Round Optimal Blind Signatures: Short Signatures with Post-Quantum Blindness

Shweta Agrawal[1], Jung Hee Cheon[2], Hyeongmin Choe[2], Damien Stehlé[3], and Anshu Yadav[1]

[1] IIT Madras
shweta@cse.iitm.ac.in, anshu.yadav06@gmail.com
[2] Seoul National University
{jhcheon, sixtail528}@snu.ac.kr
[3] ENS de Lyon and Institut Universitaire de France
damien.stehle@ens-lyon.fr

Abstract. Blind signatures are a fascinating primitive which allow a user to obtain signatures from a signer, while hiding the message. Tremendously useful, these have been studied extensively for decades. Yet, to the best of our knowledge, all concretely practical blind signatures rely on non-standard assumptions and/or achieve sub-optimal round complexity.

In this work, we provide an efficient, round-optimal (two-round) blind signature scheme from the hardness of the discrete log (DL) problem *and* the learning with errors problem in the (non black-box) random oracle model. Our construction enjoys *post-quantum* blindness and does not rely on idealizations such as the algebraic group model or generic group model. We provide a concrete instantiation of our construction. Specifically, our blind signature size and verification time is the same as base Schnorr signature scheme which is used for a building block, making the signature extremely short and the verification extremely fast.

To the best of our knowledge, ours is the first efficient candidate from standard assumptions which simultaneously achieves (very) short signatures, fast verification time, post-quantum blindness and round optimality.

Keywords: blind signature, round optimal, standard assumptions

1 Introduction

Blind signatures [Cha82] are a classic primitive which enable a user to interact with a signer to obtain a signature on a given message so that the signer is unable to learn anything about the message. In more detail, security requires *blindness* which means that an adversarial signer should not be able to associate any message-signature pair with a particular execution of the signing protocol, and *unforgeability*, which means that an adversarial user should be unable to forge "one more" signature even after receiving multiple message-signature pairs. Blind signatures have numerous applications such as e-cash [Cha82], e-voting [IKSA03], anonymous credentials [BCC+09, Bra93, CL01] and more recently, cryptocurrencies [YL19]. They have been researched intensely (please see [PS97, PS00, Sch89, Oka92, Rüc10, ABB20b, ABB20a, BECE+20, LSK+19, PHVBS19, HKLN20] and references therein), with dozens of constructions from different mathematical assumptions, optimizing different parameters ranging from rounds to signature size to communication complexity, having security proofs in different models. Despite so much attention, the state of the art in blind signatures is not fully satisfactory.

1.1 Prior Work

At a high level, there are two broad directions that have emerged, typically bifurcating theory and practice: (i) constructions based on standard assumptions which serve as feasibility results – for instance, the construction by Garg *et al.* [GRS+11] uses (many layers of) complexity leveraging, the construction of Katsumata *et al.* uses

complexity leveraging based on quantum easiness and heavy technical machinery [KNYY21], the construction of Fischlin [Fis06] uses general purpose zero-knowledge proofs, the construction of Hauck *et al.* [HKLN20] suffers exponential loss which severely limits the number of signing queries (ii) practical constructions which rely on non-standard assumptions such as the "one-more" family of assumptions [BNPS03, AKSY21, TZ22], or other non-standard assumptions [FHS15, FHKS16, Bol03] including reliance on the generic group model (GGM) [OA03] or the algebraic group model (AGM) [KLX20].

Aside from the underlying assumptions, another parameter of interest is the number of rounds. While round-optimality is a desirable goal in itself, it is additionally important in the context of blind signatures since it automatically implies security in the concurrent setting, i.e. where protocol executions may run in parallel. Concurrency in the context of blind signatures is delicate and can result in huge losses in the security proof which impacts efficiency [HKKL07, HKLN20, KLR21], and more tragically, has frequently led to errors in the security proof [Rüc10, ABB20b, ABB20a, BECE+20, LSK+19, PHVBS19]. There are very few constructions of round-optimal blind signatures in the literature [GRS+11, Fis06, GG14, KNYY21, AKSY21, LNP22]. Additionally, from standard assumptions, known constructions serve primarily as theoretical feasibility results. Below we summarize the state of the art for practically motivated blind signatures.

