ ensures admissibility by itself computing the key-pair $(\mathsf{pk}_{R_i}, \mathsf{sk}_{R_i})$ (recall in KOSK model, the secret key can be viewed as random coins of the setup algorithm), and aborts if $\mathcal{A}$ is inadmissible. Otherwise,

after parsing the key-pair as $\mathsf{lhe.ek}_i, \mathsf{ct}_i$ and $\mathsf{lhe.sk}_i, K_i$ respectively, it samples a value $r_i \leftarrow\!\!\$\ \{0,1\}^\lambda$ uniformly at random, computes $\mu_i = F_{K_i}(r_i)$. The reduction algorithm, $\mathcal{B}$, then sends μ_i as a signing query to the challenger and receives a signature σ_i in response. It computes $\widehat{\mathsf{ct}}_i \leftarrow\!\!\$\ \mathsf{Sim}(\mathsf{lhe.ek}, \sigma_i)$ and sends $(\widehat{\mathsf{ct}}_i, r_i)$ as the i^{th} presignature to $\mathcal{A}$. After ℓ such queries, $\mathcal{A}$ outputs $k > \ell$ forgeries $\{\mu_i^*, \sigma_i^*\}_{i=1}^k$ such that for all $i \in k$, $\mathsf{Verify}(\mathsf{vk}, \mu_i^*, \sigma_i^*) = 1$ and μ_i^* are all different.

Since $\mathcal{A}$ makes ℓ presignature queries, as does the reduction algorithm make ℓ signature queries to the challenger. However, all the $k > \ell$ messages μ_i^* are different. So it follows by a simple pigeonhole argument that at least one of them was never queried by the attacker (and thus by the reduction). Let μ^* be such a message, and σ^* its corresponding signature. The reduction algorithm, $\mathcal{B}$, then simply gives (μ^*, σ^*) as a forgery to the challenger, thus breaking the security of signature scheme S. □

This completes the proof for one-more unforgeability of Construction 5.2 in the KOSK model. □

5.4 Strong receiver blindness

Theorem 5.8. If NIZK is a NIZKAoK, $\mathcal{LHE}$ is a IND-CPA secure encryption scheme and F is a secure pseudorandom function, then Construction 5.2 is a strong receiver-blind NIBS protocol.

Proof. We define the following hybrids:

Hybrid_0 This corresponds to the real receiver blindness experiment.

1. The challenger, after generating the public parameters pp of the protocol, runs the receiver key generation algorithm for each $b \in \{0,1\}$ as follows:
   (a) It samples an LHE key pair $(\mathsf{lhe.sk}_b, \mathsf{lhe.ek}_b) \leftarrow\!\!\$\ \mathsf{LHE.Setup}(1^\lambda, 1^d)$ and random PRF key $K_b \leftarrow\!\!\$\ \{0,1\}^\lambda$.
   (b) It encrypts K_b as $\mathsf{ct}_b \leftarrow\!\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.sk}_b, K_b)$, and sets receiver secret and public keys as $\mathsf{sk}_{R_b} := (\mathsf{lhe.sk}_b, K_b)$ and $\mathsf{pk}_{R_b} := (\mathsf{lhe.ek}_b, \mathsf{ct}_b)$.

   It then sends pk_{R_0} and pk_{R_1} to the attacker.
2. At any point in the protocol, the adversary is allowed to make a series of $\kappa = \kappa(\lambda)$ queries for κ polynomially bounded in λ. For any $i \in [\kappa]$, the i^{th} query is explained as follows:
   (a) In the i^{th} query, the adversary chooses a bit $b^{(i)}$, a verification key $\mathsf{vk}^{(i)}$, a presignature $(\widehat{\mathsf{ct}}^{(i)}, \pi^{(i)})$ and a nonce $r^{(i)}$.
   (b) The challenger runs the NIZK verifier as $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x^{(i)} = (pk_{R_{b^{(i)}}}, \widehat{\mathsf{ct}}^{(i)}, r^{(i)}), \pi^{(i)})$ and aborts if it outputs 0.
   (c) Continuing otherwise, the challenger computes the message as $\mu^{(i)} = F_{K_{b^{(i)}}}(r^{(i)})$.
   (d) It then runs LHE decryption algorithm to compute the signature $\sigma^{(i)} \leftarrow \mathsf{LHE.Dec}(\mathsf{lhe.sk}_{b^{(i)}}, \widehat{\mathsf{ct}}^{(i)})$.
   (e) Finally it calls the signature verification algorithm $\mathsf{S.Verify}(\mathsf{vk}^{(i)}, \mu^{(i)}, \sigma^{(i)})$. If the output is 0, the check fails and it aborts. It continues otherwise.

   The challenger then sends the message-signature pair $\mu^{(i)}$ and $\sigma^{(i)}$ to the attacker.

3. For attacker's signature query $(\mathsf{vk}, ((\widehat{\mathsf{ct}}_0, \pi_0), r_0), ((\widehat{\mathsf{ct}}_1, \pi_1), r_1))$, the challenger computes two message-signature pairs, one for each $b \in \{0,1\}$, as follows:

   (a) It runs the NIZK verifier as $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x_b = (pk_{R_b}, \widehat{\mathsf{ct}}_b, r_b), \pi_b)$ and aborts if it outputs 0.

   (b) It computes the message as $\mu_b = F_{K_b}(r_b)$.

   (c) It then runs LHE decryption algorithm to compute the signature $\sigma_b \leftarrow \mathsf{LHE.Dec}(\mathsf{lhe.sk}_b, \widehat{\mathsf{ct}}_b)$

   (d) Finally it calls the signature verification algorithm $\mathsf{S.Verify}(\mathsf{vk}, \mu_b, \sigma_b)$. If the output is 0, the check fails and it aborts. It continues otherwise.

4. The challenger samples a bit $\hat{b} \leftarrow\$ \{0,1\}$ and sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to the attacker.

5. The attacker, who continues to have Obtain oracle access, outputs a bit b'.

An admissible adversary $\mathcal{A}$ wins if $b' = \hat{b}$.

Hybrid_1 This is the same as Hybrid_0, except that the challenger uses a uniformly sampled value $\tau \leftarrow\$ \{0,1\}^\lambda$ to generate nizk.crs.

1. The challenger samples $\tau \leftarrow\$ \{0,1\}^\lambda$, generates $\mathsf{nizk.crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda; \tau)$ and sets $\mathsf{pp} = \mathsf{nizk.crs}$. It then runs the receiver key generation algorithm for each $b \in \{0,1\}$ (same as before).

Hybrid_2 This is the same as Hybrid_1, except that for every Obtain query, instead of running the LHE decryption algorithm to obtain the signature, it runs the NIZKAoK extractor to extract the secret signing key (as part of the witness) and uses the signing algorithm under $\mathsf{sk}^{(i)}$ to obtain the signature on message $\mu^{(i)} = F_{K_{b^{(i)}}}(r^{(i)})$ for all $i \in [\kappa]$, i.e.,

2. (d) Challenger extracts $(\mathsf{sk}^{(i)}, \rho^{(i)}) := \mathcal{E}(\tau, x^{(i)} := (\mathsf{pk}_{R_{b^{(i)}}}, \widehat{\mathsf{ct}}^{(i)}, r^{(i)}), \pi^{(i)})$, and uses it to compute the signature $\sigma^{(i)} \leftarrow\$ \mathsf{S.Sign}(\mathsf{sk}^{(i)}, \mu^{(i)})$.

Hybrid_3 This is the same as Hybrid_2, except instead of running the LHE decryption algorithm to obtain the signature, it runs the NIZKAoK extractor to extract the secret signing key (as part of the witness) and uses the signing algorithm under sk to obtain the signature on message μ_b for each $b \in \{0,1\}$.

3. (c) Challenger extracts $(\mathsf{sk}, \rho_b) := \mathcal{E}(\tau, x_b := (\mathsf{pk}_{R_b}, \widehat{\mathsf{ct}}_b, r_b), \pi_b)$ and uses it to compute the signature $\sigma_b \leftarrow\$ \mathsf{S.Sign}(\mathsf{sk}, \mu_b)$.

Hybrid_4 This is the same as Hybrid_3, except that for each $b \in \{0,1\}$, instead of generating $\mathsf{ct}_b \leftarrow\$ \mathsf{LHE.Enc}(\mathsf{lhe.sk}_b, K_b)$, the challenger sets ct_b as $\mathsf{LHE.Enc}(\mathsf{lhe.sk}_b, 0^\lambda)$.

1. (b) It computes $\mathsf{ct}_b \leftarrow\$ \mathsf{LHE.Enc}(\mathsf{lhe.sk}_b, 0^\lambda)$ and sets receiver secret and public keys as $\mathsf{sk}_{R_b} := (\mathsf{lhe.sk}_b, K_b)$ and $\mathsf{pk}_{R_b} := (\mathsf{lhe.ek}_b, \mathsf{ct}_b)$.

Hybrid_5 This is the same as Hybrid_4, except instead of computing message μ_b as the PRF evaluation of r_b under K_b, the challenger samples it uniformly at random. for each $b \in \{0,1\}$.

3. (b) It samples message $\mu_b \leftarrow\!\$\, \{0,1\}^\lambda$ uniformly at random.

Let $\mathsf{Adv}^j_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in the strong receiver blindness game in Hybrid_j. Then, the following must hold:

Lemma 5.9. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^0_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| = 0$.

Proof. Notice that the only difference between Hybrid_0 and Hybrid_1 is that the randomness τ of the CRS generation algorithm is chosen by the challenger. Since τ is chosen uniformly by the challenger, this affects no change from the adversaries perspective. □

Lemma 5.10. If NIZK is a NIZKAoK, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. Suppose for contradiction that $\exists i \in [\kappa]$ for which the lemma does not hold. Then one of two cases must hold during Hybrid_2 against $\mathcal{A}$:

- Case 1: $\omega^{(i)} := (\mathsf{sk}^{(i)}, \rho^{(i)})$ is not a valid witness for $\pi^{(i)}$ with respect to the instance $x^{(i)}$.
- Case 2: $\omega^{(i)}$ is a valid witness for $\pi^{(i)}$ with respect to the instance $x^{(i)}$.

In the former case we have that $\omega^{(i)}$ is not a valid witness, but also due to the previous step $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x^{(i)}, \pi^{(i)}) = 1$ since otherwise the challenger would have aborted the protocol. By definition of knowledge extraction, we know the probability of this event to be negligible. In the latter case it must have been that $(\mathsf{vk}^{(i)}, \mathsf{sk}^{(i)})$ was an invalid signature key pair, in which case $\mathsf{S.Verify}(\mathsf{vk}^{(i)}, F_{K_{b^{(i)}}}(r^{(i)}), \sigma^{(i)}) \neq 1$ and the protocol is aborted. Thus the lemma holds for i, which is general, so it holds for all $i \in [\kappa]$.

□

Lemma 5.11. If NIZK is a NIZKAoK, for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. Let us consider an intermediate step between Hybrid_2 and Hybrid_3. In the intermediate step, everything remains the same in Hybrid_2, except for the following:

3. (c) Challenger extracts $(\mathsf{sk}, \rho_0) := \mathcal{E}(\tau, x_0 := (\mathsf{pk}_{R_0}, \widehat{\mathsf{ct}}_0, r_0), \pi_0)$ and uses it to compute the signature $\sigma_0 \leftarrow\!\$\, \mathsf{S.Sign}(\mathsf{sk}, \mu_0)$. It also computes $\sigma_1 \leftarrow\!\$\, \mathsf{LHE.Dec}(\mathsf{lhe.sk}_1, \widehat{\mathsf{ct}}_1)$.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{2.5}_{\mathcal{A}}$. Proceeding exactly as in the previous proof, suppose for contradiction that the adversary wins with non-negligible probability $\epsilon = \epsilon(\lambda)$. Then one of two cases must hold in the intermediate experiment against $\mathcal{A}$:

- Case 1: $\omega_0 := (\mathsf{sk}, \rho_0)$ is not a valid witness for π_0 with respect to the instance x_0.
- Case 2: ω_0 is a valid witness for π_0 with respect to the instance x_0.

In the former case we have that ω_0 is not a valid witness, but also due to the previous step $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x_0, \pi_0) = 1$ since otherwise the challenger would have aborted the protocol. By definition of knowledge extraction, we know the probability of this event to be negligible. In the latter case it must have been that $(\mathsf{vk}, \mathsf{sk})$ was an invalid signature key pair, in which case $\mathsf{S.Verify}(\mathsf{vk}, F_{K_0}(r_0), \sigma_0) \neq 1$ and the protocol is aborted. Therefore, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^{2.5}_{\mathcal{A}}| = \mathsf{negl}(\lambda)$. By a similar argument we can show that $|\mathsf{Adv}^{2.5}_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| = \mathsf{negl}(\lambda)$ and thus the lemma follows. □

Lemma 5.12. If $\mathcal{LHE}$ is an IND-CPA secure encryption scheme, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. Let us consider an intermediate step between Hybrid_2 and Hybrid_3. In the intermediate step, everything remains the same in Hybrid_3, except for the following:

1. (b) The challenger computes $\mathsf{ct}_0 \leftarrow\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.sk}_0, 0^\lambda)$ and $\mathsf{ct}_1 \leftarrow\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.sk}_1, K_1)$, and sets receiver secret and public keys as $\mathsf{sk}_{R_b} := (\mathsf{lhe.sk}_b, K_b)$ and $\mathsf{pk}_{R_b} := (\mathsf{lhe.ek}_b, \mathsf{ct}_b)$.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{2.5}_{\mathcal{A}}$, and suppose there exists such a PPT attacker $\mathcal{A}$, such that $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^{3.5}_{\mathcal{A}}| = \epsilon(\lambda)$. We design a reduction algorithm $\mathcal{B}$ that breaks the security of LHE with advantage $\epsilon(\lambda)$.

The IND-CPA challenger first samples a secret key and evaluation key pair $(\mathsf{lhe.sk}_0, \mathsf{lhe.ek}_0)$ using LHE.Setup and sends $\mathsf{lhe.ek}_0$ to the reduction algorithm $\mathcal{B}$. The reduction algorithm then generates the public parameters of the protocol pp and then samples an LHE key pair $(\mathsf{lhe.sk}_1, \mathsf{lhe.ek}_1) \leftarrow\!\$\ \mathsf{LHE.Setup}(1^\lambda, 1^d)$ and random PRF key $K_b \leftarrow\!\$\ \{0,1\}^\lambda$ for each $b \in \{0,1\}$. It sends 0^λ and K_0 to the challenger and receives a ciphertext ct_0 as the encryption of either 0^λ or K_0. It computes $\mathsf{ct}_1 \leftarrow\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.sk}_1, K_1)$, and sets receiver secret key as $\mathsf{sk}_{R_1} := (\mathsf{lhe.sk}_1, K_1)$ and the public keys as $\mathsf{pk}_{R_b} := (\mathsf{lhe.ek}_b, \mathsf{ct}_b)$. The reduction algorithm then forwards pk_{R_0} and pk_{R_1} to $\mathcal{A}$. The rest of the protocol proceeds identically to Hybrid_3. At the end, $\mathcal{B}$ samples $\hat{b} \leftarrow\!\$\ \{0,1\}$ uniformly at random and sends $((\mu_{\hat{b}}, \sigma_{\hat{b}}), (\mu_{1-\hat{b}}, \sigma_{1-\hat{b}}))$ to $\mathcal{A}$ who outputs its guess b'. If $b' = \hat{b}$, $\mathcal{B}$ sends 0 as its guess (i.e., ct_0 is an encryption of 0^λ), otherwise it sends 1 as its guess (i.e., ct_0 is an encryption of K_0).

Note that if ct_0 is an encryption of string K_0, then $\mathcal{B}$ perfectly simulates the experiment of Hybrid_3 for adversary $\mathcal{A}$. Otherwise it simulates the intermediate step. As a result, if $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^{3.5}_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks the IND-CPA security LHE with non-negligible advantage. Similarly, we have $|\mathsf{Adv}^{3.5}_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}| \le \mathsf{negl}(\lambda)$ and thus $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}| \le \mathsf{negl}(\lambda)$ □

Lemma 5.13. If F is a pseudorandom function, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^4_{\mathcal{A}} - \mathsf{Adv}^5_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. Let us consider an intermediate step between Hybrid_5 and Hybrid_6. In the intermediate step, everything remains the same in Hybrid_5, except for the following:

3. (a) Challenger samples message $\mu_0 \leftarrow\!\$\ \{0,1\}^\lambda$ uniformly at random, and evaluates $\mu_1 = F_{K_1}(r_1)$.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{4.5}_{\mathcal{A}}$. Suppose there exists such a PPT attacker $\mathcal{A}$, such that $|\mathsf{Adv}^4_{\mathcal{A}} - \mathsf{Adv}^{4.5}_{\mathcal{A}}| = \epsilon(\lambda)$. We design a reduction algorithm $\mathcal{B}$ that breaks the pseudorandomness property of F with advantage $\epsilon(\lambda)$.

The reduction algorithm $\mathcal{B}$ generates the public parameters of the protocol pp and then samples an LHE key pair $(\mathsf{lhe.sk}_b, \mathsf{lhe.ek}_b) \leftarrow\!\$\ \mathsf{LHE.Setup}(1^\lambda, 1^d)$ and random PRF key $K_b \leftarrow\!\$\ \{0,1\}^\lambda$ for each $b \in \{0,1\}$. It computes $\mathsf{ct}_b \leftarrow\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.ek}_1, 0^\lambda)$, and sets receiver secret and public keys as $\mathsf{sk}_{R_b} := (\mathsf{lhe.sk}_b, K_b)$ and $\mathsf{pk}_{R_b} := (\mathsf{lhe.ek}_b, \mathsf{ct}_b)$. $\mathcal{B}$ then forwards pk_{R_0} and pk_{R_1} to $\mathcal{A}$. For each $i \in [\kappa]$, let $(b^{(i)}, \mathsf{vk}^{(i)}, \mathsf{sk}^{(i)}, \widehat{\mathsf{ct}}^{(i)}, r^{(i)})$ be $\mathcal{A}$'s i^{th} Obtain query. If $b^{(i)} = 0$ (resp. 0), $\mathcal{B}$ queries the first (resp. second) PRF challenger on $r^{(i)}$, who responds with string $y_{b^{(i)}} \in \{0,1\}^\lambda$, which is either a PRF evaluation of $r^{(i)}$ or a uniformly random string, and sets $\mu^{(i)} = y_{b^{(i)}}$. When $\mathcal{A}$ outputs a signature query

$(\mathsf{vk}, \mathsf{sk}, (\widehat{\mathsf{ct}}_0, r_0), (\widehat{\mathsf{ct}}_1, r_1))$, $\mathcal{B}$ again queries the oracle for r_0, receives a response y_0 and sets $\mu_0 = y_0$. It also evaluates $\mu_1 = F_{K_1}(r_1)$. The rest of the protocol proceeds identically to Hybrid_5. At the end, $\mathcal{B}$ samples $\hat{b} \leftarrow\$ \{0,1\}$ uniformly at random and sends $((\mu_{\hat{b}}, \sigma_{\hat{b}}), (\mu_{1-\hat{b}}, \sigma_{1-\hat{b}}))$ to $\mathcal{A}$ who outputs its guess b'. If $b' = \hat{b}$, $\mathcal{B}$ sends 0 as its guess (i.e., PRF challenges are pseudorandom), otherwise it sends 1 as its guess (i.e., PRF challenges are truly random).

Recall that we require $\mathcal{A}$ to be an admissible adversary ie., in particular, $\mathcal{A}$ must not have queried r_0 or r_1 in an Obtain query. Then, if PRF challenges are pseudorandom, then $\mathcal{B}$ perfectly simulates the experiment of Hybrid_4 for adversary $\mathcal{A}$. Otherwise it simulates the intermediate step. As a result, if $|\mathsf{Adv}^4_{\mathcal{A}} - \mathsf{Adv}^{4.5}_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks the presudorandomness property of F with non-negligible advantage. Similarly, we have $|\mathsf{Adv}^{4.5}_{\mathcal{A}} - \mathsf{Adv}^5_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$ and thus $|\mathsf{Adv}^4_{\mathcal{A}} - \mathsf{Adv}^5_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$. □

Observe that the adversary has 0 advantage under Hybrid_5. From the above lemmas, it follows that Construction 5.2 satisfies strong recipient blindness. □

5.5 Strong nonce blindness

Theorem 5.14. If NIZK is a NIZKAoK, $\mathcal{LHE}$ is a IND-CPA secure encryption scheme and F is a secure pseudorandom function, then Construction 5.2 is a strong nonce blind NIBS protocol.

Proof. We define the following hybrids:

Hybrid_0 This corresponds to the real nonce blindness experiment.

1. The challenger, after generating the public parameters pp of the protocol, runs the receiver key generation algorithm as follows:
   (a) It samples an LHE key pair $(\mathsf{lhe.sk}, \mathsf{lhe.ek}) \leftarrow\$ \mathsf{LHE.Setup}(1^\lambda, 1^d)$ and random PRF key $K \leftarrow\$ \{0,1\}^\lambda$.
   (b) It encrypts K as $\mathsf{ct} \leftarrow\$ \mathsf{LHE.Enc}(\mathsf{lhe.sk}, K)$, and sets receiver secret and public keys as $\mathsf{sk}_R := (\mathsf{lhe.sk}, K)$ and $\mathsf{pk}_R := (\mathsf{lhe.ek}, \mathsf{ct})$.

   It then sends pk_R to the attacker.
2. At any point in the protocol, adversary is allowed to make a series of $\kappa = \kappa(\lambda)$ queries for κ polynomially bounded in λ. For any $i \in [\kappa]$, the i^{th} query is explained as follows:
   (a) In the i^{th} query, the adversary chooses a verification key $\mathsf{vk}^{(i)}$, a presignature $(\widehat{\mathsf{ct}}^{(i)}, \pi^{(i)})$ and a nonce $r^{(i)}$.
   (b) The challenger runs the NIZK verifier as $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x^{(i)} = (pk_R, \widehat{\mathsf{ct}}^{(i)}, r^{(i)}), \pi^{(i)})$ and aborts if it outputs 0.
   (c) The challenger computes the message as $\mu^{(i)} = F_K(r^{(i)})$.
   (d) It then runs LHE decryption algorithm to compute the signature $\sigma^{(i)} \leftarrow\$ \mathsf{LHE.Dec}(\mathsf{lhe.sk}, \widehat{\mathsf{ct}}^{(i)})$.
   (e) Finally it calls the signature verification algorithm $\mathsf{S.Verify}(\mathsf{vk}^{(i)}, \mu^{(i)}, \sigma^{(i)})$. If the output is 0, the check fails and it aborts. It continues otherwise.

   The challenger then sends the message-signature pair $\mu^{(i)}$ and $\sigma^{(i)}$ to the attacker.

3. For attacker's signature query $(\mathsf{vk}, ((\widehat{\mathsf{ct}}_0, \pi_0), r_0), ((\widehat{\mathsf{ct}}_1, \pi_1) r_1))$, the challenger computes two message-signature pairs, one for each $b \in \{0,1\}$, as follows:

   (a) It runs the NIZK verifier as $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x_b = (pk_R, \widehat{\mathsf{ct}}_b, r_b), \pi_b)$ and aborts if it outputs 0.

   (b) It computes the message as $\mu_b = F_K(r_b)$.

   (c) It then runs LHE decryption algorithm to compute the signature $\sigma_b \leftarrow\!\$\ \mathsf{LHE.Dec}(\mathsf{lhe.sk}, \widehat{\mathsf{ct}}_b)$

   (d) Finally it calls the signature verification algorithm $\mathsf{S.Verify}(\mathsf{vk}, \mu_b, \sigma_b)$. If the output is 0, the check fails and it aborts. It continues otherwise.

4. The challenger samples a bit $\hat{b} \leftarrow\!\$\ \{0,1\}$ and sends $\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}}$ to the attacker.

5. The attacker, who continues to have oracle access, outputs a bit b'.

   An admissible adversary $\mathcal{A}$ wins if $b' = \hat{b}$.

Hybrid_1 This is the same as Hybrid_0, except that the challenger provides a uniformly sampled value $\tau \leftarrow\!\$\ \{0,1\}^\lambda$ to generate nizk.crs.

1. The challenger samples $\tau \leftarrow\!\$\ \{0,1\}^\lambda$, generates $\mathsf{nizk.crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda; \tau)$ and sets $\mathsf{pp} = \mathsf{nizk.crs}$. It then runs the receiver key generation algorithm for each $b \in \{0,1\}$ (same as before).

Hybrid_2 This is the same as Hybrid_1, except that for every Obtain query, instead of running the LHE decryption algorithm to obtain the signature, it runs the NIZKAoK extractor to extract the secret signing key (as part of the witness) and uses the signing algorithm under $\mathsf{sk}^{(i)}$ to obtain the signature on message $\mu^{(i)}$ for all $i \in [\kappa]$, ie.,

2. (d) Challenger extracts $(\mathsf{sk}^{(i)}, \rho^{(i)}) := \mathcal{E}(\tau, x^{(i)} := (\mathsf{pk}_R, \widehat{\mathsf{ct}}^{(i)}, r^{(i)}), \pi^{(i)})$, and uses it to compute the signature $\sigma^{(i)} \leftarrow\!\$\ \mathsf{S.Sign}(\mathsf{sk}^{(i)}, \mu^{(i)})$.

Hybrid_3 This is the same as Hybrid_2, except instead of running the LHE decryption algorithm to obtain the signature, it runs the NIZKAoK extractor to extract the secret signing key (as part of the witness) and uses the signing algorithm under sk to obtain the signature on message μ_b for each $b \in \{0,1\}$.

3. (c) Challenger extracts $(\mathsf{sk}, \rho_b) := \mathcal{E}(\tau, x_b := (\mathsf{pk}_R, \widehat{\mathsf{ct}}_b, r_b), \pi_b)$ and uses the signature $\sigma_b \leftarrow\!\$\ \mathsf{S.Sign}(\mathsf{sk}, \mu_b)$.

Hybrid_4 This is the same as Hybrid_3, except that for each $b \in \{0,1\}$, instead of generating $\mathsf{ct}_b \leftarrow\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.sk}_b, K_b)$, the challenger sets ct_b as $\mathsf{LHE.Enc}(\mathsf{lhe.sk}_b, 0^\lambda)$.

1. (b) It computes $\mathsf{ct} \leftarrow\!\$\ \mathsf{LHE.Enc}(\mathsf{lhe.sk}, 0^\lambda)$ and sets receiver secret and public keys as $\mathsf{sk}_R := (\mathsf{lhe.sk}, K)$ and $\mathsf{pk}_R := (\mathsf{lhe.ek}, \mathsf{ct})$.

Hybrid_5 This is the same as Hybrid_4, except instead of computing message μ_b as the PRF evaluation of r_b under K, the challenger samples it uniformly at random. for each $b \in \{0,1\}$.

3. (b) It samples message $\mu_b \leftarrow\!\!\$\ \{0,1\}^\lambda$ uniformly at random.

Let $\mathsf{Adv}^j_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in the strong receiver blindness game in Hybrid_j. Then, the following must hold:

Lemma 5.15. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^0_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| = 0$.

Proof. (omitted) identical to the proof of Lemma 5.9. □

Lemma 5.16. If NIZK is a NIZKAoK, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. (omitted) identical to the proof of Lemma 5.10. □

Lemma 5.17. If NIZK is a NIZKAoK, for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. (omitted) almost identical to the proof of Lemma 5.11 without needing an intermediate step. □

Lemma 5.18. If LHE is an IND-CPA secure encryption scheme, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. (omitted) identical to the proof of Lemma 5.12. □

Lemma 5.19. If F is a pseudorandom function, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^4_{\mathcal{A}} - \mathsf{Adv}^5_{\mathcal{A}}| = \mathsf{negl}(\lambda)$.

Proof. (omitted) identical to the proof of Lemma 5.13. □

Observe that the adversary has 0 advantage under Hybrid_5. From the above lemmas, it follows that Construction 5.2 satisfies strong nonce blindness. □

6 The Randomized One-more ISIS Assumption

In this work, we introduce a new variant of the OM-ISIS assumption with two goals — (i) protect from attackers that can ask for preimage queries on the same target vector more than once, and (ii) allow for re-randomization of the target vectors before answering the preimage query phase to make the assumption more robust. As discussed in [AKSY22, § 4.5], there are polynomial time attacks on the OM-ISIS assumptions for certain parameter regimes. At its core, all cryptanalysis efforts on OM-ISIS exploit the fact that the attacker can submit any target vector of its choice as a preimage query. Thus, an attacker can potentially request for preimage queries for short vectors, and use those to create an approximate trapdoor. However, the quality of trapdoor computed this way is much worse than the actual trapdoor, thus if the parameter β is set appropriately (i.e., sufficiently small), then an attacker cannot break the assumption since the approximate trapdoor will not give as short preimage vectors.

While existing cryptanalytic efforts do not succeed in breaking the assumption for the desired parameter regimes, we view the combinatorial and lattice-based attacks provided by [AKSY22] as partial evidence of existing assumption not being as robust. Moreover, we believe that while OM-ISIS is a very natural step towards defining lattice-analogue of the one-more-RSA assumption by Bellare et al. [BNPS03], there exists more robust instantiations for a family of one-more

assumptions in lattice-based cryptography. To that end, we propose a strengthening of the OM-ISIS assumption that we call as randomized OM-ISIS (or, rOM-ISIS for short) assumption.

Our intuition is to draw more inspiration from GPV signatures. Recall that in GPV signatures, a signature is computed as a preimage of the output of a hash function (modeled as a random oracle). While modeling the hash function as a random oracle enables a reduction to plain ISIS by a standard RO programming argument, usage of any specific hash function (such as SHA-3) is not known to enable any efficient attacks. At a high level, the hash function protects the signer from simple algebraic manipulations of multiple preimage vectors to create a new valid preimage vector for a fresh hash output. A little more abstractly, this means that given preimage vectors $\{\mathbf{t}_i = \mathbf{A}_\varsigma^{-1}(H(\mu_i))\}_i$, it is unclear how to find short coefficients α_i s.t. $\sum \alpha_i \mathbf{t}_i = \mathbf{A}_\varsigma^{-1}(H(\mu^*))$. That is, the hash function prevents from creating a valid preimage to the hash function which is a short linear combination of other hash values.

Inspired by the intuitive structural guarantee provided by a hash function, we propose the following generalization of the OM-ISIS assumption.

Assumption 6.1 (randomized OM-ISIS)**.** Let $q, n, m, \varsigma, \beta$ be functions of security parameter λ. Consider the following experiment:

1. The challenger uniformly samples two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_q^{n \times m}$, and sends $\mathbf{A}, \mathbf{B}$ to adversary $\mathcal{A}$.
2. $\mathcal{A}$ adaptively makes queries of the following types to the challenger, in any order.

   Syndrome queries. $\mathcal{A}$ requests for a challenge vector, to which the challenger replies with a uniformly sampled vector $\mathbf{t} \leftarrow\!\!\$\, \mathbb{Z}_q^n$. We denote the set of received vectors by S.

   Preimage queries. $\mathcal{A}$ queries a vector $\widehat{\mathbf{t}} \in \mathbb{Z}_q^n$, to which the challenger replies with a short vector $\widehat{\mathbf{x}} \in \mathbb{Z}_q^m$ and a ± 1 vector $\widehat{\mathbf{y}} \in \{\pm 1\}^m$ such that $\mathbf{A} \cdot \widehat{\mathbf{x}} + \mathbf{B} \cdot \widehat{\mathbf{y}} = \widehat{\mathbf{t}}$ and $\|\widehat{\mathbf{x}}\| \leq \varsigma \sqrt{m}$. Let ℓ denote the total number of preimage queries.
3. Finally, $\mathcal{A}$ outputs $\ell + 1$ tuples of the form $\left\{(\mathbf{x}_j, \mathbf{y}_j, \mathbf{t}_j)\right\}_{j \in [\ell+1]}$. $\mathcal{A}$ wins if

$$\forall j \in [\ell + 1], \qquad \mathbf{A} \cdot \mathbf{x}_j + \mathbf{B} \cdot \mathbf{y}_j = \mathbf{t}_j, \text{ and } \left\|\mathbf{x}_j\right\| \leq \beta, \mathbf{y}_j \in \{\pm 1\}^m \text{ and } \mathbf{t}_j \in S.$$

The $\mathsf{rOM\text{-}ISIS}_{q,n,m,\varsigma,\beta}$ assumption states that for every PPT adversary $\mathcal{A}$, the probability that $\mathcal{A}$ wins is $\mathsf{negl}(\lambda)$.

Intuitively, the new assumption says an attacker does not receive preimages for arbitrary target vectors $\widehat{\mathbf{t}}$ that it selects, but for publicly re-randomized vector $\mathbf{t}' = \widehat{\mathbf{t}} - \mathbf{B} \cdot \mathbf{y}$, where $\mathbf{y}$ is a random ± 1 vector (chosen by the challenger). We even provide the attacker with the vector $\mathbf{y}$. The point is that the ability to re-randomize the target vector, gives the challenger more flexibility in answering preimage queries. Now an attacker can win as long as it creates preimage vectors for any re-randomization of the random syndrome vectors. That is, we allow the attacker to also select any arbitrary ± 1 vector and use it to re-randomize each syndrome vector. The only constraint is that the vectors $\mathbf{y}_j$ are ± 1 vectors.

Unlike [AKSY22], we do not make any additional parameter restrictions. This is primarily due to the fact that we have been unable to find any practical attacks for any standard ISIS parameter regimes for the rOM-ISIS assumption. In our perspective, the condition that $\mathbf{y}_j$ must be ± 1 vectors prevents the algebraic attacks that were mounted on the OM-ISIS assumption. Additionally, one could view the $\mathbf{B} \cdot \mathbf{y}$ term as enforcing a structural property, on the target vectors, that a hash function enforces for GPV signatures. We provide some preliminary cryptanalysis next.

6.1 Cryptanalysis of the rOM-ISIS assumption

We analyse the robustness of the randomized one-more ISIS assumption. In order to do so, we consider the two categories of attacks presented against the one-more ISIS assumption by Agrawal et al. [AKSY22], and show that each of these attacks is in fact harder in the randomized one-more ISIS setting. First, let us recall the two types of attacks presented in [AKSY22].

6.1.1 Recalling attacks on the one-more ISIS assumption

Lattice-based attack. The attacker can find a solution to the one-more ISIS problem in polynomial time as follows:

1. The attacker makes $\Theta(m^2)$ preimage queries for $\mathbf{t}_i = \mathbf{0}$ to the OM-ISIS oracle.
2. Of the $\Theta(m^2)$ preimages, m of them will be linearly independent with non-zero probability [Reg05] with norm $\Theta(\sqrt{m}\cdot\varsigma)$ [BF11]. Using this fact, one can find m linearly independent vectors that span the subspace of rank n. These m vectors can be used to form a basis $\mathbf{E}$ for $\Lambda_q^{\perp}(\mathbf{C})$ such that the corresponding Gram-Schmidt basis $\widetilde{\mathbf{E}}$ has norm $\sqrt{m}\cdot\varsigma$.
3. From here, the attacker simply runs Babai's Nearest Plane Algorithm on $(\mathbf{E}, \mathbf{u})$ where vector $\mathbf{u} \in \mathbb{Z}_q$ is a general solution to the OM-ISIS challenge, $\mathbf{t}$. The output $\mathbf{v}$ of the algorithm is such that $|\mathbf{u}-\mathbf{v}| \leq \frac{1}{2}\sqrt{\sum_{i\in[m]}\|\widetilde{\mathbf{e}_i}\|^2}$ which is at most $m\varsigma$ with overwhelming probability.

Since $\mathbf{v}$ is a point on the lattice, it follows that $(\mathbf{u}-\mathbf{v})$ is a valid solution to OM-ISIS with norm bound $\beta = \Theta(m\varsigma)$. A tighter bound of $\beta = \Theta(\sqrt{mn\log q})$ is also shown for an exponential time attack leveraging a shortest vector problem solver for the vector $\mathbf{u}$.

Combinatorial attack. The attacker can find a solution to the one-more ISIS problem in polynomial time as follows:

1. The attacker constructs a set $A = \{a\cdot\mathbf{e}_i \mid \forall i \in [n], a \in \mathbb{Z}_q\}$ of size nq, where $\mathbf{e}_i$'s are the canonical n-dimensional basis vectors.
2. It then queries the OM-ISIS oracle for a preimage of each vector in A.
3. Since any vector in $\mathbb{Z}_q^n$ can be expressed as the sum of at most n vectors in A, a preimage $\mathbf{z}$ for any challenge $\mathbf{t}$ can be found by simply adding up the preimages corresponding to the vectors in A that sum to $\mathbf{t}$.

Given that with all but negligible probability, the original preimages have norms bounded by $\sqrt{m}\cdot\varsigma$ (Lemma 3.1), it follows that $\mathbf{z}$ is a valid solution to OM-ISIS with norm bound $\beta = \Theta(\sqrt{nm}\cdot\varsigma)$ with overwhelming probability. In [AKSY22], a slightly more specialised version of this attack is presented for $Q \geq nq$ preimage queries and $\beta = \Theta\left(\sqrt{m\cdot\left(1+\frac{n\log q}{\log(Q/n^2)}\right)}\cdot\varsigma\right)$.

Remark 6.2. We also observe that an even faster attack is possible at just a small cost to the norm bound. Specifically, the attacker can instead construct the set $A = \left\{2^j\cdot\mathbf{e}_i \mid \forall i \in [n], j \in [\lfloor\log q\rfloor]\right\}$, query each vector in the set as before. The observation is that $\mathbf{t} = \begin{bmatrix} t_1 & t_2 & \cdots & t_n \end{bmatrix}^{\mathsf{T}}$ can be expressed

as $\sum_{i\in[n]} \mathbf{e}_i \cdot \sum_{j\in J_i} 2^j$, where each $J_i \subseteq [\lfloor \log q \rfloor]$ is the set of indices where the binary decomposition of t_i is 1. Thus, by taking the sum of at most $n\lfloor \log q \rfloor$ preimages of vectors in A (and making just as many queries), an attacker can construct a valid solution to OM-ISIS, with norm bound $\beta = \Theta(\sqrt{nm \log q} \cdot \varsigma)$ with overwhelming probability.

6.1.2 Attacking the rOM-ISIS assumption

We now consider variations of these attack strategies against our randomized one-more ISIS assumption.

Lattice-based attack strategy

Let us begin by considering the lattice-based attack strategy against the rOM-ISIS assumption. We will argue that under this assumption, it is hard for an attacker to even efficiently *construct* a good basis for $\Lambda_q^\perp(\mathbf{C})$.

1. Following the same first step as the lattice attack above, suppose the attacker makes Q preimage queries for $\mathbf{t}_i = \mathbf{0}$ to the rOM-ISIS oracle and receives $\{(\mathbf{z}_i, \mathbf{y}_i)\}_{i\in[Q]}$ such that for each $i \in [Q]$, $\mathbf{C} \cdot \mathbf{z}_i + \mathbf{B} \cdot \mathbf{y}_i = \mathbf{0}$, $\mathbf{z}_i$ is short and $\mathbf{y}_i \in \{\pm 1\}^m$.

Here, we make the following observation:

Observation 6.3. If there exist $a_1, a_2, \ldots, a_Q \in \mathbb{Z}_q$ such that $\mathbf{B} \cdot \sum_{i\in[Q]}(a_i \cdot \mathbf{y}_i) = \mathbf{0}$, then $\boldsymbol{\alpha} := \sum_{i\in[Q]}(a_i \cdot \mathbf{z}_i)$ is a vector on the lattice $\Lambda_q^\perp(\mathbf{C})$. Further, for γ such that $|a_i| \le \gamma$ for all $i \in [Q]$ we have

(i) $\boldsymbol{\alpha}$ has norm bounded above by $Q\gamma\sqrt{m} \cdot \varsigma$ with all but negligible probability; and

(ii) $\left\|\sum_{i\in[Q]}(a_i \cdot \mathbf{y}_i)\right\|_2 \le Q\gamma\sqrt{m} = \|\boldsymbol{\alpha}\|_2/\varsigma$

At this stage, we identify two types of potential attackers:

- Type 1: This type of attacker is able to find a collection $a_1, a_2, \ldots, a_Q \in \mathbb{Z}_q$ as defined in Observation 6.3 such that $|a_i| \le \gamma$ for all a_i. It can use this to construct a short basis for $\Lambda_q^\perp(\mathbf{C})$ and proceed as in step 3 of the lattice attack above.
- Type 2: This type of attacker is unable to find a collection $a_1, a_2, \ldots, a_Q \in \mathbb{Z}_q$ of bounded length, and instead proceeds in some other way.

Claim 6.4. Assuming the hardness of $\mathsf{SIS}_{q,n,m,\beta'}$ for $\beta' = Q\gamma\sqrt{m}$, a type 1 attacker exists with negligible probability.

Proof. This follows by recalling that $\mathbf{B} \in \mathbb{Z}_q^{n\times m}$ is a random matrix, and observing that $\sum_{i\in[Q]}(a_i \cdot \mathbf{y}_i)$ is a solution to $\mathsf{SIS}_{q,n,m,\beta'}$ where $\beta' = \|\boldsymbol{\alpha}\|/\varsigma = Q\gamma\sqrt{m}$. □

Essentially, to match the norm bound on the lattice basis as in [AKSY22] we require $\|\boldsymbol{\alpha}\| = \sqrt{m} \cdot \varsigma$, which suggests that finding even one such collection of a_i's is at least as hard as solving $\mathsf{SIS}_{q,n,m,\beta'}$ for $\beta' = \Theta(\sqrt{n \log q})$. Next, while it is evident that a type 2 attacker cannot proceed according to the lattice-based attack strategy of [AKSY22], it is less clear how else the attacker might

proceed. In particular, we find that the process of constructing a good basis would ultimately involve taking some combination of the preimage queries. However, any such combination that does not satisfy the type 1 condition will also result in a larger norm on the solutions. We leave further cryptanalysis and discovery of an alternative lattice-based attack strategy as an interesting open problem.

A more generalized attack. We can generalize this attack a little further by slightly modifying the first step:

1. Suppose the attacker makes Q preimage queries for *any* $\mathbf{t}_i \in \mathbb{Z}_q^n$ to the rOM-ISIS oracle and receives $\{(\mathbf{z}_i, \mathbf{y}_i)\}_{i\in[Q]}$ such that for each $i \in [Q]$, $\mathbf{C}\cdot\mathbf{z}_i + \mathbf{B}\cdot\mathbf{y}_i = \mathbf{t}_i$.

Now, one strategy could be to guess $\mathbf{y}_i$ and set the query $\mathbf{t}_i = \mathbf{B}\cdot\mathbf{y}_i$, but the probability of guessing correctly is negligible. Alternatively, the attacker can try to find $a_1, a_2, \ldots, a_Q \in \mathbb{Z}_q$ such that $\mathbf{B}\cdot\sum_{i\in[Q]}(a_i\cdot\mathbf{y}_i) = \sum_{i\in[Q]} a_i\cdot\mathbf{t}_i$ as in the previous case, then too $\boldsymbol{\alpha} := \sum_{i\in[Q]}(a_i\cdot\mathbf{z}_i)$ is a vector on the lattice $\Lambda_q^{\perp}(\mathbf{C})$. Indeed, a similar analysis shows that the attack makes Q queries, solves at least m instances of $\mathsf{ISIS}_{q,n,m,\beta'}$ for $\beta' = \|\boldsymbol{\alpha}\|/\varsigma$, and outputs a solution to rOM-ISIS for $\beta = \Theta\left(\sqrt{m}\cdot\|\boldsymbol{\alpha}\|\right)$. Beyond these attacks, we could not find any efficient (sub-exponential time) lattice attacks against our new assumption.

Combinatorial attack strategy

Turning now to the combinatorial attack, we notice that extending the previous strategy of [AKSY22] — i.e., solving for a preimage of the challenge by adding the preimages corresponding to the scaled canonical basis that sum to the challenge — no longer works. This is because every query response to a member of the set A now additionally contains a uniformly chosen vector $\mathbf{y}_i \in \{\pm 1\}^m$. Instead, consider an attacker that does the following:

1. The attacker constructs the set $A = \{a\cdot\mathbf{e}_i + \mathbf{B}\cdot(\mathbf{y}'_j - \mathbf{1}) \mid \forall i \in [n],\ \forall j \in [Q], a \in \mathbb{Z}_q\}$ where $\mathbf{e}_i$'s are the canonical n-dimensional basis vectors, and each $\mathbf{y}'_j \in \{\pm 1\}^m$.

2. It then queries the rOM-ISIS oracle for a preimage of each vector in A.

3. Consider any challenge vector $\mathbf{t} \in \mathbb{Z}_q^n$. The attacker finds a collection of n vectors of the form $a_i\cdot\mathbf{e}_i$ such that they sum to $n\cdot\mathbf{t}$ where each $a_i \in \mathbb{Z}_q$.

4. Let $(\mathbf{z}_{i,j}, \mathbf{y}_{i,j})$ be the oracle response for query $a_i\cdot\mathbf{e}_i + \mathbf{B}\cdot(\mathbf{y}'_j - 1)$. For each $i \in [n]$, the next step is to find any $j \in Q$ such that $\mathbf{y}'_j = \mathbf{y}_{i,j}$. For every i, the probability that such a j exists is $1 - (1 - 2^{-m})^Q \approx 1 - e^{-Q/2^m}$, which is bounded away from zero for $Q > 2^m$.

5. Let set I contain all such pairs (i, j), and set J contain the corresponding j. Then, the following holds

$$\mathbf{C}\cdot\sum_{(i,j)\in I}\mathbf{z}_{i,j} + \mathbf{B}\cdot\sum_{(i,j)\in I}\mathbf{y}_{i,j} = \sum_{i\in[n]} a_i\mathbf{e}_i + \mathbf{B}\cdot\sum_{j\in J}(\mathbf{y}'_j - \mathbf{1}) \quad .$$

   Which upon simplifying gives $\mathbf{C}\cdot\left(\sum_{(i,j)\in I}\mathbf{z}_{i,j}\right)/n + \mathbf{B}\cdot\mathbf{1} = \mathbf{t}$.

Recall that with all but negligible probability the original preimages $\mathbf{z}_{i,j}$ have norms bounded by $\sqrt{m} \cdot \varsigma$ (Lemma 3.1). Thus, $\left(\left(\sum_{(i,j)\in I} \mathbf{z}_{i,j}\right)/n, \mathbf{1}\right)$ is a valid rOM-ISIS solution for $\beta = \Theta\left(\sqrt{m/n} \cdot \varsigma\right)$ with overwhelming probability. Notably, this attack is computationally intractable for any PPT adversary making polynomially many queries, as the probability of finding set I (and J) goes to zero when $Q = o(2^m)$.

Summary. Our preliminary cryptanalysis suggests that our new assumption is relatively more robust than the OM-ISIS assumption against the same class of attacks. More precisely,

- For the lattice-based attack strategy, we are able to show a partial reduction to SIS (under certain conditions).
- For the combinatorial attack strategy, we give an *exponential* time algorithm for $\beta = \Theta\left(\sqrt{m/n} \cdot \varsigma\right)$.

  In contrast, the OM-ISIS assumption has (i) a $\Omega(nq)$ time algorithm for $\beta = \Theta\left(\sqrt{m \cdot \left(1 + \frac{n \log q}{\log(Q/n^2)}\right)} \cdot \varsigma\right)$; and (ii) a $\Omega(n\lfloor \log q \rfloor)$ time algorithm for $\beta = \Theta(\sqrt{nm \log q} \cdot \varsigma)$.

We leave the challenges of further cryptanalysis and formally proving security of the rOM-ISIS assumption as potential future directions.

7 Lattice-based NIBS

We begin this section by describing our NIBS scheme that we prove secure under the rOM-ISIS assumption.

Let parameters $n, m, \varsigma, \beta = \varsigma\sqrt{m}$, a prime number q be functions of the security parameter λ such that the randomized one-more ISIS instance $\mathsf{rOM\text{-}ISIS}_{q,n,2m,\varsigma,3\sqrt{2}\beta}$ is hard. These parameters must satisfy the following constraints:

$$n = \mathsf{poly}(\lambda),\ m > n \log q + \lambda,\ \varsigma/m = \Omega(1),\ \beta < m\varsigma \tag{1}$$

Tools required. Our construction relies on a public key encryption scheme $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, a lattice trapdoor $\mathsf{bLT} = (\mathsf{TrapGen}, \mathsf{SamplePre})$, and a NIZK proof system $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ for the following language:

Language $\mathcal{L}_2$

Instance: Each instance x is interpreted matrices $\mathbf{C}, \mathbf{A}$ and $\mathbf{B}$, PKE public key pke.pk, ciphertext ct, vector $\mathbf{w}$ and random string δ.

Witness: Witness ω consists of a secret vector $\mathbf{x}$, nonce vector $\mathbf{y}$, presignature vector $\mathbf{z}$ and randomness r.

Membership: ω is a valid witness for x if the following are satisfied:

- $\|\mathbf{z}\| \le \sqrt{2}\beta$ and $\|\mathbf{x}\| \le \sqrt{2}\beta/m$ and $\mathbf{y} \in \{\pm 1\}^{2m}$.
- ct is an encryption of $\mathbf{x}\|\mathbf{y}\|\mathbf{z}$ using randomness r, s.t. $\mathsf{ct} = \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}\|\mathbf{z}; r)$
- $\mathbf{C} \cdot \mathbf{z} + \mathbf{B} \cdot \mathbf{y} = \mathbf{A} \cdot \mathbf{x} + \mathsf{H}(\delta)$ and $\mathbf{w} = \mathbf{A}_{\mathsf{L}} \cdot \mathbf{x}_{\perp} + \mathbf{A}_{\mathsf{R}} \cdot \mathbf{z}_{\perp}$

7.1 Construction

Below we describe our non-interactive blind signature scheme.

$\mathsf{Setup}(1^\lambda, n, m, q, \varsigma, \mathsf{H}) \to \mathsf{pp}$. It samples $\mathbf{A}, \mathbf{B} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times 2m}$. Next, it runs the key generating algorithm of PKE and generates the public and secret key pair as

$$(\mathsf{pke.pk}, \mathsf{pke.sk}) \leftarrow\!\!\$\ \mathsf{PKE.KeyGen}(1^\lambda).$$

Next, it generates $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$ and outputs:

$$\mathsf{pp} := (\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs}).$$

Note that H is the hash function which we model as a random oracle.

$\mathsf{KeyGen}_S(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{vk})$. Runs the setup algorithm of lattice trapdoor and obtains

$$(\mathbf{T_C}, \mathbf{C}) \leftarrow\!\!\$\ \mathsf{bLT.TrapGen}(1^\lambda, n, 2m, q).$$

It outputs signer's secret key and verification key as $\mathsf{sk} := \mathbf{T_C}$ and $\mathsf{vk} := \mathbf{C}$.

$\mathsf{KeyGen}_R(\mathsf{pp}) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. It samples $\mathbf{x} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}_q^{2m}, \varsigma/m}$ and $\delta \leftarrow\!\!\$\ \{0,1\}^\lambda$. Next, it computes

$$\mathbf{t} = \mathbf{A}\cdot\mathbf{x} + \mathsf{H}(\delta).$$

It outputs user's secret key and public key as $(\mathsf{sk}_R := (\mathbf{x}, \delta), \mathsf{pk}_R := \mathbf{t})$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \to (\mathsf{psig}, \mathsf{nonce})$. The issue algorithm samples a random ± 1 vector $\mathbf{y} \leftarrow\!\!\$\ \{\pm 1\}^{2m}$. Next, using the signing key $\mathsf{sk} = \mathbf{T_C}$ and the receiver's public key $\mathsf{pk}_R = \mathbf{t}$, it generates

$$\mathbf{z} \leftarrow \mathsf{bLT.SamplePre}(\mathbf{C}, \mathbf{T_C}, \mathbf{t} - \mathbf{B}\cdot\mathbf{y}, \varsigma),$$

and outputs the presignature $\mathsf{psig} := \mathbf{z}$, and nonce as $\mathsf{nonce} := \mathbf{y}$.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, \mathsf{psig}, \mathsf{nonce}) \to (\mu, \sigma)$. It parses sk_R as $(\mathbf{x}, \delta)$. Then, assigns $\mathbf{C} := \mathsf{vk}$, $\mathbf{z} := \mathsf{psig}$, and $\mathbf{y} := \mathsf{nonce}$. It checks if $\mathbf{C}\cdot\mathbf{z} + \mathbf{B}\cdot\mathbf{y} = \mathbf{A}\cdot\mathbf{x} + \mathsf{H}(\delta)$, $\|\mathbf{z}\| \le \sqrt{2}\beta$ and $\mathbf{y} \in \{\pm 1\}^{2m}$. If any check fails it aborts and outputs $\perp$.

Otherwise, it generates

$$\mathsf{ct} \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}\|\mathbf{z}; r)$$

from uniformly sampled randomness $r \leftarrow\!\!\$\ \{0,1\}^\lambda$. The obtain algorithm sets $\mathbf{w}$ as $\mathbf{w} = \mathbf{A}\cdot\begin{bmatrix}\mathbf{x}_\perp \\ \mathbf{z}_\perp\end{bmatrix}$, where $\mathbf{x}_\perp, \mathbf{z}_\perp \in \mathbb{Z}_q^m$ and generates NIZK proof

$$\pi \leftarrow \mathsf{NIZK.Prove}\left(\mathsf{nizk.crs}, x := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}, \mathbf{w}, \delta), \omega := (\mathbf{x}, \mathbf{y}, \mathbf{z}, r)\right)$$

Finally, it outputs message $\mu := (\mathbf{w}, \delta)$ and signature $\sigma := (\pi, \mathsf{ct})$.

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0,1\}$. It parses μ as $(\mathbf{w}, \delta)$ and σ as (π, ct). The verification algorithm accepts and outputs 1 if and only if

$$\mathsf{NIZK.Verify}\left(\mathsf{nizk.crs}, x := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}, \mathbf{w}, \delta), \pi\right) = 1.$$

Otherwise, it outputs 0.

Completeness. Observe that by correctness of bLT.SamplePre, $\mathbf{z}$ is a presignature on $\mathbf{A}\cdot\mathbf{x}+\mathsf{H}(\delta)-\mathbf{B}\cdot\mathbf{y}$, ie., $\mathbf{C}\cdot\mathbf{z}=\mathbf{A}\cdot\mathbf{x}+\mathsf{H}(\delta)-\mathbf{B}\cdot\mathbf{y}$ such that $\|\mathbf{z}\|\leq\sqrt{2}\beta$ and $y_i\in\{\pm 1\}$ for all $i\in[2m]$ where y_i is the i^{th} element of $\mathbf{y}$. Since, it further holds by construction that $\mathbf{w}=\mathbf{A}_{\mathsf{L}}\cdot\mathbf{x}_{\perp}+\mathbf{A}_{\mathsf{R}}\cdot\mathbf{z}_{\perp}$, $\mathsf{ct}=\mathsf{PKE.Enc}(\mathsf{pke.pk},\mathbf{x}\|\mathbf{y}\|\mathbf{z};r)$, and (by Lemma 3.1) $\|\mathbf{x}\|\leq\sqrt{2}\beta/m$, we have that π is a valid NIZK proof for the language $\mathcal{L}_2$. It therefore follows from the completeness of the underlying NIZK that Construction 7.1 is complete.

Reusability. Notice that for $b\in\{0,1\}$, if a signer issues presignature-nonce pairs $(\mathbf{z}_b,\mathbf{y}_b)$ to a given receiver with secret vector $\mathbf{x}$ then,

$$\begin{aligned}\Pr[\mathbf{y}_0=\mathbf{y}_1\vee\mu_0=\mu_1]=&\ \Pr[\mu_0=\mu_1]+\Pr[\mathbf{y}_0=\mathbf{y}_1]\\ =&\ \Pr[(\mathbf{w}_0,\delta)=(\mathbf{w}_1,\delta)]+\Pr[\mathbf{y}_0=\mathbf{y}_1]\\ =&\ \Pr\left[\mathbf{A}\cdot\begin{bmatrix}\mathbf{0}\\ \mathbf{z}_{0,\perp}-\mathbf{z}_{1,\perp}\end{bmatrix}=\mathbf{0}\right]+\mathsf{negl}(\lambda)\ .\end{aligned}$$

The probability that $\mathbf{z}_{0,\perp}=\mathbf{z}_{1,\perp}$ is negligible in λ, otherwise conditioned on $\mathbf{z}_{0,\perp}\neq\mathbf{z}_{1,\perp}$, we have a short solution for SIS. Thus the overall probability is negligible and our construction satisfies the reusability property.

7.2 One-more unforgeability

Consider first, the one-more unforgeability of this protocol.

Theorem 7.1. Assume that NIZK proof system NIZK satisfies soundness, lattice trapdoor bLT satisfies well-distributedness, and the rOM-ISIS assumption holds, then our Construction 7.1 is one-more unforgeable.

Proof. Let us define the following hybrids:

Hybrid_0 This is the actual one-more unforgeability game for NIBS:

1. Challenger samples $\mathbf{A},\mathbf{B}\leftarrow\$\ \mathbb{Z}_q^{n\times 2m}$ uniformly at random, $\mathsf{pke.pk}\leftarrow\$\ \mathsf{PKE.KeyGen}(1^\lambda)$, $\mathsf{nizk.crs}\leftarrow\$\ \mathsf{NIZK.Setup}(1^\lambda)$ and sets pp as $(\mathbf{A},\mathbf{B},\mathsf{pke.pk},\mathsf{nizk.crs})$. Next, it samples $(\mathbf{T}_{\mathbf{C}},\mathbf{C})\leftarrow\$\ \mathsf{bLT.TrapGen}(1^\lambda,n,2m,q)$ and sets $\mathsf{sk}:=\mathbf{T}_{\mathbf{C}}$, $\mathsf{vk}:=\mathbf{C}$. Challenger sends pp and vk to the adversary.
2. Adversary chooses pk_R and requests a presignature, to which the challenger replies with $(\mathsf{psig},\mathsf{nonce})\leftarrow\$\ \mathsf{Issue}(\mathsf{sk},\mathsf{pk}_R)$. The adversary repeats this step a total of ℓ times.
3. Adversary outputs k ($k=\ell+1$) message and signature pairs $((\mu_1,\sigma_1),\ldots,(\mu_k,\sigma_k))$. It wins if $\mu_i\neq\mu_j$ for $1\leq i<j\leq k$, $\mathsf{Verify}(\mathsf{vk},\mu_i,\sigma_i)=1$ for all $i\in[k]$.

Hybrid_1 This is the same as Hybrid_0 aside from one key difference that instead of uniformly sampling the matrix $\mathbf{A}$, it is chosen by first sampling a matrix $\mathbf{R}\leftarrow\$\ \{0,1\}^{2m\times 2m}$ and then setting $\mathbf{A}=\mathbf{C}\cdot\mathbf{R}$.

1. Challenger samples $\mathbf{R}\leftarrow\$\ \{0,1\}^{2m\times 2m}$ uniformly at random, samples $(\mathbf{T}_{\mathbf{C}},\mathbf{C})\leftarrow\$\ \mathsf{bLT.TrapGen}(1^\lambda,n,2m,q)$, and sets $\mathbf{A}=\mathbf{C}\cdot\mathbf{R}$. Challenger then samples $\mathbf{B}\leftarrow\$\ \mathbb{Z}_q^{n\times 2m}$, $\mathsf{pke.pk}\leftarrow\$\ \mathsf{PKE.Setup}(1^\lambda)$, $\mathsf{nizk.crs}\leftarrow\$\ \mathsf{NIZK.Setup}(1^\lambda)$, and sets pp as $(\mathbf{A},\mathbf{B},\mathsf{pke.pk},\mathsf{nizk.crs})$. Next, it sets $\mathsf{sk}:=\mathbf{T}_{\mathbf{C}}$, $\mathsf{vk}:=\mathbf{C}$. Challenger sends pp and vk to the adversary.