Group Based. Schnorr based blind signatures [CP92, PS00, FPS20, KLX20] have been a favourite candidate for practical instantiations since they can rely on standard libraries in implementations, produce very short signatures, do not involve expensive pairing computations and can outperform RSA [TZ22]. Blind Schnorr [CP92], Okamoto-Schnorr [Oka92, PS00], and other other generic constructions based on identification schemes [HKL19] rely on the hardness of the ROS problem, for which a polynomial-time attack was recently found [BLL+21]. The subsequent Clause Blind Schnorr signature [FPS20] relies on the mROS problem, which is not secure against sub-exponential adversaries. Another candidate by Abe [Abe01] is concurrently secure, does not rely on ROS or mROS, and admits proofs in the algebraic group model [KLX20] and the generic group model [OA03].

Very recently, Tessaro and Zhu [TZ22] improved the above line of work by removing the necessity for the problematic ROS/mROS assumption, yielding practical three-move blind signature schemes which are concurrently secure. Their constructions are provably secure either in the generic group model (GGM) or in the algebraic group model (AGM) under the discrete logarithm (DL) or the one-more discrete logarithm (OMDL) assumption, in addition to the ROM. Moreover, their constructions achieve perfect blindness as against the previous best candidate by Abe, which only achieved computational blindness under DDH.

While very attractive in practice, we note that the AGM and GGM are much stronger idealizations than the ROM since they do not allow the adversary to access even the bit representation of the objects it receives as part of the protocol. The relative strength of these idealized models was recently studied by Zhandry and Zhang [ZZ21], where they claim that "the ROM is a strictly milder heuristic than the GGM". Note that relying on AGM or GGM is often *stronger* than a non-standard assumption – for instance, there are several non-standard assumptions, (e.g. ℓ Expanded BDHE [Wat12]) that can be proven secure in the GGM. Additionally, none of these constructions is round-optimal.

From Pairings. BLS signatures [BLS01, Bol03] can be used to obtain very simple pairing-based blind signatures in the ROM but this uses a non-standard assumption, namely the "chosen target CDH" assumption. In the standard model, Garg and Gupta [GG14] provided the first round-optimal candidate with reasonable concrete efficiency from the sub-exponential hardness of the DLIN assumption and a non-standard variant of the discrete log assumption.

Post-Quantum Security. Lattice-based blind signatures have been constructed by [Rüc10, ABB20b, ABB20a, BECE+20, LSK+19, PHVBS19] but it is argued by [HKLN20] that all these constructions have flaws in their security proofs. Other constructions from post-quantum assumptions are from codes [BGSS17] and systems of algebraic equations [PSM17], which are not as well understood. The recent construction by Hauck *et al.* [HKLN20] is primarily a theoretical result – using their provided parameters, the constructed blind signature has size $\approx$ 7.73MB, for security against adversaries limited to getting 7 signatures. The very recent work of Lyubashevsky *et al.* [LNP22] constructs a practical lattice based blind signature from the learning with errors assumption, but their signing protocol depends linearly on the maximum number signatures

queried by the adversary. This makes their construction fall short of satisfying the standard definition of blind signatures. Moreover, note that in the NIST competition, the upper bound for the number of signature queries is 2^{64}. While this is a very conservative estimate, in practice, the number of signing queries is considered a large enough parameter that such a dependence on the run time is a significant drawback for general purpose applications. Another very recent work by Agrawal *et al.* [AKSY21] provides two constructions: an overall efficient lattice based, round optimal blind signature based on a non-standard assumption, and a lattice adaptation of Fischlin's blind signature [Fis06], which uses general purpose zero knowledge proofs, making it inefficient in practice.

1.2 Our Results

In this work we ask: *can we have an efficient, round-optimal blind signature with very short signature size, efficient verification and post-quantum security without relying on the AGM/GGM?*

To the best of our knowledge, no scheme satisfying all the above desiderata is known in the literature. In the present work, we provide a new protocol that achieves round optimality, very short signatures, efficient verification and *post-quantum* blindness, the user is protected against a quantum malicious signer in the blindness game. Our construction relies on the security of Schnorr signatures when the random oracle model is instantiated by a particular hash function (such as SHA-3) and fully homomorphic encryption FHE, which in turn can be built from the (circular secure) Learning With Errors assumption. We note that these assumptions are considered fairly standard in practice. Concretely, our signature size is 6λ bits, and achieves extremely fast verification time and user complexity. However, as discussed below, communication cost and signer complexity are currently bottlenecks in our construction when instantiated with state of the art FHE implementations – however, we believe that progress in FHE implementations will substantially reduce these costs. We discuss this in detail in Section 1.4.

Unlike all prior constructions based on discrete log, we do *not* require the algebraic group model (AGM) or the generic group model (GGM) for a proof of security, nor do we incur sub-optimal round complexity. Please see Table 1 for a detailed comparison with practice oriented schemes from standard assumptions.

Work	Rounds	Assumption	PQ-Blindness	ROM/AGM/GGM	Sign Queries
[TZ22]	3	DL	Yes	ROM,AGM,GGM	Unlimited
[GG14]	2	DLIN and (variant of) DL	No	Standard	Unlimited
[LNP22]	2	LWE	Yes	ROM	Limited
This	2	DL and LWE	Yes	non-BB ROM	Unlimited

Table 1 : State of the Art Practical Schemes from Standard Assumptions. Note that [LNP22] achieves a non-standard definition of blind signatures. Non-blackbox ROM means that the hash function modeled as the random oracle is required to be evaluated homomorphically within an FHE scheme.

Post-Quantum Blindness. Note that in practical applications, the signer is typically more powerful than the user and blindness is typically a much longer term requirement than unforgeability. For instance, consider an e-voting protocol. Here, blindness is used to hide the vote while unforgeability is used to protect against a malicious user forging votes. Thus, it is very desirable that secrecy of votes be maintained even after quantum computers are realized whereas unforgeability is no longer relevant once the voting process is completed. Due to this, we believe that post quantum blindness is much more important than post quantum unforgeability.

Non Black Box ROM. Our proof is in the *non black box* random oracle model, where the hash function modeled as the random oracle is required to be evaluated homomorphically within an FHE scheme. Since the random oracle model (ROM) is only meaningful for the real world when instantiated by a concrete hash function, we believe this non black-box usage of the ROM does not qualitatively change the assumption in practice.

Concurrent Work. Independently and concurrently to our work, del Pino and Katsumata [dPK22] provided a round optimal blind signature from standard lattice assumptions. They achieve a signature size of 100KB compared to our signature of approximately 100B. Our construction relies on the usual Schnorr signature scheme, and inherits signature compactness and verification efficiency from it. However, we currently cannot accurately estimate our signer time complexity concretely due to limitations of FHE (as discussed in Section 1.4) while their signer time complexity is efficient. Moreover, they achieve lattice based unforgeability as well as blindness whereas we only achieve the latter – this makes their construction fully (conjectured) post quantum while ours is only (conjectured) post quantum for blindness. However, using discrete log based unforgeability is also what allows our construction to achieve signatures that are shorter by a factor of 1000 – this appears very hard to achieve with lattice techniques. Finally, their construction can also be shown secure in the QROM while ours is secure only in the (non black-box) ROM.

1.3 Our Techniques

The starting point of our work is the round optimal standard model construction of Garg *et al.* [GRS+11]. The primary objective of this work was to provide a construction in the standard model, which led to heavy technical machinery (such as witness indistinguishable ZAPs) as well as complexity leveraging [FS10, Pas11]. For these reasons, this protocol has remained largely in the domain of feasibility for over a decade. However, upon re-examining it via the lens of practicality, we observe that several of the main sources of inefficiency, such as complexity leveraging and witness indistinguishable ZAPs can be removed or mitigated, if we trade the standard model for the random oracle model. Since, as discussed above, most practical constructions rely anyway on idealized models (often multiple of these) as well as non-standard assumptions, using the random oracle is a small price to pay if it can lead to a practical construction. While these initial simplifications do not immediately yield a practical protocol, they do provide a promising starting point.