Let $\mathsf{Adv}^{j}_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in the one-more unforgeability game in Hybrid_j, then the following must hold:

Lemma 7.2. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^{0}_{\mathcal{A}} - \mathsf{Adv}^{1}_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Note that $\mathbf{R} \leftarrow\!\$\, \{0,1\}^{2m\times 2m}$, and thus $\mathbf{H}_{\infty}(\mathbf{R}) = 4m^2$. As $4m^2 > 2m \cdot n \log q + \lambda$. It follows from Leftover Hash Lemma (**Corollary** 3.8) that the following distributions are statistically indistinguishable:

$$\{(\mathbf{C}', \mathbf{C}' \cdot \mathbf{R}) : \mathbf{C}' \leftarrow\!\$\ \mathbb{Z}_q^{n\times 2m}, \mathbf{R} \leftarrow\!\$\, \{0,1\}^{2m\times 2m}\} \approx_s \{(\mathbf{C}', \mathbf{A}) : \mathbf{C}' \leftarrow\!\$\ \mathbb{Z}_q^{n\times 2m}, \mathbf{A} \leftarrow\!\$\, \mathbb{Z}_q^{n\times 2m}\}.$$

In the above equation, $\mathbf{C}'$ is sampled from $\mathbb{Z}_q^{n\times 2m}$ uniformly at random. By Definition 3.3, the Well Distributedness property implies that the statistical distance between $\mathbf{C}'$ and $\mathbf{C}$ is within $2^{-\Omega(\lambda)}$. Thus,

$$\{(\mathbf{C}, \mathbf{C} \cdot \mathbf{R}) : (\mathbf{T}_{\mathbf{C}}, \mathbf{C}) \leftarrow\!\$\, \mathsf{bLT.TrapGen}(1^{\lambda}, n, 2m, q), \mathbf{R} \leftarrow\!\$\, \{0,1\}^{2m\times 2m}\} \approx_s \{(\mathbf{C}, \mathbf{A}) : \mathbf{C} \leftarrow\!\$\ \mathbb{Z}_q^{n\times 2m}, \mathbf{A} \leftarrow\!\$\, \mathbb{Z}_q^{n\times 2m}\}.$$

□

Lemma 7.3. Assuming that NIZK argument is sound and the randomized one-more ISIS assumption holds, adversary $\mathcal{A}$ can have at most negligible advantage in Hybrid_1.

Proof. This follows from a case by case reduction algorithm. Suppose there exists a PPT attacker $\mathcal{A}$ that wins the one-more unforgeability game with non-negligible probability $\epsilon = \epsilon(\lambda)$. It outputs k message and signature pairs $((\mu_1, \sigma_1), \ldots, (\mu_k, \sigma_k))$, and wins if and only if $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, $\mathsf{Verify}(\mathsf{vk}, \mu_i, \sigma_i) = 1$ for $1 \leq i \leq k$, and $k = \ell + 1$. We divide the potential attacker into the following cases:

- Type 1: Consider that we parse inputs and signatures (μ_i, σ_i) from $\mathcal{A}$'s output as $((\mathbf{w}_i, \delta_i), (\pi_i, \mathsf{ct}_i))$ for all $i \in [k]$. For type 1 attacker $\mathcal{A}$, there exists some index $j \in [k]$ such that $\mathsf{NIZK.Verify}\left(x^* := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_j, \mathbf{w}_j, \delta_j), \pi_j\right) = 1$ and x^* is not a valid instance for language $\mathcal{L}_2$.
- Type 2: Consider ct_i as encryptions of $\mathbf{x}_i$, $\mathbf{y}_i$, and $\mathbf{z}_i$, such that $\mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_i \| \mathbf{y}_i \| \mathbf{z}_i; r) = \mathsf{ct}_i$ and $\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i \in \mathbb{Z}_q^{2m}$. For type 2 attacker $\mathcal{A}$, there exists some $j \in [k]$, such that $\mathbf{C} \cdot \mathbf{z}_j + \mathbf{B} \cdot \mathbf{y}_j = \mathbf{A} \cdot \mathbf{x}_j + \mathsf{H}(\delta_j)$ and the attacker never queries random oracle H using input δ_j.
- Type 3: For all $i \in [k]$, $x_i := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_i, \mathbf{w}_i, \delta_i)$ is a valid instance for language $\mathcal{L}_2$, and it makes at least one hash query $\mathsf{H}(\delta_i)$ using input δ_i.

Claim 7.4. Assuming that NIZK satisfies soundness property, then type 1 attacker $\mathcal{A}$ has at most negligible advantage.

Proof. Assume that type 1 attacker $\mathcal{A}$ has non-negligible advantage $\epsilon = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks soundness of nizk.

The NIZK soundness challenger starts by sampling and outputting $\mathsf{nizk.crs} \leftarrow\!\$\, \mathsf{NIZK.Setup}(1^{\lambda})$. With nizk.crs, $\mathcal{B}$ then generates $\mathbf{T}_{\mathbf{C}}, \mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}$ according to Hybrid 1, and sets $\mathsf{pp} := (\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs})$. It sends pp and vk to $\mathcal{A}$. Next, $\mathcal{B}$ responses to $\mathcal{A}$'s ℓ presignature queries. $\mathcal{A}$ then outputs k message and signature pairs where there exists some j such that $\mathsf{NIZK.Verify}\left(\mathsf{nizk.crs}, x^*, \pi_j\right) =$

1 and x^* is not a valid instance for $\mathcal{L}_2$, as explained above. If type 1 adversary $\mathcal{A}$ has non-negligible advantage $\epsilon(\lambda)$, then $\mathcal{B}$ univormly randomly samples $j' \leftarrow\$ [k]$ and outputs $x' := (\mathbf{C}, \mathbf{A}, \mathsf{pke.pk}, \mathsf{ct}_{j'}, \mathbf{w}_{j'}, \delta_{j'}, \pi_{j'})$. $\mathcal{B}$ breaks NIZK soundness game with a non-negligible probability $\epsilon(\lambda)/k$. □

Claim 7.5. Type 2 attacker $\mathcal{A}$ has at most negligible advantage.

Proof. Since type 2 attacker $\mathcal{A}$ never queries $\mathsf{H}(\delta_j)$, value of $\mathsf{H}(\delta_j)$ is not defined and is completely random in the view of $\mathcal{A}$, as H is used as a random oracle. Such attacker $\mathcal{A}$ can have at most negligible advantage because it must hold that $\mathsf{H}(\delta_j) = \mathbf{C} \cdot \mathbf{z}_j + \mathbf{B} \cdot \mathbf{y}_j - \mathbf{A} \cdot \mathbf{x}_j$. □

Claim 7.6. Assuming that rOM-ISIS assumption holds, type 3 adversary $\mathcal{A}$ has at most negligible advantage.

Proof. Assume that type 3 attacker $\mathcal{A}$ has non-negligible advantage $\epsilon = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks the rOM-ISIS assumption.

The rOM-ISIS challenger starts by sampling and outputting a matrix $\mathbf{C}, \mathbf{B} \in \mathbb{Z}_q^{n \times 2m}$ uniformly at random. Reduction algorithm $\mathcal{B}$ then samples $\mathbf{R} \leftarrow\$ \{0,1\}^{2m \times 2m}$ and sets $\mathbf{A}$ as $\mathbf{A} = \mathbf{C} \cdot \mathbf{R}$. Next, $\mathcal{B}$ generates $(\mathsf{pke.pk}, \mathsf{pke.sk})$ and $\mathsf{nizk.crs}$. $\mathcal{B}$ outputs $\mathsf{pp} := (\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs})$ and verification key $\mathsf{vk} := \mathbf{C}$.

Next, $\mathcal{A}$ is allowed to make a series of hash queries and presignature queries (in any order). On each fresh hash query that $\mathcal{A}$ provides an input δ, $\mathcal{B}$ simply makes a syndrome query to the challenger. $\mathcal{B}$ forwards challenger's output to $\mathcal{A}$ as the output of $\mathsf{H}(\delta)$. Meanwhile, $\mathcal{B}$ keeps a recording of all hash queries from $\mathcal{A}$. For each presignature query on input $\mathbf{t}'$ from $\mathcal{A}$, $\mathcal{B}$ makes a preimage query to the challenger using the same input $\mathbf{t}'$. Then, $\mathcal{B}$ forwards challenger's output $\mathbf{z}'$ and $\mathbf{y}'$ to $\mathcal{A}$ such that $\mathbf{C} \cdot \mathbf{z}' + \mathbf{B} \cdot \mathbf{y}' = \mathbf{t}'$. $\mathcal{A}$ makes a total of ℓ such presignature queries. To win the one-more unforgeability game, $\mathcal{A}$ outputs k message and signature pairs $((\mu_1, \sigma_1), \ldots, (\mu_k, \sigma_k))$ such that $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, $\mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \mu_i, \sigma_i) = 1$ for all $i \in [k]$, and $k = \ell + 1$.

For all $i \in [k]$, $\mathcal{B}$ parses μ_i as $(\mathbf{w}_i, \delta_i)$ and σ_i as (π_i, ct_i). $\mathcal{B}$ obtains $\mathbf{x}_i \in \mathbb{Z}_q^{2m}$, $\mathbf{y}_i \in \{\pm 1\}^{2m}$, and $\mathbf{z}_i \in \mathbb{Z}_q^{2m}$ by decrypting ct_i, where $\mathbf{x}_i \| \mathbf{y}_i \| \mathbf{z}_i = \mathsf{PKE.Dec}(\mathsf{pke.sk}, \mathsf{ct}_i)$. Thus for all $i \in [k]$,

$$\mathsf{H}(\delta_i) = \mathbf{C} \cdot \mathbf{z}_i + \mathbf{B} \cdot \mathbf{y}_i - \mathbf{A} \cdot \mathbf{x}_i = \mathbf{C} \cdot \mathbf{z}_i + \mathbf{B} \cdot \mathbf{y}_i - \mathbf{C} \cdot \mathbf{R} \cdot \mathbf{x}_i = \mathbf{C} \cdot (\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i) + \mathbf{B} \cdot \mathbf{y}_i.$$

Recall that $\|\mathbf{z}_i\| \leq \sqrt{2}\beta$, $\|\mathbf{x}_i\| \leq \sqrt{2}\beta/m$ and since $\mathbf{R} \in \{0,1\}^{2m \times 2m}$ is a binary matrix,

$$\|\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i\| \leq 3\sqrt{2}\beta.$$

Note that $\mu_i \neq \mu_j$, for $1 \leq i < j \leq k$. Then there are two cases:

Case 1: There exist $1 \leq i < j \leq k$, $\delta_i = \delta_j$ and $\mathbf{y}_i = \mathbf{y}_j$. Since $\mathsf{H}(\delta_i) = \mathbf{C} \cdot \mathbf{z}_i + \mathbf{B} \cdot \mathbf{y}_i - \mathbf{A} \cdot \mathbf{x}_i$ we have:

$$\mathbf{C} \cdot (\mathbf{z}_i - \mathbf{z}_j) = \mathbf{A} \cdot (\mathbf{x}_i - \mathbf{x}_j) \ ,$$

Further, for $\mathbf{A} = \mathbf{C} \cdot \mathbf{R}$, this equivalently means

$$\mathbf{z}_i - \mathbf{z}_j = \mathbf{R} \cdot (\mathbf{x}_i - \mathbf{x}_j) \ .$$

We show that this happens with negligible probability.

Claim 7.7. The probability $\Pr\left[\mathbf{z}_i - \mathbf{z}_j = \mathbf{R} \cdot (\mathbf{x}_i - \mathbf{x}_j)\right]$, taken over the adversary's output distribution and the randomness of $\mathbf{R}$, is negligible.

Proof. We have by the leftover hash lemma, that $(\mathbf{C}, \mathbf{A})$ is statistically close to $(\mathbf{C}, \mathbf{C} \cdot \mathbf{R})$. As a result, the distribution of the adversary's outputs $(\mathbf{x}_i, \mathbf{z}_i, \mathbf{x}_j, \mathbf{z}_j)$ where the reduction algorithm $\mathcal{B}$ generates $\mathbf{A}$ uniformly at random is statistically close to the distribution of the adversary's output when reduction algorithm $\mathcal{B}$ generates $\mathbf{A} := \mathbf{C} \cdot \mathbf{R}$. Thus, with all but negligible probability, the adversary's output is statistically independent from the choice of $\mathbf{R}$. So it follows that with all but negligible probability, $\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i \neq \mathbf{z}_j - \mathbf{R} \cdot \mathbf{x}_j$. □

Case 2: $\delta_i \neq \delta_j$. Then with all but negligible probability, we have $\mathsf{H}(\delta_i) \neq \mathsf{H}(\delta_j)$, which implies that $\mathbf{C} \cdot (\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i) + \mathbf{B} \cdot \mathbf{y}_i \neq \mathbf{C} \cdot (\mathbf{z}_j - \mathbf{R} \cdot \mathbf{x}_j) + \mathbf{B} \cdot \mathbf{y}_j$. Thus, either $\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i \neq \mathbf{z}_j - \mathbf{R} \cdot \mathbf{x}_j$, or $\mathbf{y}_i \neq \mathbf{y}_j$.

Case 3: $\delta_i = \delta_j$ and $\mathbf{y}_i \neq \mathbf{y}_j$. Since $\mathbf{y}_i \neq \mathbf{y}_j$, the analysis directly follows in this case.

As a result, $\mathcal{B}$ wins the rOM-ISIS game by sending back $\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i$, $\mathbf{y}_i$, and $H(\delta_i)$ for $i \in [k]$.

□

Note that any successful one-more unforgeability attacker $\mathcal{A}$ must be one of type 1, 2 and 3. Combining the above arguments, the lemma follows. □

This completes the proof of our main theorem. □

7.3 Receiver blindness

Next, we show that this protocol satisfies receiver blindness property.

Theorem 7.8. Assume that NIZK proof system NIZK satisfies zero knowledge property, and public key encryption scheme PKE is IND-CPA secure, then our construction 7.1 is receiver blind.

Proof. We prove the theorem through the following hybrids:

Hybrid_0 This corresponds to the real experiment.

1. Challenger samples $\mathbf{A}, \mathbf{B} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times 2m}$ uniformly at random, $\mathsf{pke.pk} \leftarrow\!\!\$\ \mathsf{PKE.KeyGen}(1^\lambda)$, and $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$. Challenger then sets $\mathsf{pp} := (\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs}, \mathsf{H})$ and sends pp to the adversary.

2. Next, challenger samples $(\mathsf{sk}_{R_b}, \mathsf{pk}_{R_b}) \leftarrow \mathsf{KeyGen}_R(\mathsf{pp})$, for $b \in \{0,1\}$. The challenger sends pk_{R_0} and pk_{R_1} to $\mathcal{A}$.

3. The adversary outputs a matrix $\mathbf{C} \in \mathbb{Z}_q^{n \times 2m}$ as the verification key vk, and issues two presignatures and nonces $(\mathsf{psig}_0, \mathsf{nonce}_0)$ and $(\mathsf{psig}_1, \mathsf{nonce}_1)$.

4. The challenger then generates message and signature pairs:

   (a) For $b \in \{0,1\}$, challenger sets $\mathbf{z}_b$ as psig_b, $\mathbf{y}_b$ as nonce_b, and parses sk_{R_b} as $(\mathbf{x}_b, \delta_b)$. It then checks whether $\mathbf{C} \cdot \mathbf{z}_b + \mathbf{B} \cdot \mathbf{y}_b = \mathbf{A} \cdot \mathbf{x}_b + \mathsf{H}(\delta_b)$ and $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ and $\mathbf{y}_b \in \{\pm 1\}^{2m}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

(b) Otherwise, for $b \in \{0,1\}$, challenger continues to generate $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_b \| \mathbf{y}_b \| \mathbf{z}_b; r_b)$ with freshly sampled randomness r_0 and r_1.

(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_{b,\perp} \\ \mathbf{z}_{b,\perp} \end{bmatrix}$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$, witness ω_b as $(\mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b, r_b)$, and generates proof $\pi_b \leftarrow \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_b, \omega_b)$.

(d) Next, challenger sets message μ_b as $(\mathbf{w}_b, \delta_b)$ and signature σ_b as (π_b, ct_b). Finally, challenger samples $\hat{b} \leftarrow\!\!\$\ \{0,1\}$ uniformly at random and sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to the adversary.

5. Adversary sends a bit b' and wins if $\hat{b} = b'$.

Hybrid_1 Instead of honestly generating the NIZK proofs π_0 and π_1, the challenger simulates π_0 and π_1 without any witness.

4.(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_{b,\perp} \\ \mathbf{z}_{b,\perp} \end{bmatrix}$ and sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$. Without setting any witnesses, challenger generates π_b using NIZK simulator.

Hybrid_2 This is the same as Hybrid_1, except that for $b \in \{0,1\}$, instead of generating $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_b \| \mathbf{y}_b \| \mathbf{z}_b; r_b)$, challenger sets ct_b as $\mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_b)$.

4.(b) Challenger generates $\mathsf{ct}_0 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_0)$ and $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_1)$ with freshly sampled randomness r_0 and r_1.

Hybrid_3 This the same as Hybrid_2, except that the challenger samples pk_{R_0} and pk_{R_1} uniformly at random.

2. When generating pk_{R_b} for $b \in \{0,1\}$, instead of setting it as $\mathbf{A} \cdot \mathbf{x}_b + \mathsf{H}(\delta_b)$, the challenger sets it as a uniformly sampled vector, such that $\mathsf{pk}_{R_b} \leftarrow\!\!\$\ \mathbb{Z}_q^n$.

Hybrid_4 This is the same as Hybrid_3, except that the challenger samples vectors $\mathbf{w}_0, \mathbf{w}_1$ uniformly at random.

4.(c) For $b \in \{0,1\}$, it samples $\mathbf{w}_b$ uniformly at random, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$, and generates π_b using NIZK simulator.

Let $\mathsf{Adv}^j_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in the NIBS blindness game in Hybrid_j. Then, the following must hold:

Lemma 7.9. Assuming zero-knowledge property of NIZK, it holds that for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^0_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. We set the following intermediate step for the proof. In the intermediate step, everything remains the same as Hybrids 0 and 1, except for the following:

4.(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_{b,\perp} \\ \mathbf{z}_{b,\perp} \end{bmatrix}$ and sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$. Challenger only sets witness ω_1 as $(\mathbf{x}_1, \mathbf{y}_1, \mathbf{z}_1, r)$ without assigning the witness for statement x_0. Challenger generates π_0 using NIZK simulator and $\pi_1 \leftarrow \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_1, \omega_1)$.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{0.5}_{\mathcal{A}}$. Suppose there exists a PPT attacker $\mathcal{A}$ such that $|\mathsf{Adv}^{0}_{\mathcal{A}} - \mathsf{Adv}^{0.5}_{\mathcal{A}}| = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks the zero-knowledge property of NIZK with advantage $\epsilon(\lambda)$.

Challenger starts by outputting nizk.crs. Reduction algorithm $\mathcal{B}$ samples $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_q^{n \times 2m}$ and pke.pk, and simply outputs $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs})$. $\mathcal{A}$ then outputs its verification key vk. Next, $\mathcal{B}$ samples $(\mathsf{sk}_{R_b}, \mathsf{pk}_{R_b}) \leftarrow \mathsf{KeyGen}_R(\mathsf{pp})$ for $b \in \{0,1\}$. $\mathcal{B}$ outputs pk_{R_0} and pk_{R_1}. Attacker $\mathcal{A}$ then sends its presignatures and nonces $(\mathsf{psig}_0, \mathsf{nonce}_0)$ and $(\mathsf{psig}_1, \mathsf{nonce}_1)$. Subsequently, $\mathcal{B}$ simulates the Obtain algorithm: For $b \in \{0,1\}$, $\mathcal{B}$ sets $\mathbf{z}_b$ as psig_b, $\mathbf{y}$ as nonce_b, parses sk_{R_b} as $(\mathbf{x}_b, \delta_b)$, and checks if $\mathbf{C} \cdot \mathbf{z}_b + \mathbf{B} \cdot \mathbf{y}_b = \mathbf{A} \cdot \mathbf{x}_b + H(\delta_b)$ and $\mathbf{y}_b \in \{\pm 1\}^{2m}$ and $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$. If any of these checks fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, for $b \in \{0,1\}$, it continues to generate $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_b \| \mathbf{y}_b \| \mathbf{z}_b; r_b)$ with freshly sampled randomness r_b. $\mathcal{B}$ then computes $\mathbf{w}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_{b,\perp} \\ \mathbf{z}_{b,\perp} \end{bmatrix}$ and sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$, witness ω_b as $(\mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b, r_b)$. Next, $\mathcal{B}$ queries the NIZK challenger with (x_0, ω_0) and challenger returns proof π_0. $\mathcal{B}$ generates $\pi_1 = \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_1, \omega_1)$ by itself. Next, $\mathcal{B}$ sets message μ_b as $(\mathbf{w}_b, \delta_b)$ and signature σ_b as (π_b, ct_b) for $b \in \{0,1\}$. $\mathcal{B}$ then samples $\hat{b} \leftarrow \{0,1\}$ uniformly at random. Finally, $\mathcal{B}$ sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to $\mathcal{A}$. $\mathcal{A}$ outputs its guess b'. If $b' = \hat{b}$, $\mathcal{B}$ sends 0 as its guess (i.e., π_0 is a simulated proof), otherwise it sends 1 as its guess (i.e., π_0 is honestly generated from $\mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_0, \omega_0)$).

Note that if the NIZK challenger honestly generates proof the π_0 using $\mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_0, \omega_0)$, then $\mathcal{B}$ perfectly simulates the experiment of Hybrid_0 for adversary $\mathcal{A}$. Otherwise it simulates the intermediate step for $\mathcal{A}$. As a result, if $|\mathsf{Adv}^{0}_{\mathcal{A}} - \mathsf{Adv}^{0.5}_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks the zero-knowledge property of NIZK with non-negligible advantage. Similarly, $|\mathsf{Adv}^{0.5}_{\mathcal{A}} - \mathsf{Adv}^{1}_{\mathcal{A}}|$ is also negligible, and thus $|\mathsf{Adv}^{0}_{\mathcal{A}} - \mathsf{Adv}^{1}_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$. □

Lemma 7.10. If public key encryption scheme PKE is an IND-CPA secure public key encryption scheme, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^{1}_{\mathcal{A}} - \mathsf{Adv}^{2}_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Consider the following intermediate step between Hybrids 1 and 2. In the intermediate step, everything remains the same in Hybrids 1 and 2, except for the following:

4.(b) $\mathcal{B}$ generates $\mathsf{ct}_0 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_0)$ and $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_1 \| \mathbf{y}_1 \| \mathbf{z}_1; r_1)$ with freshly sampled randomness r_0 and r_1.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{1.5}_{\mathcal{A}}$. Suppose there exists a PPT attacker $\mathcal{A}$ such that $|\mathsf{Adv}^{1}_{\mathcal{A}} - \mathsf{Adv}^{1.5}_{\mathcal{A}}| = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks the security of PKE with advantage $\epsilon(\lambda)$.

Challenger first samples and outputs pke.pk from PKE.KeyGen. Reduction algorithm $\mathcal{B}$ then samples $\mathbf{A}, \mathbf{B} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times 2m}$ and $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$, and outputs pp as $(\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs})$. Next, $\mathcal{A}$ outputs its verification key vk. For $b \in \{0,1\}$, $\mathcal{B}$ randomly samples key pairs $(\mathsf{sk}_{R_b}, \mathsf{pk}_{R_b}) \leftarrow \mathsf{KeyGen}_R(1^\lambda, \mathsf{pp})$. $\mathcal{B}$ sends pk_{R_0} and pk_{R_1} to $\mathcal{A}$. Adversary $\mathcal{A}$ then sends its presignatures and nonces $(\mathsf{psig}_0, \mathsf{nonce}_0)$ and $(\mathsf{psig}_1, \mathsf{nonce}_1)$. Subsequently, for $b \in \{0,1\}$, $\mathcal{B}$ sets $\mathbf{z}_b$ as psig_b, $\mathbf{y}_b$ as nonce_b, and parses sk_{R_b} as $(\mathbf{x}_b, \delta_b)$. $\mathcal{B}$ checks if $\mathbf{C} \cdot \mathbf{z}_b + \mathbf{B} \cdot \mathbf{y}_b = \mathbf{A} \cdot \mathbf{x}_b + H(\delta_b)$ and if $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ for $b \in \{0,1\}$. If any of the checks fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, it queries the challenger with the all zeros string $\mathbf{0}$ and the string $\mathbf{x}_0 \| \mathbf{y}_0 \| \mathbf{z}_0$. Challenger returns ciphertext ct_0 (which is an encryption of either $\mathbf{0}$ or $\mathbf{x}_0 \| \mathbf{z}_0$). Then $\mathcal{B}$ generates $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_1 \| \mathbf{y}_1 \| \mathbf{z}_1; r)$ with freshly

sampled randomness r. It computes $\mathbf{w}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_{b,\perp} \\ \mathbf{z}_{b,\perp} \end{bmatrix}$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$, and generates π_b as a simulated proof for $b \in \{0,1\}$. Next, $\mathcal{B}$ sets message μ_b as $(\mathbf{w}_b, \delta_b)$ and signature σ_b as (π_b, ct_b) for $b \in \{0,1\}$. Finally, $\mathcal{B}$ samples $\hat{b} \leftarrow \{0,1\}$ uniformly at random and sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to $\mathcal{A}$. $\mathcal{A}$ outputs its guess b'. If $b' = \hat{b}$, $\mathcal{B}$ sends 0 as its guess (i.e., ct_0 is an encryption of $\mathbf{0}$), otherwise it sends 1 as its guess (i.e., ct_0 is an encryption of string $\mathbf{x}_0 \| \mathbf{z}_0$).

Note that if ct_0 is an encryption of string $\mathbf{x}_0 \| \mathbf{z}_0$, then $\mathcal{B}$ perfectly simulates the experiment of Hybrid_1 for adversary $\mathcal{A}$. Otherwise it simulates the intermediate step. As a result, if $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^{1.5}_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks the security of public key encryption scheme PKE with non-negligible advantage. Similarly, $|\mathsf{Adv}^{1.5}_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$ and thus $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$. □

Lemma 7.11. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Since $\mathbf{x}_\top \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}$, we have $\mathbf{H}_\infty(\mathbf{x}_{0,\top}), \mathbf{H}_\infty(\mathbf{x}_{1,\top}) \geq m$. As $m = n \log q + \lambda$, it follows from Leftover Hash Lemma (**Corollary** 3.8) that the following distributions are statistically indistinguishable:

$$\begin{aligned} &\{(\mathbf{A}_\mathsf{L}, \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_{0,\top}, \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_{1,\top}) : \mathbf{A}_\mathsf{L} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}, \mathbf{x}_{0,\top} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}, \mathbf{x}_{1,\top} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}\} \\ &\qquad \approx_s \{(\mathbf{A}_\mathsf{L}, \boldsymbol{\rho_0}, \boldsymbol{\rho_1}) : \mathbf{A}_\mathsf{L} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}, \boldsymbol{\rho_0} \leftarrow\!\$\ \mathbb{Z}_q^n, \boldsymbol{\rho_1} \leftarrow\!\$\ \mathbb{Z}_q^n\} \end{aligned}$$

Thus for $b \in \{0,1\}$, vector $\mathbf{A} \cdot \mathbf{x}_b + \mathsf{H}(\delta_b) = \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_{b,\top} + \mathbf{A}_\mathsf{R} \cdot \mathbf{x}_{b,\perp} + \mathsf{H}(\delta_b)$ is statistically indistinguishable from a random vector in $\mathbb{Z}_q^n$. □

Lemma 7.12. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Note that $\mathbf{x}_{0,\perp}, \mathbf{x}_{1,\perp} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}$, we have $\mathbf{H}_\infty(\mathbf{x}_{0,\perp}), \mathbf{H}_\infty(\mathbf{x}_{1,\perp}) \geq m$. As $m = n \log q + \lambda$. Again, it follows from Leftover Hash Lemma (**Corollary** 3.8) that:

$$\begin{aligned} &\{(\mathbf{A}_\mathsf{L}, \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_{0,\perp}, \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_{1,\perp}) : \mathbf{A}_\mathsf{L} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}, \mathbf{x}_{0,\perp} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}, \mathbf{x}_{1,\perp} \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}\} \\ &\qquad \approx_s \{(\mathbf{A}_\mathsf{L}, \boldsymbol{\rho_0}, \boldsymbol{\rho_1}) : \mathbf{A}_\mathsf{L} \leftarrow\!\$\ \mathbb{Z}_q^{n \times m}, \boldsymbol{\rho_0} \leftarrow\!\$\ \mathbb{Z}_q^n, \boldsymbol{\rho_1} \leftarrow\!\$\ \mathbb{Z}_q^n\} \end{aligned}$$

As a result, for $b \in \{0,1\}$ and any $\boldsymbol{\mu} \in \mathbb{Z}_q^n$, vector $\mathbf{w}_b = \boldsymbol{\mu} + \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_{b,\perp}$ is statistically indistinguishable from a random vector in $\mathbb{Z}_q^n$. □

Finally, note that any adversary has 0 advantage in Hybrid_4. From the above lemmas, it follows that the above NIBS construction satisfies recipient blindness. □

7.4 Efficiency of lattice-based NIBS

We use the same basic building blocks as Agrawal et al. [AKSY22] to instantiate the protocol in Construction 7.1. For each block, we identify the similarities to their scheme and then provide details on key differences in, both, our specific instantiation as well as the choice of parameters. We appropriately modify the Python script in [AKSY22] to obtain parameter estimates for our scheme.

Trapdoor generation and preimage sampling. We instantiate Falcon-512 [FHK+17] over the ring $\mathcal{R}_{512} = \mathbb{Z}[x]/(q_\mathsf{F}, x^{512} + 1)$. The Falcon modulus, q_F is chosen to be a prime greater than $2^{12.8}$, that is congruent to 5 (mod 8) in order to simplify the linear relations proven in the zero-knowledge

proof later. Although the smallest such prime, 7213, would be the natural choice, it turns out that to satisfy certain bounds on q_{PKE}, a higher prime is needed. We set $q_F = 7349$ and carry over the other Falcon parameters $n_F = 512$, $m_F = 1024, \varsigma$ and $\beta = 1.1 \cdot \varsigma \cdot \sqrt{m_F}$ from [AKSY22].

An important distinction in our construction is the use of the Bonsai trick [CHKP10] to generate a short preimage for $\mathbf{t} - \mathbf{b} \cdot \mathbf{y}_1 + \mathbf{y}_2$, where the receiver's public key $\mathbf{t}$, the public polynomial $\mathbf{b}$, and the signer generated $\mathbf{y}_1$ and $\mathbf{y}_2$ are all polynomials in $\mathcal{R}_{512}$. For NTRU lattices, the signer computes this preimage by sampling polynomials $\mathbf{y}_1, \mathbf{y}_2, \mathbf{z}_1', \mathbf{z}_2' \in \mathcal{R}_{512}$ such that $\left\|\left[\mathbf{z}_1' \mid \mathbf{z}_2'\right]\right\|$ is small, and the coefficients of $\mathbf{y}_1$ and $\mathbf{y}_2$ are in $\{\pm 1\}$. It then uses its NTRU secret trapdoor to sample short polynomials $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{R}_{512}$ such that

$$\mathbf{c} \cdot \mathbf{z}_1 + \mathbf{z}_2 = \mathbf{t} - \mathbf{b} \cdot \mathbf{y}_1 - \mathbf{y}_2 - \mathbf{c} \cdot \mathbf{z}_1' - \mathbf{z}_2'$$

The presignature is then $(\mathbf{z}_1, \mathbf{z}_1', \mathbf{z}_2')$ and the nonce is $(\mathbf{y}_1, \mathbf{y}_2)$. Given this, the Receiver can recover $\mathbf{z}_2$. We require that $\left\|\left[\mathbf{z}_1' \mid \mathbf{z}_2'\right]\right\|_\infty \leq 2$ so as to extract short preimages in the randomized one-more ISIS game. Using this, we estimate the size of the presignature to be ≈ 1 KB. We note that the Receiver's first message (here the public key value $\mathbf{t}$) is computed identically to [AKSY22], i.e.,

$$\mathbf{t} = \mathbf{a}' \cdot \mathbf{x}_1 + \mathbf{x}_2 + \mathsf{H}(\delta), \tag{2}$$

where polynomial $\mathbf{a}'$ and hash function H are specified in the protocol's public parameters, and $\mathbf{x}_1, \mathbf{x}_2$ are sampled from $\mathcal{R}_{512}$ such that $\left\|\left[\mathbf{x}_1 \mid \mathbf{x}_2\right]\right\|_\infty \leq 2$ and $\delta \leftarrow\!\!\$\, \{0,1\}^\lambda$. The Receiver's secret key is $\left(\left[\mathbf{x}_1 \mid \mathbf{x}_2\right], \delta\right)$.

IND-CPA **secure** PKE.The Regev style public key encryption scheme from [LPS10] is instantiated over $\mathcal{R}_{128} = \mathbb{Z}[x]/(q_{\mathsf{PKE}}, x^{128} + 1)$. Here, we chose $q_{\mathsf{PKE}} = q_F \cdot 8933 = 65648617$ and the integer $p = 61583$ under the same constraints as listed in [AKSY22]. In our case, the Receiver must encrypt $(\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T} \mid \mathbf{y}_1^\mathsf{T} \mid \mathbf{z}_1^\mathsf{T} \mid \mathbf{z}_2^\mathsf{T} \mid \mathbf{z}_1'^\mathsf{T} \mid \mathbf{z}_2'^\mathsf{T}) \in (\mathcal{R}_{128}^4)^7 \cong \mathcal{R}_{128}^{28}$. Concretely, we must encrypt four additional polynomials over [AKSY22], which results in a ciphertext of 14.6 KB. Reducing the size of this ciphertext is left as an open problem.

Zero-knowledge proof. Subject to the instantiation listed above, the receiver must prove knowledge of polynomials $\mathbf{x}_1, \mathbf{x}_2, \mathbf{y}_1, \mathbf{y}_2, \mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_1', \mathbf{z}_2' \in \mathcal{R}_{512}$ satisfying

$$\mathbf{c} \cdot \mathbf{z}_1 + \mathbf{z}_2 + \mathbf{c} \cdot \mathbf{z}_1' + \mathbf{z}_2' + \mathbf{b} \cdot \mathbf{y}_1 + \mathbf{y}_2 - \mathbf{a} \cdot \mathbf{x}_1 - \mathbf{x}_2 = \mathsf{H}(\delta)$$

and the coefficients of $\mathbf{y}_1$ and $\mathbf{y}_2$ are in $\{\pm 1\}$[15]. The receiver must also prove that part of the message, $\mathbf{w} = \mathbf{a} \cdot \mathbf{z}_1 + \mathbf{x}_2$. It must further prove that it knows encryption randomness $\mathbf{s}, \mathbf{e}_1 \in \mathcal{R}_{128}^8$ and $\mathbf{e}_2 \in \mathcal{R}_{128}^{28}$ such that the ciphertext ct is an encryption of $(\mathbf{x}_1^\mathsf{T} \mid \mathbf{x}_2^\mathsf{T} \mid \mathbf{y}_1^\mathsf{T} \mid \mathbf{z}_1^\mathsf{T} \mid \mathbf{z}_2^\mathsf{T} \mid \mathbf{z}_1'^\mathsf{T} \mid \mathbf{z}_2'^\mathsf{T})$ under pke.pk. Lastly, it needs to prove that the following bounds hold

$$\text{(i)} \left\|\left[\mathbf{z}_1 \mid \mathbf{z}_2\right]\right\|_2 \leq \beta \qquad \text{(ii)} \left\|\left[\mathbf{z}_1' \mid \mathbf{z}_2'\right]\right\|_2 \leq 2 \cdot \sqrt{2 \cdot 512}$$
$$\text{(iii)} \left\|\left[\mathbf{x}_1 \mid \mathbf{x}_2\right]\right\|_2 \leq 2 \cdot \sqrt{2 \cdot 512} \qquad \text{(iv)} \left\|\left[\mathbf{s} \mid \mathbf{e}_1 \mid \mathbf{e}_2\right]\right\|_2 \leq \tau \cdot \sqrt{(2 \cdot 8 + 28) \cdot 128}$$

[15] For each $\mathbf{y}_i$, we must prove the linear relation $\mathbf{y}_i = 2\mathbf{y}_i' - \mathbf{1}$ and the quadratic relation $\mathbf{y}_i' \cdot (\mathbf{y}_i' - \mathbf{1}) = \mathbf{0}$.

Since all our relations are linear or proving norm bounds, we are able to leverage the efficient zero-knowledge proof framework of [LNP22b]. For specific details concerning the implementation, we direct the reader to confer [LNP22b] and remark only that we must prove just one extra norm bound (and two additional quadratic relations for coefficients of $\mathbf{y}_i$) over [AKSY22] and the one extra relation pertaining the message $\mathbf{w}$ can be proven essentially for free. The proof can then be further compressed using [DKL+18], in effect cutting low-order bits of the commitment.

The process for selecting the parameters is the same as in [AKSY22] so we only highlight the main differences. We set the modulus $q_{\mathsf{ZK}} = q_{\mathsf{F}} \cdot q_{\mathsf{PKE}} \cdot 99901 = 6558362486917 \approx 2^{42.5}$, we chose $m_2 = 35$ so that Module-LWE problem underlying the zero-knowledge protocol is hard, and the length of the committed vector over $\mathcal{R}_{128}$ is

$$m_1 = \underbrace{10 \cdot 4}_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{y}_1,\mathbf{y}_2,\mathbf{y}'_1,\mathbf{y}'_2,\mathbf{z}_1,\mathbf{z}_2,\mathbf{z}'_1,\mathbf{z}'_2} + \underbrace{2 \cdot 8}_{\mathbf{s},\mathbf{e}_1} + \underbrace{28}_{\mathbf{e}_2} + \underbrace{4}_{\ell_2\text{-norms}} + \underbrace{2}_{\mathbf{y}'_i \text{ coefficients}} = 90 \quad .$$

The final proof size after the cut is 53.1 KB and the signature is therefore 67.7 KB. We believe that further reduction in size may be possible using the recent optimizations given in [LN22] although we do not attempt to do so here.

Concrete security. We also estimate the concrete security of the given instantiation using the script. Our observations are summarized in Table 3.

Security Property	Assumption	Bit security (Core-SVP Hardness)
Key recovery (Falcon)	NTRU	135 bits
Unforgeability (Falcon)	MSIS	121 bits
Hiding for pk_R and $\mathbf{w}$	MLWE	122 bits
Secret key and ciphertext security (Encryption)	MLWE	117 bits
Zero-knowledge proof	MSIS and MLWE	108 and 120 bits (resp.)

Table 3: Overall security of our protocol.

8 Lattice-based NIBS with Nonce Blindness

The NIBS scheme from Section 7 can be upgraded to satisfy nonce blindness with a slight modification. As hinted at in the technical overview, the main idea is to "smudge-out" the presignature information from the signed message by using a large noise that was committed in the receiver's verification key. We elaborate on this below.

Let parameters $n, m, \varsigma, \beta = \varsigma\sqrt{m}$ and prime q be such that the randomized one-more ISIS instance $\mathsf{rOM\text{-}ISIS}_{q,n,2m,\varsigma,\beta'}$ is hard for $\beta' = (2^{\lambda/2} + 2\sqrt{2} + 2) \cdot \beta$. These parameters must satisfy the constraints in (1).

Tools required. The construction relies on a secure hash function H (modeled as a random oracle), a public key encryption scheme $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, a lattice trapdoor $\mathsf{bLT} = (\mathsf{TrapGen}, \mathsf{SamplePre})$, and a NIZK scheme $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ for the following language:

Language $\mathcal{L}_3$

Instance: Each instance x is interpreted as a matrix $\mathbf{C}$ with a secret trapdoor, random matrices $\mathbf{A}$ and $\mathbf{B}$, PKE public key pke.pk, ciphertext ct, vector $\mathbf{w}$ and random string δ.

Witness: Witness ω consists of a secret vector $\mathbf{x}$, nonce vector $\mathbf{y}$, presignature vector $\mathbf{z}$, a bit θ, and randomness r.

Membership: ω is a valid witness for x if the following are satisfied:

- $\|\mathbf{z}\| \leq \sqrt{2}\beta$ and $\|\mathbf{x}\| \leq \left(1 + 2^{(\lambda/2-1)}\right) \cdot \beta/m$ and $\mathbf{y} \in \{\pm 1\}^{2m}$.
- ct is an encryption of $\mathbf{x}\|\mathbf{y}\|\mathbf{z}\|\theta$ using randomness r, s.t. $\mathsf{ct} = \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}\|\mathbf{z}\|\theta; r)$
- $\mathbf{C} \cdot \mathbf{z} + \mathbf{B} \cdot \mathbf{y} = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + \mathsf{H}(\delta)$, $(1 - 2\theta) \cdot \mathbf{w} = \mathbf{z}_\perp - \mathbf{x}_\perp$, and $\theta(1-\theta) = 0$

8.1 Construction

Below we describe our non-interactive blind signature scheme.

$\mathsf{Setup}(1^\lambda, n, m, q, \varsigma, \mathsf{H}) \to \mathsf{pp}$. Instead of a matrix in $\mathbb{Z}_q^{n \times 2m}$, it samples the matrix $\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{n \times (2m+1)}$. Then, acting exactly as the Setup algorithm for Construction 7.1, it outputs the public parameters as $\mathsf{pp} := (\mathbf{A}, \mathsf{pke.pk}, \mathsf{nizk.crs}, n, m, q, \varsigma, \mathsf{H})$.

$\mathsf{KeyGen}_S(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{vk})$. Acting exactly as the KeyGen_S algorithm for Construction 7.1, it outputs signer's secret key and verification key as $\mathsf{sk} := \mathbf{T}_\mathbf{C}$ and $\mathsf{vk} := \mathbf{C}$ respectively.

$\mathsf{KeyGen}_R(\mathsf{pp}) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. It samples the two halves of $\mathbf{x} \in \mathbb{Z}_q^{2m}$ as $\mathbf{x}_\top \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}$, and $\mathbf{x}_\perp \leftarrow\!\$\ \left[-\frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta, \frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta\right]^m$, $\delta \leftarrow\!\$\ \{0,1\}^\lambda$ as well as a bit $\theta \leftarrow\!\$\ \{0,1\}$. Using these, where it previously computed $\mathbf{t}$ as $\mathbf{A} \cdot \mathbf{x} + \mathsf{H}(\delta)$, it computes $\mathbf{t} = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + \mathsf{H}(\delta)$ and outputs user's secret key and public key as $\mathsf{sk}_R := (\mathbf{x}, \delta, \theta)$ and $\mathsf{pk}_R := \mathbf{t}$ respectively.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \to (\mathsf{psig}, \mathsf{nonce})$. Identically to the Issue algorithm of Construction 7.1, it generates $\mathbf{z} \leftarrow \mathsf{bLT.SamplePre}(\mathbf{C}, \mathbf{T}_\mathbf{C}, \mathbf{t} - \mathbf{B} \cdot \mathbf{y}, \varsigma)$ for receiver's public key given by $\mathbf{t}$, and outputs the presignature $\mathsf{psig} := \mathbf{z}$, $\mathsf{nonce} := \mathbf{y}$.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, \mathsf{psig}, \mathsf{nonce}) \to (\mu, \sigma)$. It parses sk_R as $(\mathbf{x}, \theta, \delta)$. Then, for $\mathbf{C} := \mathsf{vk}$, $\mathbf{z} := \mathsf{psig}$, and $\mathbf{y} := \mathsf{nonce}$, it checks if $\mathbf{C} \cdot \mathbf{z} + \mathbf{B} \cdot \mathbf{y} = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + \mathsf{H}(\delta)$ and $\|\mathbf{z}\| \leq \sqrt{2}\beta$ and $\mathbf{y} \in \{\pm 1\}^{2m}$. If any check fails it aborts and outputs $\perp$.

Otherwise, it creates the ciphertext $\mathsf{ct} \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}\|\mathbf{z}\|\theta; r)$ using uniformly sampled randomness $r \leftarrow\!\$\ \{0,1\}^\lambda$. The algorithm then, where in Construction 7.1 it set $\mathbf{w}$ as $\mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_\perp \\ \mathbf{z}_\perp \end{bmatrix}$, it now sets $\mathbf{w} = (1-2\theta)(\mathbf{z}_\perp - \mathbf{x}_\perp)$. It generates NIZK proof π as $\pi \leftarrow \mathsf{NIZK.Prove}\left(\mathsf{nizk.crs}, x := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}, \mathbf{w}, \delta), \omega := (\mathbf{x}, \mathbf{y}, \mathbf{z}, \theta, r)\right)$. Finally, it outputs message $\mu := (\mathbf{w}, \delta)$ and signature $\sigma := (\pi, \mathsf{ct})$.

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0,1\}$. It parses μ as $(\mathbf{w}, \delta)$ and σ as (π, ct). The verification algorithm accepts and outputs 1 if and only if $\mathsf{NIZK.Verify}\left(\mathsf{nizk.crs}, x := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}, \mathbf{w}, \delta), \pi\right) = 1$. Otherwise, it outputs 0.

Efficiency. We estimate that the resulting proof size for an instantiation similar to the one given in Section 7.4 is about a factor of λ larger. A concrete analysis is left for the full version. We also believe that it may be possible to achieve smudging bounds using a similar technique as applied in [BdMW16] although we do not attempt to do so here, leaving it instead as a possible future direction.

Fully blind TNIBS. We remark that a fully blind TNIBS protocol can be obtained from Construction 8.1 in the same way as the receiver blind TNIBS construction C.2 was obtained from NIBS construction 7.1. Concretely, it requires a second hash function, H_2 that the signer's Issue algorithm uses to set $\mathbf{t} = \mathsf{pk}_R + \mathsf{H}_2(\tau)$ given tag τ as input. The signer then sends the tag along with the presignature and the nonce to the receiver, whose Obtain algorithm must now also include the tag τ in the NIZK instance when producing the proof π.

Completeness. As before, we have from the correctness of $\mathsf{bLT.SamplePre}$ that $\mathbf{z}$ is a presignature on $\mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + \mathsf{H}(\delta)$ such that $\|\mathbf{z}\| \leq \sqrt{2}\beta$. Furthermore, we have by construction that $\mathbf{w} = (1-2\theta)(\mathbf{z}_\perp - \mathbf{x}_\perp)$, ct is an encryption of $(\mathbf{x}\|\mathbf{z}\|\theta)$ under public key pke.pk and randomness r, and that $\|\mathbf{x}\| \leq \left(1 + 2^{(\lambda/2-1)}\right) \cdot \beta/m$. Therefore, π is a valid NIZK proof for the language $\mathcal{L}_3$ described above. It follows from the completeness of the underlying NIZK that Construction 8.1 is complete.

Reusability. Notice that for $b \in \{0,1\}$, if a signer issues presignature-nonce pairs $(\mathbf{z}_b, \mathbf{y}_b)$ to a given receiver with secret vector $\mathbf{x}$ and secret bit θ then,

$$\begin{aligned} \Pr[\mathbf{y}_0 = \mathbf{y}_1 \vee \mu_0 = \mu_1] = &\ \Pr[\mu_0 = \mu_1] + \Pr[\mathbf{y}_0 = \mathbf{y}_1] \\ = &\ \Pr[(\mathbf{w}_0, \delta) = (\mathbf{w}_1, \delta)] + \Pr[\mathbf{y}_0 = \mathbf{y}_1] \\ = &\ \Pr\left[(1-2\theta)(\mathbf{z}_{0,\perp} - \mathbf{z}_{1,\perp}) = \mathbf{0}\right] + \mathsf{negl}(\lambda) \ . \end{aligned}$$

Since, $\theta \in \{0,1\}$, this is simply the probability that $\mathbf{z}_{0,\perp} = \mathbf{z}_{1,\perp}$, which is negligible in λ by construction. Thus the construction is reusable.

8.2 Nonce blindness

We begin by proving nonce blindness of this construction.

Theorem 8.1. Assume that NIZK proof system NIZK satsifies zero-knowledge property, and public key encryption scheme PKE is IND-CPA secure, then the NIBS construction 8.1 is nonce blind.

Proof. We have the following hybrids:

Hybrid_0 This corresponds to the real experiment.

1. Challenger samples $\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{n \times (2m+1)}$, $\mathbf{B} \leftarrow\!\$\ \mathbb{Z}_q^{n \times 2m}$ uniformly at random, $\mathsf{pke.pk} \leftarrow\!\$\ \mathsf{PKE.KeyGen}(1^\lambda)$, and $\mathsf{nizk.crs} \leftarrow\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$. Challenger then sets pp as $(\mathbf{A}, \mathsf{pke.pk}, \mathsf{nizk.crs})$ and sends pp to the adversary.

2. Next, challenger samples $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{KeyGen}_R(\mathsf{pp})$ and sends pk_R to $\mathcal{A}$.

3. The adversary sends a matrix $\mathbf{C} \in \mathbb{Z}_q^{n \times 2m}$ as the verification key, and issues two presignatures and nonces $(\mathsf{psig}_0, \mathsf{nonce}_0)$ and $(\mathsf{psig}_1, \mathsf{nonce}_1)$.

4. The challenger then generates message and signature pairs as follows:

   (a) For $b \in \{0,1\}$, challenger sets $\mathbf{z}_b$ as psig_b, $\mathbf{y}_b$ as nonce_b, and parses sk_R as $(\mathbf{x}, \delta)$. Then it checks whether $\mathbf{C} \cdot \mathbf{z}_b + \mathbf{B} \cdot \mathbf{y}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + H(\delta)$ and $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ and $\mathbf{y}_b \in \{\pm 1\}^{2m}$. If the any of the checks fails, challenger outputs $\perp$ and aborts.

   (b) Otherwise, for $b \in \{0,1\}$, challenger generates $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x} \| \mathbf{y}_b \| \mathbf{z}_b \| \theta; r_b)$ with freshly sampled randomness r_0 and r_1.

   (c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = (1 - 2\theta)(\mathbf{z}_{b,\perp} - \mathbf{x}_\perp)$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta)$, witness ω_b as $(\mathbf{x}, \mathbf{y}_b, \mathbf{z}_b, \theta, r_b)$, and generates NIZK proof $\pi_b \leftarrow \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_b, \omega_b)$.

   (d) Next, challenger sets message μ_b as $(\mathbf{w}_b, \delta)$ and signature σ_b as (π_b, ct_b). Finally, challenger samples $\hat{b} \leftarrow\!\!\$\ \{0,1\}$ uniformly at random and sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to the adversary.

5. Adversary sends a bit b' and wins if $\hat{b} = b'$.

Hybrid_1 This is the same as Hybrid_0, except that instead of generating the NIZK proofs π_0 and π_1, the challenger simulates it without any witnesses.

4.(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = (1 - 2\theta)(\mathbf{z}_{b,\perp} - \mathbf{x}_\perp)$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$. Without setting any witnesses, challenger generates π_b using NIZK simulator.

Hybrid_2 This is the same as Hybrid_1, except that for $b \in \{0,1\}$, instead of generating $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x} \| \mathbf{y}_b \| \mathbf{z}_b \| \theta; r_b)$, challenger sets ct_b as $\mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_b)$,

4.(b) Challenger generates $\mathsf{ct}_0 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_0)$ and $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_1)$ with freshly sampled randomness r_0 and r_1.

Hybrid_3 This the same as Hybrid_2, except that the challenger samples pk_R uniformly at random.

2. When generating pk_R, instead of setting it as $\mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + \mathsf{H}(\delta)$, the challenger sets it as a uniformly sampled vector, such that $\mathsf{pk}_R \leftarrow\!\!\$\ \mathbb{Z}_q^n$.

let $\mathsf{Adv}_{\mathcal{A}}^j$ denote the security advantage of an adversary $\mathcal{A}$ in the NIBS blindness game in Hybrid_j. Then, the following must hold:

Lemma 8.2. Assuming zero-knowledge property of NIZK, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}_{\mathcal{A}}^0 - \mathsf{Adv}_{\mathcal{A}}^1| \leq \mathsf{negl}(\lambda)$.

Proof. We set the following intermediate step for the proof. In the intermediate step, everything remains unchanged in Hybrid_0 and Hybrid_1, except for the following:

4.(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = (1-2\theta)(\mathbf{z}_{b,\perp} - \mathbf{x}_\perp)$ and sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta)$. Challenger only sets witness ω_1 as $(\mathbf{x}, \mathbf{y}_1, \mathbf{z}_1, \theta, r_1)$ without assigning the witness for statement x_0. Challenger generates π_0 using NIZK simulator and $\pi_1 \leftarrow \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_1, \omega_1)$.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{0.5}_{\mathcal{A}}$. Suppose there exists a PPT attacker $\mathcal{A}$ such that $|\mathsf{Adv}^{0}_{\mathcal{A}} - \mathsf{Adv}^{0.5}_{\mathcal{A}}| = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks the zero-knowledge property of NIZK with advantage $\epsilon(\lambda)$.

Challenger starts by outputting nizk.crs. Reduction algorithm $\mathcal{B}$ samples $\mathbf{A} \in \mathbb{Z}_q^{n\times(2m+1)}$, $\mathbf{B} \in \mathbb{Z}_q^{n\times 2m}$ and pke.pk, and simply outputs $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs})$. $\mathcal{A}$ then outputs its verification key vk. Next, $\mathcal{B}$ samples $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{KeyGen}_R(\mathsf{pp})$ and outputs pk_R. Attacker $\mathcal{A}$ then sends two presignatures and nonces $(\mathsf{psig}_0, \mathsf{nonce}_0)$ and $(\mathsf{psig}_1, \mathsf{nonce}_1)$. Subsequently, $\mathcal{B}$ simulates the Obtain algorithm: for $b \in \{0,1\}$, $\mathcal{B}$ sets $\mathbf{z}_b$ as psig_b, $\mathbf{y}_b$ as nonce_b, parses sk_R as $(\mathbf{x}, \delta)$, and checks if $\mathbf{C}\cdot\mathbf{z}_b + \mathbf{B}\cdot\mathbf{y}_b = \mathbf{A}\cdot\begin{bmatrix}\mathbf{x}\\ \theta\end{bmatrix} + \mathsf{H}(\delta)$ and $\|\mathbf{z}_b\| \le \sqrt{2}\beta$ and $\mathbf{y}_b \in \{\pm 1\}^{2m}$. If any of these checks fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, for $b \in \{0,1\}$, it continues to generate $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}_b\|\mathbf{z}_b\|\theta; r_b)$ with freshly sampled randomness r_b. $\mathcal{B}$ then computes $\mathbf{w}_b = (1-2\theta)(\mathbf{z}_{b,\perp} - \mathbf{x}_\perp)$ and sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta)$, witness ω_b as $(\mathbf{x}, \mathbf{y}_b, \mathbf{z}_b, \theta, r_b)$. Next, $\mathcal{B}$ queries the NIZK challenger with (x_0, ω_0) and challenger returns proof π_0. $\mathcal{B}$ generates $\pi_1 = \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_1, \omega_1)$ by itself. Next, $\mathcal{B}$ sets message μ_b as $(\mathbf{w}_b, \delta)$ and signature σ_b as (π_b, ct_b) for $b \in \{0,1\}$. $\mathcal{B}$ then samples $\hat{b} \leftarrow \{0,1\}$ uniformly at random. Finally, $\mathcal{B}$ sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to $\mathcal{A}$. $\mathcal{A}$ outputs its guess b'. If $b' = \hat{b}$, $\mathcal{B}$ sends 0 as its guess (i.e., π_0 is a simulated proof), otherwise it sends 1 as its guess (i.e., π_0 is honestly generated from $\mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_0, \omega_0)$).

Notice, if the NIZK challenger honestly generates proof π_0 using $\mathsf{NIZK.Prove}$ $(\mathsf{nizk.crs}, x_0, \omega_0)$, then $\mathcal{B}$ perfectly simulates the experiment of Hybrid_0 for adversary $\mathcal{A}$. Otherwise it simulates the intermediate step for $\mathcal{A}$. As a result, if $|\mathsf{Adv}^{0}_{\mathcal{A}} - \mathsf{Adv}^{0.5}_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks the zero-knowledge property of NIZK with non-negligible advantage. Similarly, $|\mathsf{Adv}^{0.5}_{\mathcal{A}} - \mathsf{Adv}^{1}_{\mathcal{A}}|$ is also negligible, and thus $|\mathsf{Adv}^{0}_{\mathcal{A}} - \mathsf{Adv}^{1}_{\mathcal{A}}| \le \mathsf{negl}(\lambda)$. □

Lemma 8.3. If public key encryption scheme PKE is a secure public key encryption scheme, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^{1}_{\mathcal{A}} - \mathsf{Adv}^{2}_{\mathcal{A}}| \le \mathsf{negl}(\lambda)$.

Proof. Consider the following intermediate step for the proof. In the intermediate step, everything remains unchanged in Hybrid_1 and Hybrid_2, except for the following:

4.(b) $\mathcal{B}$ generates $\mathsf{ct}_0 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_0)$ and $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}_1\|\mathbf{z}_1\|\theta; r_1)$ with freshly sampled randomness r_0 and r_1.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{1.5}_{\mathcal{A}}$. Suppose there exists a PPT attacker $\mathcal{A}$ such that $|\mathsf{Adv}^{1}_{\mathcal{A}} - \mathsf{Adv}^{1.5}_{\mathcal{A}}| = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks the security of PKE with advantage $\epsilon(\lambda)$.

Challenger first samples and outputs pke.pk from PKE.KeyGen. Reduction algorithm $\mathcal{B}$ then samples $\mathbf{A} \in \mathbb{Z}_q^{n\times(2m+1)}$ and $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$, and outputs pp as $(\mathbf{A}, \mathsf{pke.pk}, \mathsf{nizk.crs})$. Next, $\mathcal{A}$ outputs its verification key vk. $\mathcal{B}$ randomly samples a key pair $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow \mathsf{KeyGen}_R(1^\lambda, \mathsf{pp})$ and sends pk_R to $\mathcal{A}$. Adversary $\mathcal{A}$ then sends two presignatures and nonces $(\mathsf{psig}_0, \mathsf{nonce}_0)$ and $(\mathsf{psig}_1, \mathsf{nonce}_1)$. Subsequently, for $b \in \{0,1\}$, $\mathcal{B}$ sets $\mathbf{z}_b$ as psig_b $\mathbf{y}_b$ as nonce_b, and, after parsing

sk_R as $(\mathbf{x}, \delta)$, it checks if $\mathbf{C} \cdot \mathbf{z}_b + \mathbf{B} \cdot \mathbf{y}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + \mathsf{H}(\delta_b)$ and if $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ for $b \in \{0,1\}$. If any of the checks fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, it queries the challenger with the all zeros string $\mathbf{0}$ and string $\mathbf{x}\|\mathbf{y}_0\|\mathbf{z}_0\|\theta$. Challenger returns ciphertext ct_0 (which is an encryption of either $\mathbf{0}$ or $\mathbf{x}\|\mathbf{y}_0\|\mathbf{z}_0\|\theta$). Then $\mathcal{B}$ generates $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}_1\|\mathbf{z}_1\|\theta; r_1)$ with freshly sampled randomness r. It computes $\mathbf{w}_b = (1-2\theta)(\mathbf{z}_{b,\perp} - \mathbf{x}_\perp)$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta)$, and generates π_b as a simulated proof for each $b \in \{0,1\}$. Next, $\mathcal{B}$ sets message μ_b as $(\mathbf{w}_b, \delta)$ and signature σ_b as (π_b, ct_b) for $b \in \{0,1\}$. Finally, $\mathcal{B}$ samples $\hat{b} \leftarrow \{0,1\}$ uniformly at random and sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to $\mathcal{A}$. $\mathcal{A}$ outputs its guess b'. If $b' = \hat{b}$, $\mathcal{B}$ sends 0 as its guess (i.e., ct_0 is an encryption of $\mathbf{0}$), otherwise it sends 1 as its guess (i.e., ct_0 is an encryption of string $\mathbf{x}\|\mathbf{y}_0\|\mathbf{z}_0\|\theta$).