With the goal of post-quantum security in mind, and given the recent exciting advances in practical fully homomorphic encryption (FHE) (e.g. [HS20, HS21, HS20, CGGI16, CGGI17]) and for other technical reasons (outlined below) we use FHE as a building block and begin with the following simple protocol. The signing and verification key of the blind signature is that of any UF-CMA signature scheme, together with the public key of a standard PKE scheme (required for technical reasons). Now, to blind the message μ, the user $\mathcal{U}$ samples the key pair for an FHE scheme, encrypts μ under FHE to obtain ct_μ and the FHE secret key under PKE to obtain $\mathsf{ct}_{\mathsf{sk}}$. It sends the FHE public key, the ciphertexts ct_μ and $\mathsf{ct}_{\mathsf{sk}}$ and a NIZK proof that the FHE key pair and ciphertext $\mathsf{ct}_{\mathsf{sk}}$ were generated honestly to the signer $\mathcal{S}$. Next, the signer $\mathcal{S}$ homomorphically evaluates the signing circuit on the encrypted message, using its signing key and some fresh randomness and obtains the ciphertext of the signature, which it sends to $\mathcal{U}$. Upon receiving the encrypted signature, $\mathcal{U}$ decrypts and verifies it. Intuitivey, blindness is achieved because of the semantic security of FHE while one-more unforgeability is inherited from the unforgeability of the underlying signature and the soundness of the proofs. A subtlety is that the challenger must extract all (say Q) messages being signed in the unforgeability proof without rewinding Q times to avoid a large security loss – this is achieved by careful use of the PKE. Please see Section 3 for details.

Dealing with a Malicious Signer. A significant issue that is brushed under the carpet in the description above is the handling of a malicious signer. Evidently, the signer can deviate from the protocol to gain advantage in the blindness game. Indeed, we can show that there is an actual attack that the malicious signer can mount in order to break blindness – please see Section 3 for details.

A natural approach to protect against malicious signers is to force the signer to provide a NIZK that the signature was generated correctly using the honest signing key. However, since the signer is performing a complicated operation, such a proof system must handle a complex statement which does not currently admit efficient instantiations. To overcome this hurdle, we design two new techniques:

Collaborative Generation of Randomness. To constrain the power of the cheating signer, our idea is to let the user participate in the generation of the randomness used in the signature. In more detail, let us say that the user samples some randomness k_u, the signer samples randomness k_s and the randomness used by

the signing algorithm is $k_u + k_s$. The user includes an FHE encryption of k_u[4] in its message, and the signing algorithm is modified to use $k_u + k_s$ instead of just k_s. The hope is that this may help to limit the effect of bad randomness chosen by the signer. However, it is unclear whether it achieves anything – for instance, what prevents the signer from discarding the user's randomness altogether? If it needs to prove that it used the randomness provided by the user, we are again stuck with an arbitrarily complex proof and may be back to square one!

Using Algebraic Structure of Specific Schemes. To overcome this hurdle, we will leverage carefully chosen algebraic structure of specific signature and encryption schemes. Appropriate algebraic structure of the building blocks interleaved by carefully designed (inexpensive) checks in the protocol will enable us to replace expensive general purpose zero knowledge proofs by efficient proofs for simple algebraic statements. We emphasize that this step leads to many subtle complexities in the proof of blindness, handling which is one of the primary technical contributions of this work. Additionally, by analyzing the structure of specific schemes, we develop explicit algorithms for homomorphic evaluation of concrete circuits. This allows us to obtain detailed cost estimates for the communication and computation incurred by the signer and user using available FHE benchmarks from the literature.

Our Protocol. We consider the classic Schnorr signature scheme, based on the standard discrete log assumption. Let us recap the scheme: consider a group $G = \langle g \rangle$ of prime order q and a hash function $H : \{0,1\}^* \to \mathbb{Z}_q^*$ modelled as a random oracle. The secret key is sampled as $\mathsf{sk} \leftarrow \mathbb{Z}_q^*$, and the public key is set as g^{sk}. To sign a message μ, the signer samples a random k, computes $r = g^k$, sets $s = k + \mathsf{sk} \cdot H(\mu, r)$ and outputs (r, s) as the signature. Given the verification key $\mathsf{vk} = g^{\mathsf{sk}}$, one can check if $g^s = r \cdot \mathsf{vk}^{H(\mu,r)}$.

We are ready to provide a simplified version of our blind signature. The signing key and verification key are simply those of the Schnorr blind signature described above. Now, the user samples the secret and public key $(\mathsf{hpk}, \mathsf{hsk})$ for an FHE scheme, and constructs a proof π_{sk} that these keys are well formed. Here we rely on a key observation that the algebraic structure of FHE schemes allows to restrict the statements being proven to linear relations with short unknowns, for which specific lattice-based techniques are known (see [YAZ+19, BLS19, ENS20, LNS21], and references therein). These techniques result in shorter proof sizes as well as efficient running times, allowing us to avoid general zero knowledge proofs as discussed above. Next, the user encrypts its message μ to obtain ct_μ, samples partial randomness $k_u \leftarrow \mathbb{Z}_q^*$, computes $r_u = g^{k_u}$ and encrypts it to obtain ct_{r_u}. Additionally, it generates proofs π_μ and π_{r_u} that these ciphertexts are well-formed – these proofs also entail linear statements with our choice of scheme and can be instantiated efficiently as above. Finally, it sends the tuple $(\mathsf{hpk}, \pi_{\mathsf{sk}}, \mathsf{ct}_{r_u}, \mathsf{ct}_\mu, \pi_{r_u}, \pi_\mu)$ to the signer. We note that the user sends the encryption of r_u, not k_u – this makes the signing process simpler, while maintaining the security in ROM.

The signer verifies the proofs π_{sk}, π_{r_u} and π_μ (efficiently). It then samples partial randomness $k_s \leftarrow \mathbb{Z}_q$ and computes $r_s = g^{k_s}$. It also generates a NIZK proof π_{k_s} that r_s is well-formed – again, this proof is for a simple algebraic statement and admits an efficient instantiation. It then homomorphically evaluates the signing circuit on $\mathsf{ct}_{r_u}, \mathsf{ct}_\mu$ using its signing key and its own partial randomness k_s to obtain ct_s. It sends $\mathsf{ct}_s, r_s, \pi_{k_s}$ to the user. The careful reader may have noticed that the step of homomorphic evaluation appears expensive and in particular, incurs evaluating a hash function – we will discuss below how this can be done with reasonable efficiency, though it remains the most expensive part of the protocol. To proceed, the user verifies π_{k_s}, computes $r = r_u \cdot r_s$ and $s = k_u + \mathsf{FHE.Dec}(\mathsf{hsk}, \mathsf{ct}_s)$. It then runs the Schnorr verification algorithm on (r, s).

Challenges in Security. We refer the reader to Section 4 for the detailed construction and proofs. Here, we only touch upon a few subtleties. Let us first see why the scheme is secure against a malicious signer who may not follow the protocol. Note that in the blindness game, the challenger can extract the witnesses k_s from the proof π_{k_s} and check whether this matches the received r_s. Since the challenger knows k_u, k_s and sk[5]

[4] In fact, for efficiency reasons, we will encrypt some function of k_u.

[5] If the signing key sk is also maliciously generated, the challenger must also extract the signing key from a proof. For simplicity, here we assume that sk is honestly generated by the challenger.

it can itself compute the Schnorr signature. Due to uniqueness of the discrete log and correctness of FHE, one can show that the computed signature has to be the same as the one generated by evaluating the signing circuit homomorphically followed by decryption, if both passed verification. This lets the challenger produce the signature without using the FHE decryption key, which in turn allows it to replace the FHE ciphertexts seen by the signer by dummy ciphertexts. At this point the challenge bit is information theoretically hidden to the adversary. A delicate issue that arises is that there are an exponential number of circuits that are equivalent to the correct signing circuit (for instance, those with dummy gates) and evaluating these too will result in correct decryption and signature verification. This "gap" between correctness of FHE and honest circuit computation creates subtle challenges in the proof, which require care to handle. Please see Section 4 for details.

Conjectured Post-Quantum Blindness. Our scheme achieves post-quantum blindness, i.e. blindness holds even against a quantum malicious signer. To see this, note that in the blindness game, the honest user is protected using the semantic security of FHE and the zero knowledge property of lattice based proofs, which are all conjectured post-quantum secure.

1.4 Challenges in Efficient Signing using FHE.

The efficiency of our protocol depends on the efficiency of the homomorphic evaluation of the Schnorr signing algorithm. While we have adapted our protocol to make homomorphic evaluation efficient, there are still some challenges that remain. Here, we provide a summary of our approach and outline some challenges for future work on FHE machinery. Progress by the community in addressing these problems in FHE efficiency will lead to further improvements in the concrete efficiency of our protocol.