Note that if ct_0 is an encryption of string $\mathbf{x}\|\mathbf{y}_0\|\mathbf{z}_0\|\theta$, then $\mathcal{B}$ perfectly simulates the experiment of Hybrid_1 for adversary $\mathcal{A}$. Otherwise it simulates the intermediate step. As a result, if $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^{1.5}_{\mathcal{A}}|$ is non-negligible, then $\mathcal{B}$ breaks the security of public key encryption scheme PKE with non-negligible advantage. Similarly, $|\mathsf{Adv}^{1.5}_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$ and thus $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$ □

Lemma 8.4. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Since $\mathbf{x}_\top \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}$, $\mathbf{H}_\infty(\mathbf{x}_\perp) \geq m > n\log q + \omega(\log n)$, and we have by the Leftover Hash Lemma (**Corollary** 3.8) that the following distributions are statistically indistinguishable:

$$\begin{aligned} &\{(\mathbf{A}_\mathsf{L}, \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_\top) : \mathbf{A}_\mathsf{L} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}, \mathbf{x}_\top \leftarrow\!\$\ \mathcal{D}_{\mathbb{Z}_q^m, \varsigma/m}\} \\ &\approx_s \{(\mathbf{A}_\mathsf{L}, \boldsymbol{\rho}) : \mathbf{A}_\mathsf{L} \leftarrow\!\$\ \mathbb{Z}_q^{n\times m}, \boldsymbol{\rho} \leftarrow\!\$\ \mathbb{Z}_q^n\} \end{aligned}$$

Thus the vector $\mathbf{A} \cdot \begin{bmatrix} \mathbf{x} \\ \theta \end{bmatrix} + \mathsf{H}(\delta) = \mathbf{A}_\mathsf{L} \cdot \mathbf{x}_\top + \mathbf{A}_\mathsf{R} \cdot \mathbf{x}_\perp + \theta\mathbf{a} + \mathsf{H}(\delta_b)$ is statistically indistinguishable from a random vector in $\mathbb{Z}_q^n$. □

Lemma 8.5. For all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^3_{\mathcal{A}} \leq 1/2 + \mathsf{negl}(\lambda)$.

Proof. We will argue the statistical indistinguishability. To see this, first observe that in step 4. (a), the challenger always checks for each $b \in \{0,1\}$ whether $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ and aborts if it is not. So at step 4. (c), it must hold that $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ and consequently every entry in $\mathbf{z}_b \in \mathbb{Z}_q^{2m}$ is in the range $[-\beta, \beta]$. Thus every entry in $\frac{\mathbf{z}_{b,\perp} + \mathbf{z}_{1-b,\perp}}{2} \in \mathbb{Z}_q^m$ is in the range $[-\beta, \beta]$. Recall further that, by construction, the integer vector $\mathbf{x}_\perp$ is uniformly sampled over $\left[-\frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta, \frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta\right]^m$. Therefore,

$$\frac{\beta}{\frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta} = \frac{m\sqrt{m}}{2^{\lambda/2-1}}$$

is at most negligible, and the smudging lemma (Lemma 3.9) applies. Then, for $\mathbf{x}_\perp \leftarrow\!\$ \left[-\frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta, \frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta\right]^m$, $\mathbf{z}_{b,\perp}, \mathbf{z}_{1-b,\perp} \in [-\beta, \beta]^m, \theta \leftarrow\!\$ \{0,1\}$, we have:

$$\begin{aligned} &\{(1-2\theta)(\mathbf{z}_{0,\perp} - \mathbf{x}_\perp), (1-2\theta)(\mathbf{z}_{1,\perp} - \mathbf{x}_\perp), \mathbf{z}_{0,\perp}, \mathbf{z}_{1,\perp}\} \\ &\approx_s \left\{(1-2\theta)(\frac{\mathbf{z}_{0,\perp} - \mathbf{z}_{1,\perp}}{2} - \mathbf{x}_\perp), (1-2\theta)(\frac{\mathbf{z}_{1,\perp} - \mathbf{z}_{0,\perp}}{2} - \mathbf{x}_\perp), \mathbf{z}_{0,\perp}, \mathbf{z}_{1,\perp}\right\}. \end{aligned}$$

Next, since $\mathbf{x}_\perp$ is uniformly sampled from $\left[-\frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta, \frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\ \beta\right]^m$,

$$\left\{(1-2\theta)(\frac{\mathbf{z}_{0,\perp}-\mathbf{z}_{1,\perp}}{2}-\mathbf{x}_\perp),(1-2\theta)(\frac{\mathbf{z}_{1,\perp}-\mathbf{z}_{0,\perp}}{2}-\mathbf{x}_\perp),\mathbf{z}_{0,\perp},\mathbf{z}_{1,\perp}\right\}$$
$$\equiv\left\{(1-2\theta)(\frac{\mathbf{z}_{0,\perp}-\mathbf{z}_{1,\perp}}{2}+\mathbf{x}_\perp),(1-2\theta)(\frac{\mathbf{z}_{1,\perp}-\mathbf{z}_{0,\perp}}{2}+\mathbf{x}_\perp),\mathbf{z}_{0,\perp},\mathbf{z}_{1,\perp}\right\}.$$

Since $\theta \leftarrow\!\!\$\ \{0,1\}$,

$$\left\{(1-2\theta)(\frac{\mathbf{z}_{0,\perp}-\mathbf{z}_{1,\perp}}{2}+\mathbf{x}_\perp),(1-2\theta)(\frac{\mathbf{z}_{1,\perp}-\mathbf{z}_{0,\perp}}{2}+\mathbf{x}_\perp),\mathbf{z}_{0,\perp},\mathbf{z}_{1,\perp}\right\}$$
$$\equiv\left\{(1-2\theta)(\frac{\mathbf{z}_{1,\perp}-\mathbf{z}_{0,\perp}}{2}-\mathbf{x}_\perp),(1-2\theta)(\frac{\mathbf{z}_{0,\perp}-\mathbf{z}_{1,\perp}}{2}-\mathbf{x}_\perp),\mathbf{z}_{0,\perp},\mathbf{z}_{1,\perp}\right\}.$$

Again by smudging,

$$\left\{(1-2\theta)(\frac{\mathbf{z}_{1,\perp}-\mathbf{z}_{0,\perp}}{2}-\mathbf{x}_\perp),(1-2\theta)(\frac{\mathbf{z}_{0,\perp}-\mathbf{z}_{1,\perp}}{2}-\mathbf{x}_\perp),\mathbf{z}_{0,\perp},\mathbf{z}_{1,\perp}\right\}$$
$$\approx_s \{(1-2\theta)(\mathbf{z}_{1,\perp}-\mathbf{x}_\perp),(1-2\theta)(\mathbf{z}_{0,\perp}-\mathbf{x}_\perp),\mathbf{z}_{0,\perp},\mathbf{z}_{1,\perp}\}.$$

The above equations imply that $\mathbf{w}_0$ is statistically indistinguishable from $\mathbf{w}_1$ in Hybrid 3. Meanwhile, by definition of Hybrid 3, π_0 and π_1, ct_0 and ct_1 are also indistinguishable. Thus our lemma follows.

□

From the lemmas above, it follows that NIBS construction 8.1 satisfies nonce blindness assuming that NIZK satisfies zero-knowledge property and PKE is secure.

□

8.3 Receiver blindness

Next, we show that this construction also satisfies receiver blindness.

Theorem 8.6. Assume that NIZK proof system NIZK satisfies zero-knowledge property and public key encryption scheme PKE is IND-CPA secure, then the NIBS construction 8.1 is receiver blind.

Proof. We prove the theorem through the following hybrids:

Hybrid_0 This corresponds to the real experiment.

1. Challenger samples $\mathbf{A} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times(2m+1)}$, $\mathbf{B} \leftarrow\!\!\$\ \mathbb{Z}_q^{n\times 2m}$ uniformly at random, $\mathsf{pke.pk} \leftarrow\!\!\$\ \mathsf{PKE.KeyGen}(1^\lambda)$, and $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$. Challenger then sets pp as $(\mathbf{C},\mathbf{A},\mathbf{B},\mathsf{pke.pk},\mathsf{nizk.crs})$ and sends pp to the adversary.

2. Next, challenger samples $(\mathsf{sk}_{R_b},\mathsf{pk}_{R_b}) \leftarrow \mathsf{KeyGen}_R(\mathsf{pp})$, for $b \in \{0,1\}$. The challenger sends pk_{R_0} and pk_{R_1} to $\mathcal{A}$.

3. The adversary sends a matrix $\mathbf{C} \in \mathbb{Z}_q^{n\times 2m}$ as the verification key, and issues two presignatures psig_0 and psig_1.

4. The challenger then generates message and signature pairs:

(a) For $b \in \{0,1\}$, challenger sets $\mathbf{z}_b$ as psig_b, $\mathbf{y}_b$ as nonce_b, parses sk_{R_b} as $(\mathbf{x}_b, \delta_b)$, and then checks whether $\mathbf{C} \cdot \mathbf{z}_b + \mathbf{B} \cdot \mathbf{y}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_b \\ \theta_b \end{bmatrix} + H(\delta_b)$ and $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ and $\mathbf{y}_b \in \{\pm 1\}^{2m}$. If the any of the checks fails, challenger outputs $\perp$ and aborts.

(b) Otherwise, for $b \in \{0,1\}$, challenger continues to generate $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_b \| \mathbf{y}_b \| \mathbf{z}_b \| \theta_b; r_b)$ with freshly sampled randomness r_0 and r_1.

(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = (1 - 2\theta)(\mathbf{z}_{b,\perp} - \mathbf{x}_{b,\perp})$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$, witness ω_b as $(\mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b, \theta_b, r_b)$, and generates NIZK proof $\pi_b \leftarrow \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_b, \omega_b)$.

(d) Next, challenger sets message μ_b as $(\mathbf{w}_b, \delta_b)$ and signature σ_b as (π_b, ct_b). Finally, challenger samples $\hat{b} \leftarrow\!\!\$\ \{0,1\}$ uniformly at random and sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to the adversary.

5. Adversary sends a bit b' and wins if $\hat{b} = b'$.

Hybrid_1 This is the same as Hybrid_0, except that instead of generating the NIZK proofs π_0 and π_1, the challenger simulates it without any witnesses.

4.(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = (1 - 2\theta)(\mathbf{z}_{b,\perp} - \mathbf{x}_{b,\perp})$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$. Without setting any witnesses, challenger generates π_b using NIZK simulator.

Hybrid_2 This is the same as Hybrid_1, except that for $b \in \{0,1\}$, instead of generating $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_b \| \mathbf{y}_b \| \mathbf{z}_b \| \theta_b; r_b)$, challenger sets ct_b as $\mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_b)$, ie.,

4.(b) Challenger generates $\mathsf{ct}_0 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_0)$ and $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_1)$ with freshly sampled randomness r_0 and r_1.

Hybrid_3 This the same as Hybrid_2, except that the challenger samples pk_{R_0} and pk_{R_1} uniformly at random.

2. When generating pk_{R_b} for $b \in \{0,1\}$, instead of setting it as $\mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_b \\ \theta_b \end{bmatrix} + \mathsf{H}(\delta_b)$, the challenger sets it as a uniformly sampled vector, such that $\mathsf{pk}_{R_b} \leftarrow\!\!\$\ \mathbb{Z}_q^n$.

Now, let $\mathsf{Adv}^j_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in the NIBS blindness game in Hybrid_j. Then, the following must hold:

Lemma 8.7. Assuming zero-knowledge property of NIZK, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^0_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by extending the proof of Lemma 7.9 to Construction 8.1. □

Lemma 8.8. If public key encryption scheme PKE is a secure public key encryption scheme, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by extending the proof of Lemma 7.10 to Construction 8.1. □

Lemma 8.9. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by the proof of Lemma 8.4 repeated each $b \in \{0,1\}$. □

Finally, will once more leverage the smudging lemma (Lemma 3.9) to argue that for each $b \in \{0,1\}$ (the distributions over) $\mathbf{w}_b = (1-2\theta_b)(\mathbf{z}_{b,\perp} - \mathbf{x}_{b,\perp})$ is statistically indistinguishable from uniform. As, in step 4. (a), the challenger always checks for each $b \in \{0,1\}$ whether $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ and aborts if it is not, it must hold that $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ after that step, and consequently every entry in $\mathbf{z}_b \in \mathbb{Z}_q^{2m}$ is in the range $[-\beta, \beta]$. Thus every entry in $\mathbf{z}_{b,\perp} \in \mathbb{Z}_q^m$ is in the range $[-\beta, \beta]$. Further each $\mathbf{x}_{b,\perp}$ is uniformly sampled over $\left[-\frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\beta, \frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\beta\right]^m$ by construction. Therefore we have that the ratio

$$\frac{\beta}{\frac{2^{(\lambda/2-1)}}{m\sqrt{m}}\beta} = \frac{m\sqrt{m}}{2^{(\lambda/2-1)}}$$

is negligible in λ, so that the lemma applies and $\mathbf{z}_{b,\perp} - \mathbf{x}_{b,\perp}$ is statistically indistinguishable from $\mathbf{x}_{b,\perp}$. Since $\mathbf{x}_{b,\perp}$ and θ_b are uniformly chosen, the claim holds.

Therefore, any adversary has negligible advantage under Hybrid_3. From the above lemmas, it follows that NIBS construction 8.1 satisfies receiver blindness assuming that NIZK satisfies zero-knowledge property and PKE is secure.

□

Lastly we give the following theorem concerning the one-more unforgeability of this construction.

8.4 One-more unforgeability

Theorem 8.10. Assume that NIZK proof system NIZK satisfies soundness, lattice trapdoor bLT satisfies well-distributedness, and the rOM-ISIS assumption holds, then our construction 8.1 is one-more unforgeable.

Proof. Consider the following hybrids:

Hybrid_0 This is the actual one-more unforgeability game for NIBS:

1. Challenger samples $\mathbf{A} \leftarrow\!\$\ \mathbb{Z}_q^{n\times(2m+1)}$, $\mathbf{B} \leftarrow\!\$\ \mathbb{Z}_q^{n\times 2m}$ uniformly at random, $\mathsf{pke.pk} \leftarrow\!\$\ \mathsf{PKE.KeyGen}(1^\lambda)$, $\mathsf{nizk.crs} \leftarrow\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$ and sets pp as $(\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs})$. Next, it samples $\mathbf{T_C}, \mathbf{C} \leftarrow\!\$\ \mathsf{bLT.TrapGen}(1^\lambda, n, 2m, q)$ and sets $\mathsf{sk} := \mathbf{T_C}$, $\mathsf{vk} := \mathbf{C}$. Challenger sends pp and vk to the adversary.

2. Adversary chooses pk_R and requests a presignature, to which the challenger replies with $(\mathsf{psig}, \mathsf{nonce}) := \mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R)$. The adversary repeats this step a total of ℓ times.

3. Adversary outputs k message and signature pairs $((\mu_1, \sigma_1), \ldots, (\mu_k, \sigma_k))$. Adversary wins if $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, $\mathsf{Verify}(\mathsf{vk}, \mu_i, \sigma_i) = 1$ for all $i \in [k]$, and $k = \ell + 1$.

Hybrid_1 This is the same as Hybrid_0 aside from one key difference that instead of uniformly sampling the matrix $\mathbf{A}$, it is chosen by first sampling a matrix $\mathbf{R} \leftarrow\!\$\ \{0,1\}^{2m\times(2m+1)}$ and then setting $\mathbf{A} = \mathbf{C} \cdot \mathbf{R}$.

1. Challenger samples $\mathbf{R} \leftarrow\$ \{0,1\}^{2m\times(2m+1)}$ uniformly at random and sets $\mathbf{A} = \mathbf{C}\cdot\mathbf{R}$. Challenger then samples $\mathbf{B} \leftarrow\$ \mathbb{Z}_q^{n\times 2m}$, $\mathsf{pke.pk} \leftarrow\$ \mathsf{PKE.Setup}(1^\lambda)$, $\mathsf{nizk.crs} \leftarrow\$ \mathsf{NIZK.Setup}(1^\lambda)$, and sets pp as $(\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs})$. Next, it samples $\mathbf{T_C}, \mathbf{C} \leftarrow\$ \mathsf{bLT.TrapGen}(1^\lambda, n, 2m, q)$ and sets $\mathsf{sk} := \mathbf{T_C}$, $\mathsf{vk} := \mathbf{C}$. Challenger sends pp and vk to the adversary.

Let $\mathsf{Adv}^j_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in the NIBS one-more unforgeability game in Hybrid_j. Then, the following must hold:

Lemma 8.11. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^0_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| \le \mathsf{negl}(\lambda)$.

Proof. Omitted as it is identical to the proof of Lemma 7.2 but with matrices $\mathbf{R} \in \{0,1\}^{2m\times(2m+1)}$ and $\mathbf{A} \in \mathbb{Z}_q^{n\times(2m+1)}$, and the divergence over the distribution of $\mathbf{R}$ scaled accordingly. □

Lemma 8.12. Assuming that NIZK argument is sound and the rOM-ISIS assumption holds, the one-more-unforgeability adversary $\mathcal{A}$ can have at most negligible advantage in Hybrid_1.

Proof. Suppose there exists a PPT attacker $\mathcal{A}$ that wins the one-more unforgeability game with non-negligible probability $\epsilon = \epsilon(\lambda)$. It outputs k message and signature pairs $((\mu_1,\sigma_1),\ldots,(\mu_k,\sigma_k))$, and wins if and only if $\mu_i \neq \mu_j$ for $1 \le i < j \le k$, $\mathsf{Verify}(\mathsf{vk}, \mu_i, \sigma_i) = 1$ for $1 \le i \le k$, and $k > \ell$. As before, we parse inputs and signatures (μ_i, σ_i) from $\mathcal{A}$'s output as $((\mathbf{w}_i, \delta_i), (\pi_i, \mathsf{ct}_i))$ for all $i \in [k]$. Let us divide the potential attacker into the following cases,

- Type 1: This is same as the type 1 attacker from Lemma 7.3 extended to NIZK, the NIZK for language $\mathcal{L}_3$.
- Type 2: This is same as the type 2 attacker from Lemma 7.3 except for this attacker, there is at least one tuple of $(\mathbf{x}_j, \mathbf{y}_j, \mathbf{z}_j, \theta_j)$ for some $j \in [k]$, such that $\mathbf{C}\cdot\mathbf{z}_j + \mathbf{B}\cdot\mathbf{y}_j = \mathbf{A}\cdot\begin{bmatrix}\mathbf{x}_j\\ \theta_j\end{bmatrix} + \mathsf{H}(\delta_j)$ where the attacker never queries the random oracle H on δ_j.
- Type 3: It is exactly the same as the type 3 attacker from Lemma 7.3.

Claim 8.13. Assuming that NIZK satisfies soundness property, then type 1 attacker $\mathcal{A}$ has at most negligible advantage.

Proof. (omitted) follows by extending the proof of Claim 7.4 to Construction 8.1. □

Claim 8.14. Type 2 attacker $\mathcal{A}$ has at most negligible advantage.

Proof. By the same token as the proof of Claim 7.5, $\mathsf{H}(\delta^*)$ is not defined by the random oracle and is completely random in the view of $\mathcal{A}$ so that it has at most negligible advantage in finding δ^* such that $\mathsf{H}(\delta^*) = \mathbf{C}\cdot\mathbf{z}_j - \mathbf{A}\cdot\begin{bmatrix}\mathbf{x}_j\\ \theta_j\end{bmatrix}$. □

Claim 8.15. Assuming that the rOM-ISIS assumption holds, type 3 adversary $\mathcal{A}$ has at most negligible advantage.

Proof. Assume that type 3 attacker $\mathcal{A}$ has non-negligible advantage $\epsilon = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks the randomized one-more ISIS assumption.

The rOM-ISIS challenger starts by sampling and outputting a matrix $\mathbf{C}, \mathbf{B} \in \mathbb{Z}_q^{n \times 2m}$ uniformly at random. Reduction algorithm $\mathcal{B}$ then samples $\mathbf{R} \leftarrow\!\!\$\, \{0,1\}^{2m \times (2m+1)}$ and sets $\mathbf{A}$ as $\mathbf{A} = \mathbf{C} \cdot \mathbf{R}$. Next, $\mathcal{B}$ generates $(\mathsf{pke.pk}, \mathsf{pke.sk})$ and $\mathsf{nizk.crs}$. $\mathcal{B}$ outputs $\mathsf{pp} := (\mathbf{A}, \mathsf{pke.pk}, \mathsf{nizk.crs})$ and verification key $\mathsf{vk} := \mathbf{C}$.

Next, $\mathcal{A}$ is allowed to make a series of hash queries and presignature queries (in any order). On each fresh hash query that $\mathcal{A}$ provides an input δ, $\mathcal{B}$ simply makes a syndrome query to the challenger. $\mathcal{B}$ forwards challenger's output to $\mathcal{A}$ as the output of $\mathsf{H}(\delta)$. Meanwhile, $\mathcal{B}$ keeps a recording of all hash queries from $\mathcal{A}$. For each presignature query on input $\mathbf{t}'$ from $\mathcal{A}$, $\mathcal{B}$ makes a preimage query to the challenger using the same input $\mathbf{t}'$. Then, $\mathcal{B}$ forwards challenger's output $\mathbf{z}'$ and $\mathbf{y}'$ to $\mathcal{A}$ such that $\mathbf{C} \cdot \mathbf{z}' + \mathbf{B} \cdot \mathbf{y}' = \mathbf{t}'$. $\mathcal{A}$ makes a total of ℓ such presignature queries. To win the one-more unforgeability game, $\mathcal{A}$ outputs k message and signature pairs $((\mu_1, \sigma_1), \ldots, (\mu_k, \sigma_k))$ such that $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, $\mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \mu_i, \sigma_i) = 1$ for all $i \in [k]$, and $k = \ell + 1$.

For all $i \in [k]$, $\mathcal{B}$ parses μ_i as $(\mathbf{w}_i, \delta_i)$ and σ_i as (π_i, ct_i). $\mathcal{B}$ obtains $\mathbf{x}_i \in \mathbb{Z}_q^{2m}$, $\mathbf{y}_i \in \{\pm 1\}^{2m}$, $\mathbf{z}_i \in \mathbb{Z}_q^{2m}$, and θ_i by decrypting ct_i, where $\mathbf{x}_i \| \mathbf{y}_i \| \mathbf{z}_i \| \theta_i = \mathsf{PKE.Dec}(\mathsf{pke.sk}, \mathsf{ct}_i)$. Thus for all $i \in [k]$,

$$\mathsf{H}(\delta_i) = \mathbf{C} \cdot \mathbf{z}_i + \mathbf{B} \cdot \mathbf{y}_i - \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_i \\ \theta_i \end{bmatrix} = \mathbf{C} \cdot \left(\mathbf{z}_i - \mathbf{R} \cdot \begin{bmatrix} \mathbf{x}_i \\ \theta_i \end{bmatrix} \right) + \mathbf{B} \cdot \mathbf{y}_i.$$

Moreover, since $\mathbf{R}$ is a binary matrix, $\|\mathbf{z}_i\| \leq \sqrt{2}\beta$, and $\|\mathbf{x}_i\| \leq \left(1 + 2^{(\lambda/2-1)}\right) \cdot \beta/m$

$$\left\| \mathbf{z}_i - \mathbf{R} \cdot \begin{bmatrix} \mathbf{x}_i \\ \theta_i \end{bmatrix} \right\| \leq (2^{\lambda/2} + 2\sqrt{2} + 2) \cdot \beta = \beta'$$

Note that $\mu_i \neq \mu_j$, for $1 \leq i < j \leq k$. There are two cases:

Case 1: For some $1 \leq i < j \leq k$, $\delta_i = \delta_j$. By definition of type 4 attacker, we have $\mathbf{y}_i \neq \mathbf{y}_j$.

Case 2: $\delta_i \neq \delta_j$. Then with all but negligible probability, we have $\mathsf{H}(\delta_i) \neq \mathsf{H}(\delta_j)$, which implies that $\mathbf{C} \cdot \left(\mathbf{z}_i - \mathbf{R} \cdot \begin{bmatrix} \mathbf{x}_i \\ \theta_i \end{bmatrix} \right) + \mathbf{B} \cdot \mathbf{y}_i \neq \mathbf{C} \cdot \left(\mathbf{z}_j - \mathbf{R} \cdot \begin{bmatrix} \mathbf{x}_j \\ \theta_j \end{bmatrix} \right) + \mathbf{B} \cdot \mathbf{y}_i$. Thus, either $\mathbf{z}_i - \mathbf{R} \cdot \begin{bmatrix} \mathbf{x}_i \\ \theta_i \end{bmatrix} \neq \mathbf{z}_j - \mathbf{R} \cdot \begin{bmatrix} \mathbf{x}_j \\ \theta_j \end{bmatrix}$, or $\mathbf{y}_i \neq \mathbf{y}_j$.

As a result, $\mathcal{B}$ wins the rOM-ISIS game by sending back $\mathbf{z}_i - \mathbf{R} \cdot \begin{bmatrix} \mathbf{x}_i \\ \theta_i \end{bmatrix}$ and $\mathbf{y}_i$ for $i \in [k]$. □

Note that any successful one-more unforgeability attacker $\mathcal{A}$ must be one of type 1, 2, and 3. Thus, combining the above arguments, the lemma follows.

□

This completes the proof for one-more unforgeability of Construction 8.1.

□

Acknowledgements. We thank Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav for answering questions about the OM-ISIS assumption. We also thank Lucjan Hanzlik for helpful discussions on the motivation behind NIBS, potential applications enabled by varying levels of blindness security as well as pointing out a subtle issue in an earlier version of our NIBS definition.

References

[ABB10a] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 553–572, French Riviera, May 30 – June 3, 2010. Springer, Heidelberg, Germany.

[ABB10b] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 98–115, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Heidelberg, Germany.

[AF96] Masayuki Abe and Eiichiro Fujisaki. How to date blind signatures. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 244–251, Kyongju, Korea, November 3–7, 1996. Springer, Heidelberg, Germany.

[AJL+12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 483–501, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany.

[Ajt96] M. Ajtai. Generating hard instances of lattice problems (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96, page 99–108, New York, NY, USA, 1996. Association for Computing Machinery.

[AKSY22] Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. Practical, round-optimal lattice-based blind signatures. In Heng Yin, Angelos Stavrou, Cas Cremers, and Elaine Shi, editors, *ACM CCS 2022*, pages 39–53, Los Angeles, CA, USA, November 7–11, 2022. ACM Press.

[AMPR14] Arash Afshar, Payman Mohassel, Benny Pinkas, and Ben Riva. Non-interactive secure computation based on cut-and-choose. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 387–404, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany.

[BCC04] Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick McDaniel, editors, *ACM CCS 2004*, pages 132–145, Washington, DC, USA, October 25–29, 2004. ACM Press.

[BdMW16] Florian Bourse, Rafaël del Pino, Michele Minelli, and Hoeteck Wee. FHE circuit privacy almost for free. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 62–89, Santa Barbara, CA, USA, August 14–18, 2016. Springer, Heidelberg, Germany.

[BF11] Dan Boneh and David Mandell Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011*, volume 6571 of *LNCS*, pages 1–16, Taormina, Italy, March 6–9, 2011. Springer, Heidelberg, Germany.

[BJOV18] Saikrishna Badrinarayanan, Abhishek Jain, Rafail Ostrovsky, and Ivan Visconti. Non-interactive secure computation from one-way functions. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part III*, volume 11274 of *LNCS*, pages 118–138, Brisbane, Queensland, Australia, December 2–6, 2018. Springer, Heidelberg, Germany.

[BL13] Foteini Baldimtsi and Anna Lysyanskaya. Anonymous credentials light. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 2013*, pages 1087–1098, Berlin, Germany, November 4–8, 2013. ACM Press.

[BLNS23a] Ward Beullens, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Lattice-based blind signatures: Short, efficient, and round-optimal. Cryptology ePrint Archive, Report 2023/077, 2023. https://eprint.iacr.org/2023/077.

[BLNS23b] Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Alessandro Sorniotti. A framework for practical anonymous credentials from lattices. In *CRYPTO 2023, Part II*, LNCS, pages 384–417, Santa Barbara, CA, USA, August 2023. Springer, Heidelberg, Germany.

[BNPS02] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The power of RSA inversion oracles and the security of Chaum's RSA-based blind signature scheme. In Paul F. Syverson, editor, *FC 2001*, volume 2339 of *LNCS*, pages 319–338, Grand Cayman, British West Indies, February 19–22, 2002. Springer, Heidelberg, Germany.

[BNPS03] Mihir Bellare, Chanathip Namprempre, David Pointcheval, and Michael Semanko. The one-more-RSA-inversion problems and the security of Chaum's blind signature scheme. *Journal of Cryptology*, 16(3):185–215, June 2003.

[Bol03] Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In Yvo Desmedt, editor, *PKC 2003*, volume 2567 of *LNCS*, pages 31–46, Miami, FL, USA, January 6–8, 2003. Springer, Heidelberg, Germany.

[CDI+19] Melissa Chase, Yevgeniy Dodis, Yuval Ishai, Daniel Kraschewski, Tianren Liu, Rafail Ostrovsky, and Vinod Vaikuntanathan. Reusable non-interactive secure computation. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 462–488, Santa Barbara, CA, USA, August 18–22, 2019. Springer, Heidelberg, Germany.

[CGT06] Sébastien Canard, Matthieu Gaud, and Jacques Traoré. Defeating malicious servers in a blind signatures based voting system. In Giovanni Di Crescenzo and Avi Rubin, editors, *FC 2006*, volume 4107 of *LNCS*, pages 148–153, Anguilla, British West Indies, February 27 – March 2, 2006. Springer, Heidelberg, Germany.

[Cha83] David Chaum. Blind signature system. In David Chaum, editor, *CRYPTO'83*, page 153, Santa Barbara, CA, USA, 1983. Plenum Press, New York, USA.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, pages 523–552, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[DGS+18] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *PoPETs*, 2018(3):164–180, July 2018.

[dK22] Rafaël del Pino and Shuichi Katsumata. A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 306–336, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Heidelberg, Germany.

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *Transactions on Cryptographic Hardware and Embedded Systems*, 2018, Issue 1:238–268, 2018.

[DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540, Interlaken, Switzerland, May 2–6, 2004. Springer, Heidelberg, Germany.

[FHK+17] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-fourier lattice-based compact signatures over ntru. Technical report, 2017.

[FHKS16] Georg Fuchsbauer, Christian Hanser, Chethan Kamath, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model from weaker assumptions. In Vassilis Zikas and Roberto De Prisco, editors, *SCN 16*, volume 9841 of *LNCS*, pages 391–408, Amalfi, Italy, August 31 – September 2, 2016. Springer, Heidelberg, Germany.