Communication Cost. To use FHE as a building block of the blind signature, the size of the public key can be a problem since it is large compared to other primitives. The reason for the large size is mainly because the signer needs many rotation keys for the hash function evaluation and the bootstrapping. However, the communication cost can be decreased by trading off with signing time, in particular by constructing the rotation keys at the signer's end as suggested by Lee *et al.* [LLKN22]. Then, it remains to reduce the signing complexity incurred by homomorphic evaluation. We discuss this next.

Large Fields. Evaluating the signing protocol of Schnorr signatures on a finite field requires evaluation of the modulo operation with 1000-3000 bit primes over FHE which are too large for FHE implementations. An alternative is to use the Schnorr signature based on elliptic curve cryptography, which makes the modulus sizes smaller, to about 256 bits. However, this is not compatible with the currently used (meaningful) parameters for FHE [HS21, LLK+22, CLPX18, CLOT21]. Even in the EC Schnorr case, we need to instantiate FHE with parameters that are not efficient, especially with regards to bootstrapping [HS21]. The main reason for the inefficiency is that the current FHE machinery focuses only on much smaller moduli, or moduli of some specific forms. Hence an interesting open question posed by our work is the development of FHE machinery for large fields. We believe this is a natural direction for research which will have more applications.

Expensive Group Operations. Evaluating group operations on a large multiplicative group of prime order q over 256 bits can be another challenge. In particular, exponentiation is one of the most challenging operations since it requires a sequential multiplication with a huge-depth circuit, which becomes a dominant factor in running time. To bypass this operation, we modify the blind signature scheme that enjoys the programmability of ROM more actively. However, progress in efficient FHE evaluation of such operations would significantly speed up the concrete running time of the protocol.

Conversion of the Plaintext Space. In our protocol, the hash function takes as input a message in $\mathcal{M}$ and a group element in G where the group G is defined over a finite field $\mathbb{F}_p$ (or over an elliptic curve over a finite field $\mathbb{F}_p$), and outputs a finite field element in $\mathbb{Z}_q$. Therefore, there should be an evaluation of a mapping

between the spaces with different modulus p and q, which is very inefficient using current state of art in FHE. For our protocol, FHE plaintext space conversion can be a core technique, since it makes the signing process very efficient – just two constant multiplications and one addition, except for hashing and converting. Theoretically, this is possible through recryption. However, an efficient evaluation of a circuit $[[\star]_Q]_p$ over FHE with plaintext modulus q and ciphertext modulus Q', in the recryption methodology, appears challenging given current techniques. We leave this as an important direction for future work.

2 Preliminaries

In this section, we provide some preliminaries used in our work.

Notation. We write vectors with bold lower-case letters and matrices with bold upper-case letters. For any vector $\mathbf{v}$, we denote its ith element by $\mathbf{v}[i]$ or $\mathbf{v}_i$. Similarly, for any matrix $\mathbf{M}$, $\mathbf{M}[i][j]$ or $\mathbf{M}_{ij}$ represents the element in the jth column of ith row. Let S be any set, then $|S|$ represents the cardinality of S, while in case of any $x \in \mathbb{R}$, $|x|$ represents absolute value of x. For any $n \in \mathbb{N}$, we let the set $\{1, 2, \ldots, n\}$ be denoted by $[n]$. For a distribution D over a countable set $\mathcal{X}$, we let $H_\infty(D) = -\max_{x \in \mathcal{X}} \log_2 D(x)$ denote the min-entropy of D. The statistical distance between two distributions D_0 and D_1 over $\mathcal{X}$ is defined as $\frac{1}{2}\sum_{x \in \mathcal{X}} |D_0(x) - D_1(x)|$.

We use standard definitions for pseudo-random functions (PRF), public-key encryption (PKE), signatures and commitments. We place ourselves in a setup that allows the attackers to run in time $2^{o(\lambda)}$ and succeed with probability $2^{-o(\lambda)}$, but that forbids them to make more than $\text{poly}(\lambda)$ interactions with honest users. Compared to the setup of polynomially bounded attackers, this allows to better reflect practice and to better differentiate between operations that the adversary can do on its own and are only limited by the adversary runtime (such as hash queries) and operations that require interaction with a honest user and are much more limited (such as signature queries). We note that if we limit ourselves to polynomially bounded adversaries, then all our reductions of our security proofs involve polynomial-time reductions and would not require subexponential hardness assumptions.