[FHS15] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 233–253, Santa Barbara, CA, USA, August 16–20, 2015. Springer, Heidelberg, Germany.

[FHS19] Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Structure-preserving signatures on equivalence classes and constant-size anonymous credentials. *Journal of Cryptology*, 32(2):498–546, April 2019.

[Fis06] Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In Cynthia Dwork, editor, *CRYPTO 2006*, volume 4117 of *LNCS*, pages 60–77, Santa Barbara, CA, USA, August 20–24, 2006. Springer, Heidelberg, Germany.

[Gha17] Essam Ghadafi. Efficient round-optimal blind signatures in the standard model. In Aggelos Kiayias, editor, *FC 2017*, volume 10322 of *LNCS*, pages 455–473, Sliema, Malta, April 3–7, 2017. Springer, Heidelberg, Germany.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206, Victoria, BC, Canada, May 17–20, 2008. ACM Press.

[GRS+11] Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round optimal blind signatures. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 630–648, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Heidelberg, Germany.

[HAB+17] Ethan Heilman, Leen Alshenibr, Foteini Baldimtsi, Alessandra Scafuro, and Sharon Goldberg. TumbleBit: An untrusted bitcoin-compatible anonymous payment hub. In *NDSS 2017*, San Diego, CA, USA, February 26 – March 1, 2017. The Internet Society.

[Han23] Lucjan Hanzlik. Non-interactive blind signatures for random messages. In Carmit Hazay and Martijn Stam, editors, *EUROCRYPT 2023, Part V*, volume 14008 of *LNCS*, pages 722–752, Lyon, France, April 23–27, 2023. Springer, Heidelberg, Germany.

[HBG16] Ethan Heilman, Foteini Baldimtsi, and Sharon Goldberg. Blindly signed contracts: Anonymous on-blockchain and off-blockchain bitcoin transactions. In Jeremy Clark, Sarah Meiklejohn, Peter Y. A. Ryan, Dan S. Wallach, Michael Brenner, and Kurt Rohloff, editors, *Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers*, volume 9604 of *Lecture Notes in Computer Science*, pages 43–60. Springer, 2016.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[HKKL07] Carmit Hazay, Jonathan Katz, Chiu-Yuen Koo, and Yehuda Lindell. Concurrently-secure blind signatures without random oracles or setup assumptions. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 323–341, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Heidelberg, Germany.

[HS14] Christian Hanser and Daniel Slamanig. Structure-preserving signatures on equivalence classes and their application to anonymous credentials. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 491–511, Kaoshiung, Taiwan, R.O.C., December 7–11, 2014. Springer, Heidelberg, Germany.

[HS21] Lucjan Hanzlik and Daniel Slamanig. With a little help from my friends: Constructing practical anonymous credentials. In Giovanni Vigna and Elaine Shi, editors, *ACM CCS 2021*, pages 2004–2023, Virtual Event, Republic of Korea, November 15–19, 2021. ACM Press.

[IKO+11] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, Manoj Prabhakaran, and Amit Sahai. Efficient non-interactive secure computation. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 406–425, Tallinn, Estonia, May 15–19, 2011. Springer, Heidelberg, Germany.

[KNYY21] Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Round-optimal blind signatures in the plain model from classical and quantum standard assumptions. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021*, *Part I*, volume 12696 of *LNCS*, pages 404–434, Zagreb, Croatia, October 17–21, 2021. Springer, Heidelberg, Germany.

[Lin08] Yehuda Lindell. Lower bounds and impossibility results for concurrent self composition. *Journal of Cryptology*, 21(2):200–249, April 2008.

[LN22] Vadim Lyubashevsky and Ngoc Khanh Nguyen. BLOOM: Bimodal lattice one-out-of-many proofs and applications. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022*, *Part IV*, volume 13794 of *LNCS*, pages 95–125, Taipei, Taiwan, December 5–9, 2022. Springer, Heidelberg, Germany.

[LNP22a] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Efficient lattice-based blind signatures via gaussian one-time signatures. In Goichiro Hanaoka, Junji Shikata, and Yohei Watanabe, editors, *PKC 2022*, *Part II*, volume 13178 of *LNCS*, pages 498–527, Virtual Event, March 8–11, 2022. Springer, Heidelberg, Germany.

[LNP22b] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022*, *Part II*, volume 13508 of *LNCS*, pages 71–101, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Heidelberg, Germany.

[LPS10] Vadim Lyubashevsky, Adriana Palacio, and Gil Segev. Public-key cryptographic primitives provably as secure as subset sum. In Daniele Micciancio, editor, *Theory of Cryptography*, pages 382–400, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany.

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-subgroup multisignatures: Extended abstract. In Michael K. Reiter and Pierangela Samarati, editors, *ACM CCS 2001*, pages 245–254, Philadelphia, PA, USA, November 5–8, 2001. ACM Press.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*,

volume 7237 of *LNCS*, pages 700–718, Cambridge, UK, April 15–19, 2012. Springer, Heidelberg, Germany.

[MR04] D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 372–381, 2004.

[MR17] Payman Mohassel and Mike Rosulek. Non-interactive secure 2PC in the offline/online and batch settings. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part III*, volume 10212 of *LNCS*, pages 425–455, Paris, France, April 30 – May 4, 2017. Springer, Heidelberg, Germany.

[MW16] Pratyay Mukherjee and Daniel Wichs. Two round multiparty computation via multi-key FHE. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 735–763, Vienna, Austria, May 8–12, 2016. Springer, Heidelberg, Germany.

[OPP14] Rafail Ostrovsky, Anat Paskin-Cherniavsky, and Beni Paskin-Cherniavsky. Maliciously circuit-private FHE. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 536–553, Santa Barbara, CA, USA, August 17–21, 2014. Springer, Heidelberg, Germany.

[PS96] David Pointcheval and Jacques Stern. Provably secure blind signature schemes. In Kwangjo Kim and Tsutomu Matsumoto, editors, *ASIACRYPT'96*, volume 1163 of *LNCS*, pages 252–265, Kyongju, Korea, November 3–7, 1996. Springer, Heidelberg, Germany.

[PZ13] Christian Paquin and Greg Zaverucha. U-prove cryptographic specification v1.1 (revision 3). Technical report, Microsoft Corporation, December 2013.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93, Baltimore, MA, USA, May 22–24, 2005. ACM Press.

[SC12] Jae Hong Seo and Jung Hee Cheon. Beyond the limitation of prime-order bilinear groups, and round optimal blind signatures. In Ronald Cramer, editor, *TCC 2012*, volume 7194 of *LNCS*, pages 133–150, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Heidelberg, Germany.

A Equivalence of Two NIBS Formalizations

Let us first recall the syntax of non-interactive blind signatures given by (NIBS) [Han23]. Formally, a non-interactive blind signature scheme consists of the following polynomial-time algorithms:

$\mathsf{KeyGen}_S(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$. On input parameter λ, it samples a public-secret key pair $(\mathsf{sk}, \mathsf{vk})$.

$\mathsf{KeyGen}_R(1^\lambda) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. On input parameter λ, it samples a public-secret key pair $(\mathsf{sk}_R, \mathsf{pk}_R)$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce}) \to \mathsf{psig}$. It takes as input the signer's secret key sk, receiver's public key pk_R and a nonce nonce. It then outputs a presignature psig.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}, \mathsf{nonce})) \to (\mu, \sigma)$. Given the receiver's secret key sk_R as an input, along with a verification key vk and presignature-nonce pair (psig, nonce), it outputs a message-signature pair (μ, σ) or aborts (in which case it outputs $\perp$).

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0, 1\}$. This is the signature scheme verification algorithm that takes as input a verification key and message-signature pair, and outputs 0/1.

Correctness. A non-interactive blind signature scheme satisfies correctness if for every security parameter $\lambda \in \mathbb{N}$, $(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\, \mathsf{KeyGen}_S(1^\lambda)$, $(\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow\!\!\$\, \mathsf{KeyGen}_R(1^\lambda)$, the following holds:

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

where the probability is taken over the random coins of Issue and Obtain.

Definition A.1 (One-more unforgeability)**.** A NIBS scheme $\mathcal{S}$ satisfies one-more unforgeability, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{l} \bigwedge_{i\in[\ell+1]} \mathsf{Verify}(\mathsf{vk}, \mu_i, \sigma_i) = 1 \\ \wedge \left(\bigwedge_{i\neq j\in[\ell+1]} \mu_i \neq \mu_j\right) \end{array} : \begin{array}{r} \mathsf{pp} \leftarrow\!\!\$\, \mathsf{Setup}(1^\lambda) \\ (\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\, \mathsf{KeyGen}_S(\mathsf{pp}) \\ \{(\mu_i, \sigma_i)\}_{i=1}^{\ell+1} \leftarrow\!\!\$\, \mathcal{A}^{O_{\mathsf{sk}}(\cdot,\cdot)}(\mathsf{vk}) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $O_{\mathsf{sk}}(\cdot,\cdot)$ takes as input a receiver's public key pk_{R_i} and a nonce nonce_i, and outputs a presignature psig_i by running $\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_{R_i}, \mathsf{nonce}_i)$, and $\mathcal{A}$ is an admissible adversary iff $\mathcal{A}$ makes at most ℓ queries to O_{sk}.

Definition A.2 (Receiver blindness)**.** A NIBS scheme $\mathcal{S}$ satisfies receiver blindness, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{l} \mathcal{A}(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}}) = \hat{b} : \\ \qquad\qquad \mathsf{pp} \leftarrow\!\!\$\, \mathsf{Setup}(1^\lambda),\ \hat{b} \leftarrow\!\!\$\, \{0,1\}, \\ \qquad\qquad \forall b \in \{0,1\} :\ (\mathsf{sk}_{R_b}, \mathsf{pk}_{R_b}) \leftarrow\!\!\$\, \mathsf{KeyGen}_R(\mathsf{pp}) \\ \qquad\qquad (\mathsf{vk}, (\mathsf{psig}_b, \mathsf{nonce}_b)) \leftarrow\!\!\$\, \mathcal{A}(\mathsf{pk}_{R_0}, \mathsf{pk}_{R_1}) \\ \qquad\qquad \forall b \in \{0,1\} :\ (\mu_b, \sigma_b) \leftarrow\!\!\$\, \mathsf{Obtain}(\mathsf{sk}_{R_b}, \mathsf{vk}, (\mathsf{psig}_b, \mathsf{nonce}_b)) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where $\mathcal{A}$ is an admissible adversary iff $\sigma_0, \sigma_1 \neq \perp$ (i.e., Obtain algorithm does not abort).

Definition A.3 (Nonce blindness)**.** A NIBS scheme $\mathcal{S}$ satisfies nonce blindness, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{l}\mathcal{A}(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}}) = \hat{b} : \\ \qquad \mathsf{pp} \leftarrow\!\!\$\ \mathsf{Setup}(1^\lambda),\ (\mathsf{sk}_R, \mathsf{pk}_R) \leftarrow\!\!\$\ \mathsf{KeyGen}_R(\mathsf{pp})\ \hat{b} \leftarrow\!\!\$\ \{0,1\} \\ \qquad\qquad \forall b \in \{0,1\} :\ (\mathsf{vk}, (\mathsf{psig}_b, \mathsf{nonce}_b)) \leftarrow\!\!\$\ \mathcal{A}(\mathsf{pk}_R), \\ \qquad\qquad \forall b \in \{0,1\} :\ (\mu_b, \sigma_b) \leftarrow\!\!\$\ \mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}_b, \mathsf{nonce}_b))\end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where $\mathcal{A}$ is an admissible adversary iff $\sigma_0, \sigma_1 \neq \perp$ (i.e., Obtain algorithm does not abort).

As we previously noted, our definition and that given by [Han23], where the nonce is an explicit parameter to the Issue algorithm, are equivalent. Let us now expand on the the formal transformation. To that end, first, let $\mathsf{NIBS}' = (\mathsf{KeyGen}'_S, \mathsf{KeyGen}'_R, \mathsf{Issue}', \mathsf{Obtain}', \mathsf{Verify}')$ be a NIBS scheme according to our definition in Section 4. Then, given a pseudorandom function F, we show how to build a NIBS scheme $\mathsf{NIBS} = (\mathsf{KeyGen}_S, \mathsf{KeyGen}_R, \mathsf{Issue}, \mathsf{Obtain}, \mathsf{Verify})$ according to the definition of [Han23].

$\mathsf{KeyGen}_S(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$. The signer's setup algorithm creates $(\mathsf{sk}', \mathsf{vk}') \leftarrow\!\!\$\ \mathsf{KeyGen}'_S(1^\lambda)$ and also samples a PRF key $K \leftarrow\!\!\$\ \{0,1\}^\lambda$. It sets $\mathsf{sk} := (\mathsf{sk}', K)$ and $\mathsf{vk} := \mathsf{vk}'$.

$\mathsf{KeyGen}_R(1^\lambda) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. The receiver's setup algorithm creates $(\mathsf{sk}'_R, \mathsf{pk}'_R) \leftarrow\!\!\$\ \mathsf{KeyGen}'_R(1^\lambda)$ and sets $\mathsf{sk}_R := \mathsf{sk}'_R$ and $\mathsf{pk}_R := \mathsf{pk}'_R$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce}) \to \mathsf{psig}$. The issue algorithm parses sk as (sk', K). It then computes $r := F_K(\mathsf{nonce}, \mathsf{pk}_R)$ and then generates $\mathsf{psig}', \mathsf{nonce}' \leftarrow \mathsf{Issue}'(\mathsf{sk}', \mathsf{pk}_R; r)$. It then outputs $\mathsf{psig} := (\mathsf{psig}', \mathsf{nonce}')$.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}, \mathsf{nonce})) \to (\mu, \sigma)$. The obtain algorithm parses psig as $(\mathsf{psig}', \mathsf{nonce}')$. It then generates $(\mu, \sigma) \leftarrow\!\!\$\ \mathsf{Obtain}'(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}', \mathsf{nonce}'))$ and outputs it.

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0,1\}$. The verification algorithm runs $\mathsf{Verify}'(\mathsf{vk}, \mu, \sigma)$ and returns its output.

Both receiver and nonce blindness follow from that of the underlying NIBS scheme. We now sketch out the proof for one-more unforgeability. More precisely, given an attacker against the one-more unforgeability (per Definition A.1) of NIBS we must build a reduction that breaks the one-more unforgeability (per Definition 4.3) of NIBS′. The essential aspect of the reduction is that it must simulate the oracle O_{sk} (per Definition A.1) to the attacker, who can choose both the pk_R and nonce parts of the input. In order to do so, the reduction must internally maintain a state of all queries $(\mathsf{pk}_{R_i}, \mathsf{nonce}_i)$ made by the attacker along with the response psig_i. Then, on each oracle query it first checks if it was previously queried and if so it returns the corresponding psig_i. Otherwise, it in turn queries the one-more unforgeability challenger (per Definition 4.3) for pk_{R_i} and returns the response $(\mathsf{psig}'_i, \mathsf{nonce}'_i)$ as the psig_i to the attacker (also storing it with the query). Since for every unique query $(\mathsf{pk}_{R_i}, \mathsf{nonce}_i)$, the output of $F_K(\mathsf{nonce}_i, \mathsf{pk}_{R_i})$ is indistinguishable from uniform, the reduction is able to succesfully simulate O_{sk} to the adversary. Finally, the reduction simply outputs the attacker's forgery as its own.

Conversely let $\mathsf{NIBS}' = (\mathsf{KeyGen}'_S, \mathsf{KeyGen}'_R, \mathsf{Issue}', \mathsf{Obtain}', \mathsf{Verify}')$ be a NIBS scheme according to the definition of [Han23]. We show how to build a NIBS scheme $\mathsf{NIBS} = (\mathsf{KeyGen}_S, \mathsf{KeyGen}_R, \mathsf{Issue}, \mathsf{Obtain}, \mathsf{Verify})$ according to our definition in Section 4.

$\mathsf{KeyGen}_S(1^\lambda) \to (\mathsf{sk}, \mathsf{vk})$. The signer's setup algorithm creates $(\mathsf{sk}', \mathsf{vk}') \leftarrow\!\!\$\ \mathsf{KeyGen}'_S(1^\lambda)$ and sets $\mathsf{sk} := \mathsf{sk}'$ and $\mathsf{vk} := \mathsf{vk}'$.

$\mathsf{KeyGen}_R(1^\lambda) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. The receiver's setup algorithm creates $(\mathsf{sk}'_R, \mathsf{pk}'_R) \leftarrow\!\!\$\ \mathsf{KeyGen}'_R(1^\lambda)$ and sets $\mathsf{sk}_R := \mathsf{sk}'_R$ and $\mathsf{pk}_R := \mathsf{pk}'_R$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \to (\mathsf{psig}, \mathsf{nonce})$. The issue algorithm samples nonce uniformly from the appropriate domain and then generates $\mathsf{psig}' \leftarrow\!\!\$\ \mathsf{Issue}'(\mathsf{sk}, \mathsf{pk}_R, \mathsf{nonce})$. It then sets $\mathsf{psig} := \mathsf{psig}'$ and $\mathsf{nonce} := \mathsf{nonce}'$, and outputs them.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}, \mathsf{nonce})) \to (\mu, \sigma)$. The obtain algorithm generates $(\mu, \sigma) \leftarrow\!\!\$\ \mathsf{Obtain}'(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}, \mathsf{nonce}))$ and outputs it.

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0, 1\}$. The verification algorithm runs $\mathsf{Verify}'(\mathsf{vk}, \mu, \sigma)$ and returns its output.

Given an attacker against the one-more unforgeability (per Definition 4.3) of NIBS we sketch out a reduction that breaks the one-more unforgeability (per Definition A.1) of NIBS′. This turns out to be even more straightforward. In order to simulate the oracle O_{sk} (per Definition 4.3) to the attacker, who chooses the input psig_i, the reduction simply samples a nonce nonce_i uniformly, forwards the query $(\mathsf{pk}_{R_i}, \mathsf{nonce}_i)$ to the challenger. It returns the challenger's oracle response psig'_i as the corresponding psig_i along with nonce_i to the attacker. At the end the reduction outputs the attacker's forgery as its own.

B Knowledge of Secret Key Assumption

Our claim is that any NIBS scheme that is secure in the KOSK model can be generically upgraded to a fully secure scheme, ie., one without the KOSK assumption using a NIZKAoK. We show this formally by building a secure NIBS scheme (without KOSK) given a NIBS scheme, $\mathsf{NIBS}' = (\mathsf{Setup}', \mathsf{KeyGen}'_S, \mathsf{KeyGen}'_R, \mathsf{Issue}', \mathsf{Obtain}', \mathsf{Verify}')$ that is secure in the KOSK model.

Language $\mathcal{L}'$

Instance: Each instance x is interpreted as a public key pk' and a bit $b \in \{0, 1\}$.
Witness: Witness ω consists of a secret key sk', and randomness $\rho \in \{0, 1\}^\lambda$.
Membership: Let C_0, C_1 be two circuits encoding $\mathsf{KeyGen}'_S(\mathsf{pp}', 1^\lambda; \cdot)$ and $\mathsf{KeyGen}'_R(\mathsf{pp}', 1^\lambda; \cdot)$ respectively. Then ω is a valid witness for x if the following is satisfied:

- $(\mathsf{sk}', \mathsf{pk}') = C_b(\rho)$

$\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$. The setup algorithm generates $\mathsf{pp}' \leftarrow\!\!\$\ \mathsf{Setup}'(1^\lambda)$, and then runs the setup algorithms for NIZK (for language $\mathcal{L}'$) to generate $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(\mathsf{pp}', 1^\lambda)$ and outputs it as the public parameter pp of the protocol.

$\mathsf{KeyGen}_S(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{vk})$. The signer's setup algorithm samples a random value $r_S \leftarrow\!\!\$\ \{0, 1\}^\lambda$ and runs $(\mathsf{sk}', \mathsf{pk}') \leftarrow \mathsf{KeyGen}'_S(\mathsf{pp}', 1^\lambda; r_S)$. It creates a NIZK proof $\pi_S \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x := (\mathsf{pk}', 0), \omega := (\mathsf{sk}', r_S))$ for the language $\mathcal{L}'$. It outputs $\mathsf{sk} := \mathsf{sk}'$ and $\mathsf{vk} := (\mathsf{pk}', \pi_S)$.

$\mathsf{KeyGen}_R(\mathsf{pp}) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. The receiver's setup algorithm samples a random value $r_R \leftarrow\!\!\$\ \{0,1\}^\lambda$ and runs $(\mathsf{sk}', \mathsf{pk}') = \mathsf{KeyGen}'_R(\mathsf{pp}', 1^\lambda; r_R)$. It creates a NIZK proof $\pi_S \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x := (\mathsf{pk}', 1), \omega := (\mathsf{sk}', r_R))$ for the language $\mathcal{L}'$. It outputs $\mathsf{sk} := \mathsf{sk}'$ and $\mathsf{vk} := (\mathsf{pk}', \pi_R)$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \to (\mathsf{psig}, \mathsf{nonce})$. The issue algorithm parses the receiver's public key as $(\mathsf{pk}', \pi) := \mathsf{pk}_R$ and runs the NIZK verifier $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x := (\mathsf{pk}', 1), \pi)$. If the verifier outputs 0, the signer aborts. Otherwise it continues to execute the issue algorithm.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, \mathsf{psig}, \mathsf{nonce}) \to (\mu, \sigma)$. The obtain algorithm parses the signer's public key as $(\mathsf{pk}', \pi) := \mathsf{vk}$ and runs the NIZK verifier $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x := (\mathsf{pk}', 0), \pi_S)$. If the verifier outputs 0, the receiver aborts. Otherwise it continues to execute the obtain algorithm.

$\mathsf{Verify}(\mathsf{vk}, \mu, \sigma) \to \{0,1\}$. The verification algorithm runs $\mathsf{Verify}'(\mathsf{vk}, \mu, \sigma)$ and outputs whatever it outputs.

It is intuitively obvious that if NIBS′ is a secure NIBS in the KOSK model and NIZK is an argument of knowledge, then the above construction NIBS = (Setup, KeyGen_S, KeyGen_R, Issue, Obtain, Verify) is secure in the standard model. Essentially, in the security proof, instead of receiving the adversary's secret key (as in the KOSK model), the challenger must now run the NIZK extractor to obtain the adversary's secret. The rest of the proof would then proceed identically to that of NIBS′.

C Lattice-based Tagged NIBS

Let us begin this section by recalling the security definitions for TNIBS.

C.1 Tagged NIBS definitions

We also recall the definition for a tagged non-interactive blind signature (TNIBS) scheme. A TNIBS scheme has the same set of algorithms as a regular NIBS scheme, except with the following differences:

$\mathsf{Setup}, \mathsf{KeyGen}_S, \mathsf{KeyGen}_R$ are defined as for a regular NIBS scheme.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \tau) \to (\mathsf{psig}, \mathsf{nonce})$. The issue algorithm now takes a tag $\tau \in \mathcal{T}$ as an additional input, rest is as before.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, (\mathsf{psig}, \mathsf{nonce}), \tau) \to (\mu, \sigma)$. The obtain algorithm also takes a tag τ as an additional input, with the rest of the syntax being same as before.

$\mathsf{Verify}(\mathsf{vk}, \mu, \tau, \sigma) \to \{0,1\}$. The verification algorithm also takes a tag τ as an additional input.

As with a NIBS scheme, tagged NIBS is required to satisfy the following security properties.

Definition C.1 (One-more unforgeability)**.** A TNIBS scheme $\mathcal{S}$ satisfies one-more unforgeability, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that

for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{l} \bigwedge_{i\in[\ell+1]} \mathsf{Verify}(\mathsf{vk},\mu_i,\tau_i,\sigma_i)=1 \\ \wedge \left(\bigwedge_{i\neq j\in[\ell+1]} \mu_i \neq \mu_j\right) \end{array} : \begin{array}{r} \mathsf{pp} \leftarrow\!\$\, \mathsf{Setup}(1^\lambda) \\ (\mathsf{sk},\mathsf{vk}) \leftarrow\!\$\, \mathsf{KeyGen}_S(\mathsf{pp}) \\ (\{(\tau_i,\mu_i,\sigma_i)\}_{i=1}^{\ell+1}) \leftarrow\!\$\, \mathcal{A}^{O_{\mathsf{sk}}(\cdot,\cdot)}(\mathsf{vk}) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

where $O_{\mathsf{sk}}(\cdot,\cdot)$ takes as input a receiver's public key pk_{R_i} and a tag τ, and outputs a presignature-nonce pair $(\mathsf{psig}_i, \mathsf{nonce}_i)$ by running $\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_{R_i}, \tau)$, and $\mathcal{A}$ is an admissible adversary iff $\mathcal{A}$ makes at most ℓ queries to O_{sk}.

Definition C.2 (Receiver blindness)**.** A TNIBS scheme $\mathcal{S}$ satisfies receiver blindness, if for every *stateful admissible* PPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, the following holds:

$$\Pr\left[\begin{array}{l} \mathcal{A}(\mu_{\hat{b}},\sigma_{\hat{b}},\mu_{1-\hat{b}},\sigma_{1-\hat{b}})=\hat{b}: \\ \qquad\qquad \mathsf{pp} \leftarrow\!\$\, \mathsf{Setup}(1^\lambda),\ \hat{b} \leftarrow\!\$\, \{0,1\}, \\ \qquad\qquad \forall b\in\{0,1\}:\ (\mathsf{sk}_{R_b},\mathsf{pk}_{R_b}) \leftarrow\!\$\, \mathsf{KeyGen}_R(\mathsf{pp}) \\ \qquad\qquad (\mathsf{vk},(\mathsf{psig}_b,\mathsf{nonce}_b),\tau) \leftarrow\!\$\, \mathcal{A}(\mathsf{pk}_{R_0},\mathsf{pk}_{R_1}) \\ \qquad\qquad \forall b\in\{0,1\}:\ (\mu_b,\sigma_b) \leftarrow\!\$\, \mathsf{Obtain}(\mathsf{sk}_{R_b},\mathsf{vk},(\mathsf{psig}_b,\mathsf{nonce}_b),\tau) \end{array}\right] \leq \frac{1}{2} + \mathsf{negl}(\lambda),$$

where $\mathcal{A}$ is an admissible adversary iff $\sigma_0, \sigma_1 \neq \perp$ (i.e., Obtain algorithm does not abort).

Definition C.3 (Reusability)**.** A TNIBS scheme $\mathcal{S}$ satisfies the reusability property, if there exists a negligible function $\mathsf{negl}(\cdot)$ such that for every $\lambda \in \mathbb{N}$, every tag $\tau \in \mathcal{T}$, the following holds:

$$\Pr\left[\mu_0=\mu_1 : \begin{array}{c} \mathsf{pp} \leftarrow\!\$\, \mathsf{Setup}(1^\lambda) \\ (\mathsf{sk},\mathsf{vk}) \leftarrow\!\$\, \mathsf{KeyGen}_S(\mathsf{pp}), (\mathsf{sk}_R,\mathsf{pk}_R) \leftarrow\!\$\, \mathsf{KeyGen}_R(\mathsf{pp}) \\ \forall b\in\{0,1\}: (\mathsf{psig}_b,\mathsf{nonce}_b) \leftarrow\!\$\, \mathsf{Issue}(\mathsf{sk},\mathsf{pk}_R,\tau) \\ \forall b\in\{0,1\}: (\mu_b,\sigma_b) \leftarrow\!\$\, \mathsf{Obtain}(\mathsf{sk}_R,\mathsf{vk},(\mathsf{psig}_b,\mathsf{nonce}_b),\tau) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

C.2 Construction

We now upgrade our NIBS scheme to a tagged NIBS. Let parameters $n, m, \varsigma, \beta = \varsigma\sqrt{m}$ and prime q be such that the randomized one-more ISIS instance $\mathsf{rOM\text{-}ISIS}_{q,n,2m,\varsigma,(3\sqrt{2}+\epsilon)\beta}$ is hard, where $\epsilon = \mathcal{O}(m^{-1/2})$. These parameters must satisfy the constraints in (1).

Tools required. The construction relies on a public key encryption scheme $\mathsf{PKE} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, a lattice trapdoor $\mathsf{bLT} = (\mathsf{TrapGen}, \mathsf{SamplePre})$, and a NIZK scheme $\mathsf{NIZK} = (\mathsf{Setup}, \mathsf{Prove}, \mathsf{Verify})$ for the following language:

Language $\mathcal{L}_4$

Instance: Each instance x is interpreted as matrices $\mathbf{C}, \mathbf{A}$ and $\mathbf{B}$, PKE public key pke.pk, ciphertext ct, vector $\mathbf{w}$, random string δ and the tag τ.

Witness: Witness ω consists of a secret vector $\mathbf{x}$, nonce vector $\mathbf{y}$, presignature vector $\mathbf{z}$ and randomness r.

Membership: ω is a valid witness for x if the following are satisfied:

- $\|\mathbf{z}\| \leq \sqrt{2}\beta$ and $\|\mathbf{x}\| \leq \sqrt{2}\beta/m$ and $\mathbf{y} \in \{\pm 1\}^{2m}$.
- $\mathsf{ct} = \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}\|\mathbf{z}; r)$
- $\mathbf{C}\cdot\mathbf{z} + \mathbf{B}\cdot\mathbf{y} = \mathbf{A}\cdot\mathbf{x} + \mathsf{H}_1(\delta) + \mathsf{H}_2(\tau)$ and $\mathbf{w} = \mathbf{A}_{\mathsf{L}}\cdot\mathbf{x}_{\perp} + \mathbf{A}_{\mathsf{R}}\cdot\mathbf{z}_{\perp}$

Below we describe our tagged non-interactive blind signature scheme. Both, completeness and reusability of the above construction follows from the completeness and reusability of construction 7.1.

$\mathsf{Setup}(1^\lambda, n, m, q, \varsigma, \mathsf{H}_1, \mathsf{H}_2) \to \mathsf{pp}$. It is exactly the same as the Setup algorithm for Construction 7.1, except for the following: It takes as input two hash functions H_1 and H_2, which are used as random oracles in the design. It outputs the public parameters as

$$\mathsf{pp} := (\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs}).$$

$\mathsf{KeyGen}_S(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{vk})$. Signer's key generator algorithm is exactly the same as KeyGen_S in Construction 7.1.