2.1 Blind Signatures

To begin, we introduce some notation for interactive executions between algorithms $\mathcal{X}$ and $\mathcal{Y}$. By $(a, b) \leftarrow \langle \mathcal{X}(x), \mathcal{Y}(y) \rangle$, we denote the joint execution of $\mathcal{X}$ and $\mathcal{Y}$ where $\mathcal{X}$ has private input x, $\mathcal{Y}$ has private input y and $\mathcal{X}$ receives private output a while $\mathcal{Y}$ receives private output b.

Definition 1 (Blind Signature). *A blind signature scheme BS consists of PPT algorithms* Gen, Vrfy *along with interactive PPT algorithms* $\mathcal{S}$, $\mathcal{U}$ *such that for any* λ*:*

- $\mathsf{Gen}(1^\lambda)$ *generates a key pair* $(\mathsf{BS.sk}, \mathsf{BS.vk})$.
- *The joint execution of* $\mathcal{S}(\mathsf{BS.sk})$ *and* $\mathcal{U}(\mathsf{BS.vk}, \mu)$*, where* $\mu \in \{0,1\}^*$*, generates an output* σ *for the user and no output for the signer; this is denoted as* $(\perp, \sigma) \leftarrow \langle \mathcal{S}(\mathsf{BS.sk}), \mathcal{U}(\mathsf{BS.vk}, \mu) \rangle$.
- *Algorithm* $\mathsf{Vrfy}(\mathsf{BS.vk}, \mu, \sigma)$ *outputs a bit* b.

The scheme must satisfy completeness: for any $(\mathsf{BS.sk}, \mathsf{BS.vk}) \leftarrow \mathsf{Gen}(1^\lambda)$, $\mu \in \{0,1\}^*$ *and* σ *output by* $\mathcal{U}$ *in the joint execution of* $\mathcal{S}(\mathsf{BS.sk})$ *and* $\mathcal{U}(\mathsf{BS.vk}, \mu)$*, it holds that* $\mathsf{Vrfy}(\mathsf{BS.vk}, \mu, \sigma) = 1$ *with probability* $1 - \lambda^{-\omega(1)}$.

Blind signatures must satisfy the following two security properties [JLO97]:

Definition 2 (One More Unforgeability). *The blind signature* $BS = (\mathsf{Gen}, \mathcal{S}, \mathcal{U}, \mathsf{Vrfy})$ *is one more unforgeable if for any polynomial* Q*, and any* $\mathcal{U}^*$ *with run-time* $2^{o(\lambda)}$*, the success probability of* $\mathcal{U}^*$ *in the following game is* $2^{-\Omega(\lambda)}$*:*

1. $\mathsf{Gen}(1^\lambda)$ *outputs* $(\mathsf{BS.sk}, \mathsf{BS.vk})$*, and* $\mathcal{U}^*$ *is given* $\mathsf{BS.vk}$.
2. *Algorithm* $\mathcal{U}^*$ *interacts concurrently with* Q *instances* $\mathcal{S}^1_{\mathsf{BS.sk}}, \ldots, \mathcal{S}^Q_{\mathsf{BS.sk}}$.

3. *Algorithm* $\mathcal{U}^*$ *outputs* $(\mu_1, \sigma_1, \ldots, \mu_{Q+1}, \sigma_{Q+1})$.

Algorithm $\mathcal{U}^*$ *succeeds if the* μ_i*'s are distinct and* $\mathsf{Vrfy}(\mathsf{BS.vk}, \mu_i, \sigma_i) = 1$ *for all* $i \in [Q+1]$.