$\mathsf{KeyGen}_R(\mathsf{pp}) \to (\mathsf{sk}_R, \mathsf{vk}_R)$. It first samples $\mathbf{x} \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}_q^{2m}, \varsigma/m}$ and $\delta \leftarrow\!\!\$\ \{0,1\}^\lambda$. Next, it computes $\mathbf{t} = \mathbf{A}\cdot\mathbf{x} + \mathsf{H}_1(\delta)$. It outputs user's secret key and public key as $(\mathsf{sk}_R := (\mathbf{x}, \delta), \mathsf{pk}_R := \mathbf{t})$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \tau) \to (\mathsf{psig}, \mathsf{nonce})$. Same as the Issue algorithm in Construction 7.1, except for the following: Instead of setting $\mathbf{t}$ as pk_R, it sets $\mathbf{t} = \mathsf{pk}_R + \mathsf{H}_2(\tau)$.

Next, it still sets $\mathbf{z}$ as $\mathsf{bLT.SamplePre}(\mathbf{C}, \mathbf{T}_{\mathbf{C}}, \mathbf{t} - \mathbf{B}\cdot\mathbf{y}, \varsigma)$. It outputs presignature $\mathsf{psig} := \mathbf{z}$ and nonce $\mathsf{nonce} := \mathbf{y}$.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, \mathsf{psig}, \mathsf{nonce}, \tau) \to (\mu, \sigma, \tau)$. It parses sk_R as $(\mathbf{x}, \delta)$. Then, it assigns $\mathbf{C} := \mathsf{vk}$, $\mathbf{y} := \mathsf{nonce}$, and $\mathbf{z} := \mathsf{psig}$. It first checks if $\mathbf{C}\cdot\mathbf{z} + \mathbf{B}\cdot\mathbf{y} = \mathbf{A}\cdot\mathbf{x} + \mathsf{H}_1(\delta) + \mathsf{H}_2(\tau)$ and $\|\mathbf{z}\| \leq \sqrt{2}\beta$ and $\mathbf{y} \in \{\pm 1\}^{2m}$. If any check fails then it aborts and outputs $\perp$.

Otherwise, it generates $\mathsf{ct} \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}\|\mathbf{y}\|\mathbf{z}; r)$ from uniformly sampled randomness $r \leftarrow\!\!\$\ \{0,1\}^\lambda$. The obtain algorithm sets $\mathbf{w}$ as $\mathbf{w} = \mathbf{A}\cdot\begin{bmatrix}\mathbf{x}_\perp \\ \mathbf{z}_\perp\end{bmatrix}$, where $\mathbf{x}_\perp, \mathbf{z}_\perp \in \mathbb{Z}_q^m$ and generates NIZK proof π as $\pi \leftarrow \mathsf{NIZK.Prove}\Big(\mathsf{nizk.crs}, x := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}, \mathbf{w}, \delta, \tau), \omega := (\mathbf{x}, \mathbf{y}, \mathbf{z}, r)\Big)$. Finally, it outputs message $\mu := (\mathbf{w}, \delta)$, signature $\sigma := (\pi, \mathsf{ct})$, and tag τ.

$\mathsf{Verify}(\mathsf{vk}, \mu, \tau, \sigma) \to \{0,1\}$. It parses μ as $(\mathbf{w}, \delta)$ and σ as (π, ct). The verification algorithm accepts and outputs 1 if and only if $\mathsf{NIZK.Verify}\Big(\mathsf{nizk.crs}, x := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}, \mathbf{w}, \delta, \tau), \pi\Big) = 1$. Otherwise, it outputs 0.

Completeness. Observe that by correctness of bLT.SamplePre, $\mathbf{z}$ is a presignature on $\mathbf{A}\cdot\mathbf{x}+\mathsf{H}_1(\delta)+\mathsf{H}_2(\tau)-\mathbf{B}\cdot\mathbf{y}$, ie., $\mathbf{C}\cdot\mathbf{z}=\mathbf{A}\cdot\mathbf{x}+\mathsf{H}_1(\delta)+\mathsf{H}_2(\tau)-\mathbf{B}\cdot\mathbf{y}$ such that $\|\mathbf{z}\|\leq\sqrt{2}\beta$ and $y_i\in\{\pm 1\}$ for all $i\in[2m]$ where y_i is the i^{th} element of $\mathbf{y}$. Since, it further holds by construction that $\mathbf{w}=\mathbf{A}_{\mathsf{L}}\cdot\mathbf{x}_{\perp}+\mathbf{A}_{\mathsf{R}}\cdot\mathbf{z}_{\perp}$, $\mathsf{ct}=\mathsf{PKE.Enc}(\mathsf{pke.pk},\mathbf{x}\|\mathbf{y}\|\mathbf{z};r)$, and (by Lemma 3.1) $\|\mathbf{x}\|\leq\sqrt{2}\beta/m$, we have that π is a valid NIZK proof for the language $\mathcal{L}_4$. It therefore follows from the completeness of the underlying NIZK that Construction C.2 is complete.

Reusability. The reusability of the tagged NIBS protocol can be shown identically as that of the (un-tagged) NIBS protocol 7.1.

C.3 One-more unforgeability

We first prove the one-more unforgeability of this protocol.

Theorem C.4. Assume that NIZK proof system NIZK satisfies soundness, lattice trapdoor bLT satisfies well-distributedness, and the rOM-ISIS assumption holds, then our construction C.2 is one-more unforgeable.

Proof. We present the following hybrids:

Hybrid_0 This is the actual one-more unforgeability game for tagged NIBS:

1. Challenger samples $\mathbf{A},\mathbf{B}\leftarrow\$\,\mathbb{Z}_q^{n\times 2m}$ uniformly at random, $\mathsf{pke.pk}\leftarrow\$\,\mathsf{PKE.KeyGen}(1^\lambda)$, $\mathsf{nizk.crs}\leftarrow\$\,\mathsf{NIZK.Setup}(1^\lambda)$ and sets pp as $(\mathbf{A},\mathbf{B},\mathsf{pke.pk},\mathsf{nizk.crs})$. Next, it samples $(\mathbf{T}_{\mathbf{C}},\mathbf{C})\leftarrow\$\,\mathsf{bLT.TrapGen}(1^\lambda,n,2m,q)$ and sets $\mathsf{sk}:=\mathbf{T}_{\mathbf{C}}$, $\mathsf{vk}:=\mathbf{C}$. Challenger sends pp and vk to the adversary.

2. Adversary chooses pk_R and tag τ, and requests a presignature, to which the challenger replies with $(\mathsf{psig},\mathsf{nonce}):=\mathsf{Issue}(\mathsf{sk},\mathsf{pk}_R,\tau)$. The adversary repeats this step a total of ℓ times. In this step, adversary is allowed to make hash queries to H_1 and H_2, which are considered as random oracle queries in our design.

3. Adversary outputs tag τ and k message and signature pairs $((\tau_1,\mu_1,\sigma_1),\ldots,(\tau_k,\mu_k,\sigma_k))$. Adversary wins if $\mu_i\neq\mu_j$ for $1\leq i<j\leq k$, $\mathsf{Verify}(\mathsf{vk},\mu_i,\sigma_i,\tau_i)=1$ for all $i\in[k]$, and $k=\ell+1$.

Hybrid_1 This is the same as Hybrid_0 aside from one key difference that instead of uniformly sampling the matrix $\mathbf{A}$, it is chosen by first sampling a matrix $\mathbf{R}\leftarrow\$\,\{0,1\}^{2m\times 2m}$ and then setting $\mathbf{A}=\mathbf{C}\cdot\mathbf{R}$.

1. Challenger samples $\mathbf{R}\leftarrow\$\,\{0,1\}^{2m\times 2m}$ uniformly at random and sets $\mathbf{A}=\mathbf{C}\cdot\mathbf{R}$. Challenger then samples $\mathbf{B}\leftarrow\$\,\mathbb{Z}_q^{n\times 2m}$, $\mathsf{pke.pk}\leftarrow\$\,\mathsf{PKE.Setup}(1^\lambda)$, and sets pp as $(\mathbf{A},\mathbf{B},\mathsf{pke.pk},\mathsf{nizk.crs})$. Next, it samples $\mathbf{T}_{\mathbf{C}},\mathbf{C}\leftarrow\$\,\mathsf{bLT.TrapGen}(1^\lambda,n,2m,q)$ and sets $\mathsf{sk}:=\mathbf{T}_{\mathbf{C}}$, $\mathsf{vk}:=\mathbf{C}$. Challenger sends pp and vk to the adversary.

Hybrid_2 This is exactly the same as Hybrid_1 except the following: Whenever $\mathcal{A}$ makes a fresh hash query to H_2, instead of responding with a uniformly random output, challenger instead outputs $\mathbf{C}\cdot\mathbf{r}$ where $\mathbf{r}\leftarrow\$\,\mathcal{D}_{\mathbb{Z}_q^{2m},\varsigma/m}$.

2. Adversary chooses pk_R and tag τ, and requests a presignature, to which the challenger replies with $(\mathsf{psig}, \mathsf{nonce}) := \mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R, \tau)$. The adversary repeats this step a total of ℓ times. In this step, adversary is allowed to make hash queries to H_1 and H_2. When adversary makes fresh hash queries to H_1, challenger replies with freshly sampled random vectors in $\mathbb{Z}_q^{2m}$. When adversary makes fresh hash queries to H_2, challenger samples $\mathbf{r} \leftarrow\$ \mathcal{D}_{\mathbb{Z}_q^{2m}, \varsigma/m}$ and replies with $\mathbf{C} \cdot \mathbf{r}$.

Let $\mathsf{Adv}^i_{\mathcal{A}}$ denote the advantage of an adversary $\mathcal{A}$ in the TNIBS one-more unforgeability game in Hybrid_i. Then, the following must hold:

Lemma C.5. For any PPT adversary $\mathcal{A}$, $|\mathsf{Adv}^0_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$

Proof. The proof is omitted as it is identical to the proof of Lemma 7.2. □

Lemma C.6. For any PPT adversary $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$

Proof. By leftover hash lemma (Corollary 3.8), the following holds:

$$\{(\mathbf{C}', \mathbf{C}' \cdot \mathbf{r}) : \mathbf{C}' \leftarrow\$ \ \mathbb{Z}_q^{n \times 2m}, \mathbf{r} \leftarrow\$ \mathcal{D}_{\mathbb{Z}_q^{2m}, \varsigma/m}\} \approx_s (\mathbf{C}', \mathbf{v}) : \mathbf{C}' \leftarrow\$ \ \mathbb{Z}_q^{n \times 2m}, \mathbf{v} \leftarrow\$ \mathbb{Z}_q^n\}\}$$

By Corollary 3.8, $\mathbf{C}'$ is statistically close to $\mathbf{C}$ and thus $\mathbf{C}' \cdot \mathbf{r}$ is indistinguishable from a randomly sampled vector. □

Lemma C.7. Assume that NIZK argument is sound and rOM-ISIS assumption holds, then the one-more-unforgeability adversary $\mathcal{A}$ can have at most negligible advantage in Hybrid_2.

Proof. Suppose there exists a PPT attacker $\mathcal{A}$ that wins the one-more unforgeability game with non-negligible advantage $\epsilon = \epsilon(\lambda)$. It outputs a signature and k message and signature pairs $((\tau_1, \mu_1, \sigma_1), \ldots, (\tau_k, \mu_k, \sigma_k))$, and wins if and only if $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, $\mathsf{Verify}(\mathsf{vk}, \mu_i, \tau_i, \sigma_i) = 1$ for $i \in [k]$ and $k = \ell + 1$. Consider that we parse the input and signature pairs (μ_i, σ_i) as $((\omega_i, \delta_i), (\pi_i, \mathsf{ct}_i))$ for all $i \in [k]$. Divide the potential attacker into the following types:

- Type 1: This is nearly identical to the type 1 attacker in the proof of Lemma 7.3, except that instance x^* is defined as $x^* := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_j, \mathbf{w}_j, \delta_j, \tau_j)$.
- Type 2: This is a natural extension from type 2 attacker in Lemma 7.3. For type 2 attacker, there exists some $j \in [k]$, such that $\mathbf{C} \cdot \mathbf{z}_j + \mathbf{B} \cdot \mathbf{y}_j = \mathbf{A} \cdot \mathbf{x}_j + \mathsf{H}_1(\delta_j) + \mathsf{H}_2(\tau_j)$, and $\mathcal{A}$ only queries at most one of $\mathsf{H}_1(\delta_j)$ and $\mathsf{H}_2(\tau_j)$.
- Type 3: Definition of type 3 attacker follows from type 3 attacker in Lemma 7.3. We still require the following: For all $i \in [k]$, $x_i := (\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_i, \mathbf{w}_i, \delta_i, \tau_i)$ is a valid instance for language $\mathcal{L}_4$. $\mathcal{A}$ queries $\mathsf{H}_1(\delta)$ at least once for each fresh δ in the message $(\mathbf{w}, \delta)$ it outputs. Additionally, it makes at least one hash query to $\mathsf{H}_2(\tau)$ for each fresh tag τ.

Claim C.8. Assuming that NIZK satisfies soundness property, then type 1 attacker has at most negligible advantage.

Proof. Omitted as it is nearly identical to the proof of Claim 7.4. □

Claim C.9. Type 2 attacker $\mathcal{A}$ has at most negligible advantage.

Proof. Following from the proof of Claim 7.5, $\mathsf{H}_1(\delta_j)$ and $\mathsf{H}_2(\tau_j)$ look random in the view of $\mathcal{A}$. Thus without querying $\mathsf{H}_1(\delta_j)$ or $\mathsf{H}_2(\tau_j)$, $\mathcal{A}$ has at most negligible advantage in finding both δ_j and τ_j such that $\mathsf{H}_1(\delta_j) + \mathsf{H}_2(\tau_j) = \mathbf{C} \cdot \mathbf{z}_j + \mathbf{B} \cdot \mathbf{y}_j - \mathbf{A} \cdot \mathbf{x}_j$. □

Claim C.10. Assuming that rOM-ISIS assumption holds, then type 3 attacker $\mathcal{A}$ has at most negligible advantage.

Proof. If type 3 attacker $\mathcal{A}$ has non-negligible advantage, then we build reduction algorithm $\mathcal{B}$ that breaks the rOM-ISIS security game. Here, $\mathcal{B}$ is nearly identical to the reduction algorithm in the proof of Claim 7.6, except for the following: When $\mathcal{A}$ makes a hash query to H_1 using fresh input δ, $\mathcal{B}$ makes a syndrome query to the rOM-ISIS challenger. It forwards the output from the challenger to $\mathcal{A}$ as the output of $\mathsf{H}_1(\delta)$. When $\mathcal{A}$ makes a hash query $\mathsf{H}_2(\tau_i)$ using fresh input τ_i, $\mathcal{B}$ samples a random vector $\mathbf{r}_i \leftarrow\!\!\$\ \mathcal{D}_{\mathbb{Z}_q^{2m}, \varsigma/m}$ and sets $\mathsf{H}_2(\tau_i)$ as $\mathbf{C} \cdot \mathbf{r}_i$. $\mathcal{A}$ wins the one-more unforgeability game by outputting $((\tau_1, \mu_1, \sigma_1), \ldots, (\tau_k, \mu_k, \sigma_k))$ such that $\mu_i \neq \mu_j$ for $1 \leq i < j \leq k$, and $\mathsf{Verify}(\mathsf{pp}, \mathsf{vk}, \mu_i, \tau_i, \sigma_i) = 1$ for all $i \in [k]$.

Consider that for all $i \in [k]$, $\mathcal{B}$ decrypts ct_i using $\mathsf{pke.sk}$ and obtains $\mathbf{x}_i$ and $\mathbf{z}_i$ following from the proof of Claim 7.6, we have:

$$\mathsf{H}_1(\delta_i) = \mathbf{C} \cdot \mathbf{z}_i + \mathbf{B} \cdot \mathbf{y}_i - \mathbf{A} \cdot \mathbf{x}_i - \mathsf{H}_2(\tau_i) = \mathbf{C} \cdot (\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i - \mathbf{r}_i) + \mathbf{B} \cdot \mathbf{y}_i.$$

Since $\|\mathbf{z}_i\| \leq \sqrt{2}\beta$, $\|\mathbf{x}_i\| \leq \sqrt{2}\beta/m$, $\|\mathbf{r}_i\| \leq \sqrt{2}\beta/m$, and $\mathbf{R} \in \{0,1\}^{2m \times 2m}$,

$$\|\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i - \mathbf{r}_i\| \leq 3\sqrt{2}\beta + \sqrt{2}\beta/m.$$

Following from the case by case argument of Claim 7.6, $\mathcal{B}$ wins the randomized one-more ISIS game with non-negligible advantage by sending back $\mathbf{z}_i - \mathbf{R} \cdot \mathbf{x}_i - \mathbf{r}_i$ and $\mathbf{y}_i$ for $i \in [k]$. □

Any successful one-more unforgeability attacker $\mathcal{A}$ must be one of type 1, 2, 3, 4. Thus the lemma follows. □

Combining the above lemmas, our main theorem follows.

□

C.4 Receiver blindness

Theorem C.11. Assume that NIZK proof system NIZK satisfies zero knowledge property, and public key encryption scheme PKE is IND-CPA secure, then the tagged NIBS construction C.2 is receiver blind.

Proof. Below are the hybrids:

Hybrid_0 This corresponds to the real experiment.

1. Challenger samples $\mathbf{A}, \mathbf{B} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times 2m}$ uniformly at random, $\mathsf{pke.pk} \leftarrow\!\!\$\ \mathsf{PKE.KeyGen}(1^\lambda)$, and $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$. Challenger then sets pp as $(\mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{nizk.crs}, \mathsf{H}_1, \mathsf{H}_2)$ and sends pp to the adversary.

2. Next, challenger samples $(\mathsf{sk}_{R_b}, \mathsf{pk}_{R_b}) \leftarrow \mathsf{KeyGen}_R(\mathsf{pp})$, for $b \in \{0,1\}$. The challenger sends pk_{R_0} and pk_{R_1} to $\mathcal{A}$.

3. The adversary outputs a matrix $\mathbf{C} \in \mathbb{Z}_q^{n \times 2m}$ as the verification key vk, two presignatures and nonces $(\mathsf{psig}_0, \mathsf{nonce}_0)$ and $(\mathsf{psig}_1, \mathsf{nonce}_1)$, and tag τ.

4. The challenger then generates message and signature pairs:

   (a) For $b \in \{0,1\}$, challenger sets $\mathbf{z}_b$ as psig_b, $\mathbf{y}_b$ as nonce_b, and parses sk_{R_b} as $(\mathbf{x}_b, \delta_b)$. It then checks whether $\mathbf{C} \cdot \mathbf{z}_b + \mathbf{B} \cdot \mathbf{y}_b = \mathbf{A} \cdot \mathbf{x}_b + \mathsf{H}_1(\delta_b) + \mathsf{H}_2(\tau)$ and $\|\mathbf{z}_b\| \leq \sqrt{2}\beta$ and $\mathbf{y}_b \in \{\pm 1\}^{2m}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

   (b) Otherwise, for $b \in \{0,1\}$, challenger continues to generate $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_b \| \mathbf{y}_b \| \mathbf{z}_b; r_b)$ with freshly sampled randomness r_0 and r_1.

   (c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_{b,\perp} \\ \mathbf{z}_{b,\perp} \end{bmatrix}$, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b, \tau)$, witness ω_b as $(\mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b, r_b)$, and generates proof $\pi_b \leftarrow \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_b, \omega_b)$.

   (d) Next, challenger sets message μ_b as $(\mathbf{w}_b, \delta_b)$ and signature σ_b as (π_b, ct_b). Finally, challenger samples $\hat{b} \leftarrow\!\!\$\ \{0,1\}$ uniformly at random and sends $(\mu_{\hat{b}}, \sigma_{\hat{b}}, \mu_{1-\hat{b}}, \sigma_{1-\hat{b}})$ to the adversary.

Hybrid_1 Instead of honestly generating the NIZK proofs π_0 and π_1, the challenger simulates π_0 and π_1 without any witness.

4.(c) For $b \in \{0,1\}$, it computes $\mathbf{w}_b = \mathbf{A} \cdot \begin{bmatrix} \mathbf{x}_{b,\perp} \\ \mathbf{z}_{b,\perp} \end{bmatrix}$ and sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathbf{B}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b)$. Without setting any witnesses, challenger generates π_b using NIZK simulator.

Hybrid_2 For $b \in \{0,1\}$, instead of generating $\mathsf{ct}_b \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{x}_b \| \mathbf{y}_b \| \mathbf{z}_b; r_b)$, challenger sets ct_b as $\mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_b)$.

4.(b) Challenger generates $\mathsf{ct}_0 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_0)$ and $\mathsf{ct}_1 \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathbf{0}; r_1)$ with freshly sampled randomness r_0 and r_1.

Hybrid_3 This the same as Hybrid_2, except that the challenger samples pk_{R_0} and pk_{R_1} uniformly at random.

2. When generating pk_{R_b} for $b \in \{0,1\}$, instead of setting it as $\mathbf{A} \cdot \mathbf{x}_b + \mathsf{H}_1(\delta_b) + \mathsf{H}_2(\tau)$, the challenger sets it as a uniformly sampled vector, such that $\mathsf{pk}_{R_b} \leftarrow\!\!\$\ \mathbb{Z}_q^n$.

Hybrid_4 This is the same as Hybrid_3, except that the challenger samples vectors $\mathbf{w}_0, \mathbf{w}_1$ uniformly at random.

4.(c) For $b \in \{0,1\}$, it samples $\mathbf{w}_b$ uniformly at random, sets statement x_b as $(\mathbf{C}, \mathbf{A}, \mathsf{pke.pk}, \mathsf{ct}_b, \mathbf{w}_b, \delta_b, \tau)$, and generates π_b using NIZK simulator.

Lemma C.12. Assuming zero-knowledge property of NIZK, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^0_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by extending the proof of Lemma 7.9 to Construction C.2. □

Lemma C.13. If public key encryption scheme PKE is an IND-CPA secure public key encryption scheme, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by extending the proof of Lemma 7.10. □

Lemma C.14. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^3_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (ommited) follows by extending the proof of Lemma 7.11. □

Lemma C.15. For all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^4_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (ommited) follows by extending the proof of Lemma 7.12. □

Since the adversary has 0 advantage in Hybrid_4, Construction C.2 satisfies recipient blindness. □

D NIBS from General Purpose NIZKs

Tools required. The construction relies on a pseudorandom function F, a perfectly binding commitment scheme $\mathsf{COM} = (\mathsf{COM.Setup}, \mathsf{COM.Com}, \mathsf{COM.Verify})$, a standard signature scheme $\mathsf{S} = (\mathsf{S.Setup}, \mathsf{S.Sign}, \mathsf{S.Verify})$, and a NIZKAoK proof system $\mathsf{NIZK} = (\mathsf{NIZK.Setup}, \mathsf{NIZK.Prove}, \mathsf{NIZK.Verify})$ for the following language:

Language $\mathcal{L}_5$

Instance: Each instance x is interpreted as a verification key vk, crs for a commitment scheme com.crs, and a message m.

Witness: Witness ω consists of a signature $\mathsf{s}.\sigma$, randomness r, PRF key K, and commitment opening op.

Membership: ω is a valid witness for x if the following are satisfied:

- $\mathsf{s}.\sigma$ is a valid signature for message '$c\|r$' under key vk, where $c = \mathsf{COM.Com}(\mathsf{com.crs}, K; \mathsf{op})$. Namely,

$$\mathsf{S.Verify}(\mathsf{s.vk}, c\|r, \mathsf{s}.\sigma) = 1$$

- m is a PRF evaluation of r w.r.t. key K. Namely, $m = F_K(r)$.

D.1 Construction

Below we describe our non-interactive blind signature scheme NIBS.

$\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$. It runs the setup algorithms for NIZK (for language $\mathcal{L}_5$) and COM schemes. Namely, it generates

$$\mathsf{nizk.crs} \leftarrow\!\$\, \mathsf{NIZK.Setup}(1^\lambda), \quad \mathsf{com.crs} \leftarrow\!\$\, \mathsf{COM.Setup}(1^\lambda).$$

It outputs $\mathsf{pp} = (\mathsf{nizk.crs}, \mathsf{com.crs})$.

$\mathsf{KeyGen}_S(\mathsf{pp}) \to (\mathsf{sk}, \mathsf{vk})$. The signer's setup algorithm runs the setup algorithm for the signature scheme S. Namely, it generates keys as

$$(\mathsf{sk}, \mathsf{vk}) \leftarrow\!\!\$\ \mathsf{S.Setup}(1^\lambda).$$

$\mathsf{KeyGen}_R(\mathsf{pp}) \to (\mathsf{sk}_R, \mathsf{pk}_R)$. The receiver's setup algorithm first samples a random PRF key $K \leftarrow\!\!\$\ \{0,1\}^\lambda$ and commitment randomness $\mathsf{op} \leftarrow\!\!\$\ \{0,1\}^\lambda$. Next, it computes a commitment as $c = \mathsf{COM.Com}(\mathsf{com.crs}, K; \mathsf{op})$. Finally, it outputs receiver's secret key and public key as $\mathsf{sk}_R := (K, \mathsf{op})$ and $\mathsf{pk}_R := c$.

$\mathsf{Issue}(\mathsf{sk}, \mathsf{pk}_R) \to (\mathsf{psig}, \mathsf{nonce})$. The issue algorithm samples a random message $r \leftarrow\!\!\$\ \{0,1\}^\lambda$, and creates a signature of message '$\mathsf{pk}_R \| r$'. Namely,

$$\mathsf{s}.\sigma \leftarrow\!\!\$\ \mathsf{S.Sign}(\mathsf{sk}, \mathsf{pk}_R \| r).$$

Finally, it outputs the pre-signature as $\mathsf{psig} := \mathsf{s}.\sigma$ and $\mathsf{nonce} := r$.

$\mathsf{Obtain}(\mathsf{sk}_R, \mathsf{vk}, \mathsf{psig}, \mathsf{nonce}) \to (m, \sigma)$. The receiver first computes the message as $m = F_K(r)$, where $(K, \mathsf{op}) := \mathsf{sk}_R$, $\mathsf{s}.\sigma := \mathsf{psig}$, and $r := \mathsf{nonce}$. It then runs the NIZK prover to create the signature σ as a NIZK proof as follows:

$$\sigma \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x = (\mathsf{vk}, \mathsf{com.crs}, m), \omega = (\mathsf{s}.\sigma, r, K, \mathsf{op})).$$

It sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, K; \mathsf{op})$. If $\mathsf{s}.\sigma$ is not a valid signature for $c \| r$, then it aborts and outputs $\perp$. Otherwise, it outputs the above message-signature/proof pair.

$\mathsf{Verify}(\mathsf{vk}, m, \sigma) \to \{0,1\}$. The verification algorithm runs the NIZK verifier and outputs whether $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x = (\mathsf{vk}, \mathsf{com.crs}, m), \sigma) = 1$.

Completeness. Observe that by correctness of the signature scheme, S, $\mathsf{s}.\sigma$ is a valid (pre)signature on $\mathsf{pk}_R \| r$. Also, by correctness of the commitment scheme COM, $\mathsf{pk}_R := c$ is a valid commitment to PRF key K with respect to some fixed opening op. Similarly, correctness of the PRF F ensures that $m = F_K(r)$. Thus, we have that σ is a valid NIZK proof for the relation language $\mathcal{L}_5$ described directly above, and follows from the completeness of the underlying NIZK that Construction D.1 is complete.

Reusability. For $b \in \{0,1\}$, if a signer issues presignature-nonce pairs $(\mathsf{s}.\sigma_b, r_b)$ to a given receiver with $\mathsf{pk}_R := c$, then

$$\begin{aligned}
\Pr[r_0 = r_0 \vee \mu_0 = \mu_1] = \ & \Pr[\mu_0 = \mu_1] + \Pr[r_0 = r_1] - \Pr[\mu_0 = \mu_1 \mid r_0 = r_1] \cdot \Pr[r_0 = r_1] \\
= \ & \Pr[c \| r_0 = c \| r_1] + \Pr[r_0 = r_1] - \Pr[c \| r_0 = c \| r_1 \mid r_0 = r_1] \cdot \Pr[r_0 = r_1] \\
= \ & \Pr[r_0 = r_1] = \mathsf{negl}(\lambda) \ .
\end{aligned}$$

So the construction is reusable.

D.2 One-more unforgeability

Theorem D.1. Assume that NIZK scheme NIZK is an argument of knowledge, and the signature scheme S is secure, then our NIBS construction D.1 is one-more unforgeable.

Proof. This follows from a case by case reduction algorithm.

- Type 1: Consider that type 1 attacker $\mathcal{A}$ outputs $\ell+1$ message and signature pairs $((m_1,\sigma_1), \ldots, (m_{\ell+1},\sigma_{\ell+1}))$. Set $x_i := \mathsf{vk}, \mathsf{com.crs}, m_i$, for all $i \in [\ell+1]$. For type 1 attacker $\mathcal{A}$, there exists some $j \in [\ell+1]$, such that $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x_j, \sigma_j) = 1$ and $\omega = \mathcal{E}(\tau, x_j, \pi_j)$ is not a valid witness for some instance x_j of $\mathcal{L}_5$.
- Type 2: Type 2 attacker is defined as the complementary of type 1 attacker, such that $\omega = \mathcal{E}(\tau, x_j, \pi_j)$ is a valid witness for all $j \in [\ell+1]$.

Claim D.2. Assuming that NIZK has a valid knowledge extractor $\mathcal{E}$, type 1 attacker $\mathcal{A}$ has at most negligible advantage.

Proof. Assume that type 1 adversary $\mathcal{A}$ has non-negligible advantage $\epsilon = \epsilon(\lambda)$, we build a reduction algorithm $\mathcal{B}$ that breaks the knowledge extractor of NIZK.

Challenger starts by generating $\tau \leftarrow\!\!\$\ \{0,1\}^\lambda$ uniformly at random, and then generates $\mathsf{nizk.crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda;\tau)$. It sends nizk.crs to $\mathcal{B}$. $\mathcal{B}$ then generates $\mathsf{com.crs} \leftarrow\!\!\$\ \mathsf{COM.Setup}(1^\lambda)$ and outputs $\mathsf{pp} := (\mathsf{nizk.crs}, \mathsf{com.crs})$. Next, $\mathcal{B}$ generates $(\mathsf{sk},\mathsf{vk}) \leftarrow\!\!\$\ \mathsf{KeyGen}_S(\mathsf{pp})$ and outputs vk. Next, $\mathcal{A}$ makes a series of presignature queries. For each query, $\mathcal{A}$ sends a public key pk_R to $\mathcal{B}$ and $\mathcal{B}$ sends back $(\mathsf{psig},\mathsf{nonce}) = \mathsf{Issue}(\mathsf{sk},\mathsf{pk}_R)$. Finally, $\mathcal{A}$ outputs $\ell+1$ message and signature pairs $((m_1,\sigma_1),\ldots,(m_{\ell+1},\sigma_{\ell+1}))$. Consider that we set $x_i := (\mathsf{vk},\mathsf{com.crs},m_i)$, for all $i \in [\ell+1]$. Then type 1 attacker $\mathcal{A}$ guarantees that there exists some $j \in [\ell+1]$, such that $\mathsf{NIZK.Verify}(\mathsf{nizk.crs}, x_j, \sigma_j) = 1$ and $\omega = \mathcal{E}(\tau, x_j, \pi_j)$ is not a valid witness for $x \in \mathcal{L}_5$. As such, $\mathcal{B}$ could break the knowledge extractor of NIZK using the output by $\mathcal{A}$. □

Claim D.3. Assuming signature scheme S is secure, type 2 attacker $\mathcal{A}$ has at most negligible advantage.

Assume that $\mathcal{A}$ has non-negligible advantage $\epsilon = \epsilon(\lambda)$, we build a reduction algorithm $\mathcal{B}$ that breaks the security of S with non-negligible advantage.

The challenger starts by generating $(\mathsf{sk},\mathsf{vk}) \leftarrow\!\!\$\ \mathsf{S.Setup}(1^\lambda)$, and it sends vk to reduction algorithm $\mathcal{B}$. Next, $\mathcal{B}$ samples $\mathsf{com.crs} \leftarrow\!\!\$\ \mathsf{COM.Setup}(1^\lambda)$ and $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$. $\mathcal{B}$ then outputs public parameter $\mathsf{pp} := (\mathsf{nizk.crs},\mathsf{com.crs})$ and signer's verification key vk.

Next, $\mathcal{A}$ is allowed to make a number of ℓ presignature queries, where in each query, $\mathcal{A}$ sends pk_R to $\mathcal{B}$. $\mathcal{B}$ then randomly samples $r \leftarrow\!\!\$\ \{0,1\}^\lambda$ and sends $\mathsf{pk}_R \| r$ to the challenger. Challenger then returns $\mathsf{s}.\sigma$ as the signature of $\mathsf{pk}_R \| r$. $\mathcal{B}$ then outputs presignature $\mathsf{psig} := \mathsf{s}.\sigma, r$ and $\mathsf{nonce} := r$.

To win the one-more unforgeability game, $\mathcal{A}$ outputs $((m_1,\sigma_1),\ldots,(m_{\ell+1},\sigma_{\ell+1}))$. For all $i \in [\ell+1]$, $\mathcal{B}$ sets x_i as $(\mathsf{vk},\mathsf{com.crs},m_i)$. Then, $\mathcal{B}$ extracts witness ω_i using NIZK knowledge extractor, such that $\omega_i = \mathcal{E}(\tau, x_i, \sigma_i)$. $\mathcal{B}$ parses ω_i as $(\mathsf{s}.\sigma_i, r_i, K_i, \mathsf{op}_i)$. Next, $\mathcal{B}$ sets c_i as $\mathsf{COM.Com}(\mathsf{com.crs}, K_i; \mathsf{op})$. Finally, $\mathcal{B}$ breaks the unforgeability property of signature scheme S by outputting message and signature pairs $(c_i \| r_i, \mathsf{s}.\sigma_i)$, for all $i \in [\ell+1]$.

We argue that such reduction algorithm $\mathcal{B}$, is a valid attacker against S security challenger. Recall that type 2 attacker $\mathcal{A}$ guarantees that $m_i = F_{K_i}(r_i)$. For all $1 \le i < j \le \ell+1$, $\mathcal{A}$ as a valid attacker guarantees that $m_i \neq m_j$, thus either $K_i \neq K_j$ or $r_i \neq r_j$. If $K_i \neq K_j$, then the perfect binding

property of our commitment scheme COM implies $c_i \neq c_j$. As a result, $c_i \| r_i \neq c_j \| r_j$. Type 2 attacker also guarantees that for all $i \in [\ell + 1]$, $\mathsf{S.Verify}(\mathsf{vk}, c_i \| r_i, \mathsf{s}.\sigma_i) = 1$, thus $\mathcal{B}$ is a successful attacker. □

D.3 Strong receiver blindness

Theorem D.4. Assume that general-purpose NIZK protocol NIZK satisfies zero-knowledge, commitment scheme COM is computationally hiding, and F is a secure PRF, then our NIBS construction D.1 satisfies strong receiver blindness.

Proof. Define the following hybrids: Hybrid_0 This corresponds to the original experiment:

1. Challenger starts by generating $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$ and $\mathsf{com.crs} \leftarrow\!\!\$\ \mathsf{COM.Setup}(1^\lambda)$. It outputs $\mathsf{pp} := (\mathsf{nizk.crs}, \mathsf{com.crs})$.

2. For $b \in \{0,1\}$, challenger samples PRF key $K_b \leftarrow\!\!\$\ \{0,1\}^\lambda$ and commitment randomness $\mathsf{op}_b \leftarrow\!\!\$\ \{0,1\}^\lambda$. Next, it computes $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, K_b; \mathsf{op}_b)$. Challenger sets sk_{R_b} as (K_b, op_b), and pk_{R_b} as c_b. It outputs pk_{R_0} and pk_{R_1}.

3. The adversary is allowed to make a series of k queries for k polynomially bounded in λ. The i-th query is explained as the following ($i \in [k]$):

   (a) In the i-th query, adversary chooses bit $b^{(i)}$, verifcation key $\mathsf{vk}^{(i)}$, presignature $\mathsf{psig}^{(i)}$ and nonce $\mathsf{nonce}^{(i)}$. Adversary sends them as the query's input to challenger.

   (b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. Challenger computes message as $m = F_{K_{b^{(i)}}}(r^{(i)})$. It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$ and $\omega := (\mathsf{s}.\sigma^{(i)}, r^{(i)}, K_{b^{(i)}}, \mathsf{op}_{b^{(i)}})$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, K_{b^{(i)}}; \mathsf{op}_{b^{(i)}})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

   (c) Otherwise, challenger runs NIZK prover to generate signature $\sigma \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x, \omega)$. It outputs m and σ.

4. Adversary then does the following:

   (a) Adversary chooses and outputs vk, $(\mathsf{psig}_b, \mathsf{nonce}_b)_b$ for $b \in \{0,1\}$.

   (b) For $b \in \{0,1\}$. Challenger sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. Challenger computes message as $m_b = F_{K_b}(r_b)$. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$ and $\omega_b := (\mathsf{s}.\sigma_b, r, K_b, \mathsf{op}_b)$. Challenger sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, K_b; \mathsf{op}_b)$. Challenger checks if $\mathsf{s}.\sigma_b$ is a valid signature for message $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

   (c) Otherwise, challenger runs NIZK prover to generate signature $\sigma_b \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_b, \omega_b)$, for $b \in \{0,1\}$. Next, challenger chooses $\hat{b}$. It outputs $(m_{\hat{b}}, \sigma_{\hat{b}}, m_{1-\hat{b}}, \sigma_{1-\hat{b}})$.

5. Admissible adversary outputs bit b' and wins if $b' = \hat{b}$.

Hybrid_1 Instead of honestly generating NIZK proofs for step 3 and 4, challenger outputs simulated proofs.

1. Challenger starts by generating $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathcal{S}(1^\lambda)$.

3.(c) Otherwise, challenger runs NIZK simulator to generate signature σ as a simulated proof for instance x, such that $\sigma = \mathsf{Sim}(\mathsf{nizk.crs}, x)$.

4.(c) Otherwise, challenger runs NIZK simulator to generate signature $\sigma_b \leftarrow\$ \mathsf{Sim}(\mathsf{nizk.crs}, x_b)$, for $b \in \{0,1\}$. Next, challenger chooses $\hat{b}$. It outputs $(m_{\hat{b}}, \sigma_{\hat{b}}, m_{1-\hat{b}}, \sigma_{1-\hat{b}})$.

Hybrid_2 For $b \in \{0,1\}$, instead of setting $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, K_b; \mathsf{op}_b)$, challenger sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$.

2. For $b \in \{0,1\}$, challenger samples PRF key $K_b \leftarrow\$ \{0,1\}^\lambda$ and commitment randomness $\mathsf{op}_b \leftarrow\$ \{0,1\}^\lambda$. Next, it computes $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$. Challenger sets sk_{R_b} as (K_b, op_b), and pk_{R_b} as c_b. It outputs pk_{R_0} and pk_{R_1}.

3.(b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. Challenger computes message as $m = F_{K_{b^{(i)}}}(r^{(i)})$. It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_{b^{(i)}})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

4.(b) For $b \in \{0,1\}$. Challenger sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. Challenger computes message as $m_b = F_{K_b}(r_b)$. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$. Challenger sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$. Challenger checks if $\mathsf{s}.\sigma_b$ is a valid signature for message $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

Hybrid_3 On step 3, 4, instead of setting messages as an output of PRF F, challenger samples messages uniformly at random.

3.(b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. Challenger samples message as m uniformly at random. It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_{b^{(i)}})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

4.(b) For $b \in \{0,1\}$. Challenger sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. Challenger samples message as m_b uniformly at random. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$. Challenger sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$. Challenger checks if $\mathsf{s}.\sigma_b$ is a valid signature for message $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

Let $\mathsf{Adv}^j_{\mathcal{A}}$ denote the security advantage of an adversary $\mathcal{A}$ in Hybrid_j, then the following must hold:

Lemma D.5. Assuming that our NIZK scheme NIZK satisfies zero-knowledge property, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^0_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. Suppose there exists some PPT adversary $\mathcal{A}$ such that $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^0_{\mathcal{A}}| = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ that breaks the security of NIZK with non-negligible advantage.

NIZK challenger starts by sampling $b^* \leftarrow\$ \{0,1\}$. If $b^* = 0$, it samples $\mathsf{nizk.crs} \leftarrow\$ \mathsf{NIZK.Setup}(1^\lambda)$. Otherwise, it samples $\mathsf{nizk.crs} \leftarrow\$ \mathcal{S}(1^\lambda)$. Challenger sends nizk.crs to $\mathcal{B}$. $\mathcal{B}$ then samples $\mathsf{com.crs} \leftarrow\$ \mathsf{COM.Setup}(1^\lambda)$ and outputs $\mathsf{pp} := (\mathsf{nizk.crs}, \mathsf{com.crs})$. Next, for $b \in \{0,1\}$, $\mathcal{B}$ samples PRF key $K_b \leftarrow\$$

$\{0,1\}^\lambda$ and commitment randomness $\mathsf{op}_b \leftarrow\$ \{0,1\}^\lambda$. $\mathcal{B}$ computes $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, K_b; \mathsf{op}_b)$. Then, $\mathcal{B}$ sets sk_{R_b} as (K_b, op_b) and pk_{R_b} as c_b. $\mathcal{B}$ sends pk_{R_0} and pk_{R_1} to $\mathcal{A}$.

$\mathcal{A}$ then makes a series of k queries: In the i-th query, $\mathcal{A}$ outputs $b^{(i)}$, $\mathsf{vk}^{(i)}$, $\mathsf{psig}^{(i)}$, and $\mathsf{nonce}^{(i)}$. $\mathcal{B}$ sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. $\mathcal{B}$ computes message as $m = F_{K_{b^{(i)}}}(r^{(i)})$. It sets $x := (\mathsf{vk}, \mathsf{com.crs}, m)$, $\omega := (\mathsf{s}.\sigma^{(i)}, r^{(i)}, K_{b^{(i)}}, \mathsf{op}_{b^{(i)}})$. $\mathcal{B}$ also sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, K_{b^{(i)}}; \mathsf{op}_{b^{(i)}})$. $\mathcal{B}$ then checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, $\mathcal{B}$ queries NIZK challenger with (x, ω) and challenger replies with signature σ. It outputs m and σ.

Finally, $\mathcal{A}$ outputs vk, $(\mathsf{psig}_b, \mathsf{nonce}_b)_b$ for $b \in \{0,1\}$. $\mathcal{B}$ sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. For $b \in \{0,1\}$, $\mathcal{B}$ computes message as $m_b = F_{K_b}(r_b)$. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$ and $\omega_b := (\mathsf{s}.\sigma, r, K_b, \mathsf{op}_b)$. $\mathcal{B}$ also sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, K_b; \mathsf{op}_b)$. $\mathcal{B}$ checks if $\mathsf{s}.\sigma_b$ is a valid signature for $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, $\mathcal{B}$ chooses $\hat{b}$. Next, it queries NIZK challenger with (x_0, ω_0) and (x_1, ω_1). Challenger sends back proofs σ_0 and σ_1. $\mathcal{B}$ outputs $(m_{\hat{b}}, \sigma_{\hat{b}}, m_{1-\hat{b}}, \sigma_{1-\hat{b}})$. $\mathcal{A}$ then outputs bit b'. If $b' = \hat{b}$, $\mathcal{B}$ outputs 0, indicating that it receives an honestly generated proof from NIZK challenger. Otherwise, $\mathcal{B}$ outputs 1.

Note that if challenger uses a simulator, $\mathcal{B}$ then perfectly simulates Hybrid_1 for $\mathcal{A}$. Otherwise, it simulates Hybrid_0. Thus, if $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^0_{\mathcal{A}}|$ is non-negligible, $\mathcal{B}$ would have non-negligible advantage against the NIZK security challenger. □

Lemma D.6. Assume that the commitment scheme COM is secure, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. We set the following intermediate step for the proof. In the intermediate step, everything remains the same as Hybrid_1 and Hybrid_2, except for the following:

2. For $b \in \{0,1\}$, challenger samples PRF key $K_b \leftarrow\$ \{0,1\}^\lambda$ and commitment randomness $\mathsf{op} \leftarrow\$ \{0,1\}^\lambda$. Next, it computes $c_0 = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_0)$, and $c_1 = \mathsf{COM.Com}(\mathsf{com.crs}, K_1; \mathsf{op}_1)$. Challenger sets sk_{R_b} as (K_b, op_b), and pk_{R_b} as c_b. It outputs pk_{R_0} and pk_{R_1}.

3.(b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. Challenger computes message as $m = F_{K_{b^{(i)}}}(r^{(i)})$. It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$ and $\omega := (\mathsf{s}.\sigma^{(i)}, r^{(i)}, K_{b^{(i)}}, \mathsf{op}_{b^{(i)}})$. If $b^{(i)} = 0$, challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_{b^{(i)}})$. Otherwise, challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, K_{b^{(i)}}; \mathsf{op}_{b^{(i)}})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{1.5}_{\mathcal{A}}$. Suppose there exists a PPT adversary $\mathcal{A}$ such that $|\mathsf{Adv}^{1.5}_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ which breaks the hiding property of commitment scheme.

Reduction algorithm $\mathcal{B}$ first outputs 1^n, where n is the size of PRF key K. The commitment challenger starts by setting $\mathsf{com.crs} \leftarrow\$ \mathsf{COM.Setup}(1^\lambda, 1^n)$ and $b^* \leftarrow\$ \{0,1\}$. $\mathcal{B}$ then samples $\mathsf{nizk.crs} \leftarrow\$ \mathcal{S}(1^\lambda)$ and outputs $\mathsf{pp} := (\mathsf{nizk.crs}, \mathsf{com.crs})$. Next, for $b \in \{0,1\}$, $\mathcal{B}$ samples PRF key $K_b \leftarrow\$ \{0,1\}^\lambda$. $\mathcal{B}$ then queries the challenger with K_b and $\mathbf{0}$ and the challenger replies commitment c_0. $\mathcal{B}$ keeps a recording of the value of c_0. Next, $\mathcal{B}$ generates $c_1 = \mathsf{COM.Com}(\mathsf{com.crs}, K_1; \mathsf{op}_1)$. $\mathcal{B}$ sends $(\mathsf{pk}_{R_0} := c_0, \mathsf{pk}_{R_1} := c_1)$ to $\mathcal{A}$.

$\mathcal{A}$ then makes a series of k queries. In the i-th query, $\mathcal{A}$ sends $b^{(i)}$, $\mathsf{vk}^{(i)}$, $\mathsf{psig}^{(i)}$, and $\mathsf{nonce}^{(i)}$. $\mathcal{B}$ sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. $\mathcal{B}$ computes message as $m = F_{K_{b^{(i)}}}(r^{(i)})$. It sets $x :=$

$(\mathsf{vk}, \mathsf{com.crs}, m)$. When $b^{(i)} = 0$, $\mathcal{B}$ checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c_0 \| r^{(i)}$ and outputs $\perp$ and aborts if the check fails. Otherwise when $b^{(i)} = 1$, $\mathcal{B}$ randomly picks op_1 and sets $c_1 = \mathsf{COM.Com}(\mathsf{com.crs}, K_1; \mathsf{op}_1)$. It checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c_1 \| r^{(i)}$, and outputs $\perp$ and aborts if fails. Note that in this step, c_0 is stored. If the check does not fail, $\mathcal{B}$ generates $\sigma \leftarrow\!\$\, \mathsf{Sim}(\mathsf{nizk.crs}, x)$. It outputs m and σ.

Finally, $\mathcal{A}$ outputs vk, $(\mathsf{psig}_b, \mathsf{nonce}_b)_b$ for $b \in \{0,1\}$. $\mathcal{B}$ sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. For $b \in \{0,1\}$, $\mathcal{B}$ computes message as $m_b = F_{K_b}(r_b)$. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$. $\mathcal{B}$ also sets $c_1 = \mathsf{COM.Com}(\mathsf{com.crs}, K_1; \mathsf{op}_1)$. $\mathcal{B}$ checks if $\mathsf{s}.\sigma_b$ is a valid signature for $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, $\mathcal{B}$ chooses $\hat{b}$. Next, it generates simulated proofs σ_0 and σ_1. $\mathcal{B}$ outputs $(m_{\hat{b}}, \sigma_{\hat{b}}, m_{1-\hat{b}}, \sigma_{1-\hat{b}})$. $\mathcal{A}$ then outputs bit b'. If $b' = \hat{b}$, $\mathcal{B}$ outputs 0, indicating that c_0 is a commitment of K_0. Otherwise, $\mathcal{B}$ outputs 1, indicating that c_1 is a commitment of $\mathbf{0}$. Thus, if $|\mathsf{Adv}^{1.5}_{\mathcal{A}} - \mathsf{Adv}^{1}_{\mathcal{A}}|$ is non-negligible, $\mathcal{B}$ would have non-negligible advantage against the hiding property of the underlying commitment scheme. Following from this proof, we also have $|\mathsf{Adv}^{2}_{\mathcal{A}} - \mathsf{Adv}^{1}_{\mathcal{A}}|$ to be negligible. □

Lemma D.7. Assume that the PRF function F is secure, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^{3}_{\mathcal{A}} - \mathsf{Adv}^{2}_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. We define the following intermediate step for the proof. In the intermediate step, everything remains the same as Hybrid_2 and Hybrid_3, except for the following:

3.(b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. If $b^{(i)} = 0$, challenger samples message as m uniformly at random. Otherwise, it computes $m = F_{K_{b^{(i)}}}(r^{(i)})$ It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_{b^{(i)}})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

4.(b) For $b \in \{0,1\}$. Challenger sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. Challenger samples message as m_0 uniformly at random, and computes $m_1 = F_{K_1}(R_1)$. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$. Challenger sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$. Challenger checks if $\mathsf{s}.\sigma_b$ is a valid signature for message $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

Define adversary $\mathcal{A}$'s advantage on the intermediate step as $\mathsf{Adv}^{2.5}_{\mathcal{A}}$. Suppose there exists a PPT adversary $\mathcal{A}$ such that $|\mathsf{Adv}^{2.5}_{\mathcal{A}} - \mathsf{Adv}^{2}_{\mathcal{A}}| = \epsilon(\lambda)$, we design a reduction algorithm $\mathcal{B}$ which breaks the PRF security.

Challenger starts by sampling a PRF key K, and samples a bit $b^* \leftarrow\!\$\, \{0,1\}$. Reduction algorithm $\mathcal{B}$ first sets $\mathsf{com.crs} \leftarrow\!\$\, \mathsf{COM.Setup}(1^\lambda, 1^n)$ and $\mathsf{nizk.crs} \leftarrow\!\$\, \mathcal{S}(1^\lambda)$. $\mathcal{B}$ outputs $\mathsf{pp} := (\mathsf{nizk.crs}, \mathsf{com.crs})$. Next, for $b \in \{0,1\}$, $\mathcal{B}$ generates $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$ using randomness op_b. $\mathcal{B}$ sends $(\mathsf{pk}_{R_0} := c_0, \mathsf{pk}_{R_1} := c_1)$ to $\mathcal{A}$.

$\mathcal{A}$ then makes a series of k queries. In the i-th query, $\mathcal{A}$ sends $b^{(i)}$, $\mathsf{vk}^{(i)}$, $\mathsf{psig}^{(i)}$, and $\mathsf{nonce}^{(i)}$. $\mathcal{B}$ sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. If $b^{(i)} = 0$, $\mathcal{B}$ sends $r^{(i)}$ to the PRF challenger and challenger returns m. Otherwise, $\mathcal{B}$ computes message as $m = F_{K_1}(r^{(i)})$. It sets $x := (\mathsf{vk}, \mathsf{com.crs}, m)$. Next, $\mathcal{B}$ sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_{b^{(i)}})$. It checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$, and outputs $\perp$ and aborts if fails. Otherwise, $\mathcal{B}$ generates $\sigma \leftarrow\!\$\, \mathsf{Sim}(\mathsf{nizk.crs}, x)$. It outputs m and σ.

Finally, $\mathcal{A}$ outputs vk, $(\mathsf{psig}_b, \mathsf{nonce}_b)_b$ for $b \in \{0,1\}$. $\mathcal{B}$ sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. $\mathcal{B}$ queries the PRF challenger with r_0 and the challenger replies with m_0. $\mathcal{B}$ computes message as $m_1 = F_{K_1}(r_1)$. For $b \in \{0,1\}$, it sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$, $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$. $\mathcal{B}$

checks if $\mathsf{s}.\sigma_b$ is a valid signature for $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, $\mathcal{B}$ outputs $\perp$ and aborts. Otherwise, $\mathcal{B}$ chooses $\hat{b}$. Next, it generates simulated proofs σ_0 and σ_1. $\mathcal{B}$ outputs $(m_{\hat{b}}, \sigma_{\hat{b}}, m_{1-\hat{b}}, \sigma_{1-\hat{b}})$. $\mathcal{A}$ then outputs bit b'. If $b' = \hat{b}$, $\mathcal{B}$ outputs 0, indicating that m_0 is the output of the PRF function. Otherwise, $\mathcal{B}$ outputs 1, indicating that m_0 is randomly sampled. Thus, if $|\mathsf{Adv}^{2.5}_{\mathcal{A}} - \mathsf{Adv}^{2}_{\mathcal{A}}|$ is non-negligible, $\mathcal{B}$ would have non-negligible advantage against the PRF security. Following from this proof, we also have $|\mathsf{Adv}^{3}_{\mathcal{A}} - \mathsf{Adv}^{2}_{\mathcal{A}}|$ to be negligible. □

Note that any adversary has 0 advantage in Hybrid_3. Thus the above lemma completes the proof of our main theorem. □

D.4 Strong nonce blindness

Theorem D.8. Assume that general-purpose NIZK protocol NIZK satisfies zero-knowledge, commitment scheme COM is computationally hiding, and F is a secure PRF, then our NIBS construction D.1 satisfies strong receiver blindness.

Proof. We present the hybrids as the following: Hybrid_0 This corresponds to the original experiment:

1. Challenger starts by generating $\mathsf{nizk.crs} \leftarrow\!\!\$\ \mathsf{NIZK.Setup}(1^\lambda)$ and $\mathsf{com.crs} \leftarrow\!\!\$\ \mathsf{COM.Setup}(1^\lambda)$. It outputs $\mathsf{pp} := (\mathsf{nizk.crs}, \mathsf{com.crs})$.

2. Challenger samples PRF key $K \leftarrow\!\!\$\ \{0,1\}^\lambda$ and commitment randomness $\mathsf{op} \leftarrow\!\!\$\ \{0,1\}^\lambda$. Next, it computes $c = \mathsf{COM.Com}(\mathsf{com.crs}, K; \mathsf{op})$. Challenger sets sk_R as (K, op), and pk_R as c. It outputs pk_R.

3. The adversary is allowed to make a series of k queries for k polynomially bounded in λ. The i-th query is explained as the following ($i \in [k]$):

   (a) In the i-th query, adversary chooses verification key $\mathsf{vk}^{(i)}$, presignature $\mathsf{psig}^{(i)}$ and nonce $\mathsf{nonce}^{(i)}$. Adversary sends them as the query's input to challenger.

   (b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. Challenger computes message as $m = F_K(r^{(i)})$. It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$ and $\omega := (\mathsf{s}.\sigma^{(i)}, r^{(i)}, K, \mathsf{op})$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, K; \mathsf{op})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

   (c) Otherwise, challenger runs NIZK prover to generate signature $\sigma \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x, \omega)$. It outputs m and σ.

4. Adversary then does the following:

   (a) Adversary chooses and outputs vk, $(\mathsf{psig}_b, \mathsf{nonce}_b)_b$ for $b \in \{0,1\}$.

   (b) For $b \in \{0,1\}$. Challenger sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. Challenger computes message as $m_b = F_K(r_b)$. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$ and $\omega_b := (\mathsf{s}.\sigma_b, r, K, \mathsf{op})$. Challenger sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, K; \mathsf{op})$. Challenger checks if $\mathsf{s}.\sigma_b$ is a valid signature for message $c_b \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

   (c) Otherwise, challenger runs NIZK prover to generate signature $\sigma_b \leftarrow\!\!\$\ \mathsf{NIZK.Prove}(\mathsf{nizk.crs}, x_b, \omega_b)$, for $b \in \{0,1\}$. Next, challenger chooses $\hat{b}$. It outputs $(m_{\hat{b}}, \sigma_{\hat{b}}, m_{1-\hat{b}}, \sigma_{1-\hat{b}})$.

5.Admissible adversary outputs bit b' and wins if $b' = \hat{b}$.

Hybrid_1 Instead of honestly generating NIZK proofs for step 3 and 4, challenger outputs simulated proofs.

1. Challenger starts by generating $\mathsf{nizk.crs} \leftarrow\$ \mathcal{S}(1^\lambda)$.

3.(c) Otherwise, challenger runs NIZK simulator to generate signature σ as a simulated proof for instance x, such that $\sigma = \mathsf{Sim}(\mathsf{nizk.crs}, x)$.

4.(c) Otherwise, challenger runs NIZK simulator to generate signature $\sigma_b \leftarrow\$ \mathsf{Sim}(\mathsf{nizk.crs}, x_b)$, for $b \in \{0,1\}$. Next, challenger chooses $\hat{b}$. It outputs $(m_{\hat{b}}, \sigma_{\hat{b}}, m_{1-\hat{b}}, \sigma_{1-\hat{b}})$.

Hybrid_2 For $b \in \{0,1\}$, instead of setting $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, K_b; \mathsf{op}_b)$, challenger sets $c_b = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op}_b)$.

2. For $b \in \{0,1\}$, challenger samples PRF key $K \leftarrow\$ \{0,1\}^\lambda$ and commitment randomness $\mathsf{op} \leftarrow\$ \{0,1\}^\lambda$. Next, it computes $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op})$. Challenger sets sk_R as (K, op), and pk_R as c. It outputs pk_R.

3.(b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. Challenger computes message as $m = F_K(r^{(i)})$. It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

4.(b) For $b \in \{0,1\}$. Challenger sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. Challenger computes message as $m_b = F_K(r_b)$. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op})$. Challenger checks if $\mathsf{s}.\sigma_b$ is a valid signature for message $c \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

Hybrid_3 On step 3, 4, instead of setting messages as an output of PRF F, challenger samples messages uniformly at random.

3.(b) Challenger sets $\mathsf{s}.\sigma^{(i)} := \mathsf{psig}^{(i)}$ and $r^{(i)} := \mathsf{nonce}^{(i)}$. Challenger samples message as m uniformly at random. It sets $x := (\mathsf{vk}^{(i)}, \mathsf{com.crs}, m)$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op})$. Next, challenger checks if $\mathsf{s}.\sigma^{(i)}$ is a valid signature for message $c \| r^{(i)}$. If the check fails, challenger outputs $\perp$ and aborts.

4.(b) For $b \in \{0,1\}$. Challenger sets $\mathsf{s}.\sigma_b := \mathsf{psig}_b$ and $r_b := \mathsf{nonce}_b$. Challenger samples message as m_b uniformly at random. It sets $x_b := (\mathsf{vk}, \mathsf{com.crs}, m_b)$. Challenger sets $c = \mathsf{COM.Com}(\mathsf{com.crs}, \mathbf{0}; \mathsf{op})$. Challenger checks if $\mathsf{s}.\sigma_b$ is a valid signature for message $c \| r_b$, for $b \in \{0,1\}$. If any of the checks fails, challenger outputs $\perp$ and aborts.

Lemma D.9. Assuming that our NIZK scheme NIZK satisfies zero-knowledge property, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^1_{\mathcal{A}} - \mathsf{Adv}^0_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by extending the proof of Lemma D.5. □

Lemma D.10. Assume that the commitment scheme COM is secure, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^2_{\mathcal{A}} - \mathsf{Adv}^1_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by extending the proof of Lemma D.6. □

Lemma D.11. Assume that the PRF function F is secure, then for all PPT adversaries $\mathcal{A}$, $|\mathsf{Adv}^3_{\mathcal{A}} - \mathsf{Adv}^2_{\mathcal{A}}| \leq \mathsf{negl}(\lambda)$.

Proof. (omitted) follows by extending the proof of Lemma D.7. □

Since adversary could only have 0 advantage in Hybrid_3. Thus the above lemma completes the proof of our main theorem. □