Definition 3 (Honest Signer Blindness). *The blind signature* $\mathsf{BS} = (\mathsf{Gen}, \mathcal{S}, \mathcal{U}, \mathsf{Vrfy})$ *satisfies honest signer blindness if for any algorithm* $\mathcal{S}^*$ *with run-time* $2^{o(\lambda)}$*, the advantage of* $\mathcal{S}^*$ *in the following game is* $2^{-\Omega(\lambda)}$*:*

1. $\mathsf{Gen}(1^\lambda)$ *outputs* $(\mathsf{BS.sk}, \mathsf{BS.vk})$ *and gives it to* $\mathcal{S}^*$*; algorithm* S^* *outputs two messages* μ_0, μ_1 *of its choice.*
2. *A random bit* b *is chosen and* $\mathcal{S}^*$ *interacts concurrently with* $\mathcal{U}_0 := \mathcal{U}(\mathsf{BS.vk}, \mu_b)$ *and* $\mathcal{U}_1 := \mathcal{U}(\mathsf{BS.vk}, \mu_{\bar{b}})$ *possibly maliciously; when* $\mathcal{U}_0$ *and* $\mathcal{U}_1$ *have completed their executions, the values* $\sigma_b, \sigma_{\bar{b}}$ *are defined as follows:*
   - *If either* $\mathcal{U}_0$ *or* $\mathcal{U}_1$ *aborts, then* $(\sigma_b, \sigma_{\bar{b}}) := (\perp, \perp)$.
   - *Otherwise, let* σ_b *(resp.* $\sigma_{\bar{b}}$*) be the output of* $\mathcal{U}_0$ *(resp.* $\mathcal{U}_1$*).*

   Algorithm $\mathcal{S}^*$ *is given* (σ_0, σ_1).
3. *Algorithm* $\mathcal{S}^*$ *outputs a bit* b'.

Algorithm $\mathcal{S}^*$ *succeeds if* $b' = b$*. If* succ *denotes the latter event, then the advantage of* $\mathcal{S}^*$ *is defined as* $|\Pr[\mathsf{succ}] - 1/2|$.

Full-fledged blindness lets the adversary $\mathcal{S}^*$ sample its own pair $(\mathsf{BS.sk}, \mathsf{BS.vk})$ at Step 1 (possibly maliciously), and gives $\mathsf{BS.vk}$ to the challenger. We also consider the notion of **very honest signer blindness** for which the signer $\mathcal{S}^*$ follows the protocol honestly.

2.2 Homomorphic Encryption (FHE)

A homomorphic encryption scheme is an encryption scheme that allows computations on encrypted data.

Definition 4 (Homomorphic Encryption). *A homomorphic encryption scheme* FHE *is a tuple of PPT algorithms* $\mathsf{FHE} = (\mathsf{FHE.KeyGen}, \mathsf{FHE.Enc}, \mathsf{FHE.Eval}, \mathsf{FHE.Dec})$ *defined as follows:*

- $\mathsf{FHE.KeyGen}(1^\lambda, 1^d) \to (\mathsf{pk}, \mathsf{sk})$*: On input the security parameter* λ *and a depth bound* d*, the* KeyGen *algorithm outputs a key pair* $(\mathsf{pk}, \mathsf{sk})$*.*
- $\mathsf{FHE.Enc}(\mathsf{pk}, \mu) \to \mathsf{ct}$*: On input a public key* pk *and a message* $\mu \in \{0,1\}$*, the encryption algorithm outputs a ciphertext* ct*.*
- $\mathsf{FHE.Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k) \to \hat{\mathsf{ct}}$*: On input a public key* pk*, a circuit* $\mathcal{C} : \{0,1\}^k \to \{0,1\}$ *of depth at most* d*, and a tuple of ciphertexts* $\mathsf{ct}_1, \ldots, \mathsf{ct}_k$*, the evaluation algorithm outputs an evaluated ciphertext* $\hat{\mathsf{ct}}$*.*
- $\mathsf{FHE.Dec}(\mathsf{pk}, \mathsf{sk}, \hat{\mathsf{ct}}) \to \hat{\mu}$*: On input a public key* pk*, a secret key* sk *and a ciphertext* $\hat{\mathsf{ct}}$*, the decryption algorithm outputs a message* $\hat{\mu} \in \{0, 1, \perp\}$*.*

The definition above can be adapted to handle plaintexts over larger sets than $\{0,1\}$. Note that the evaluation algorithm takes as input a (deterministic) circuit rather than a possibly randomized algorithm. An FHE should satisfy the compactness, correctness and security properties defined below.

Definition 5 (Compactness). *We say that an* FHE *scheme is compact if there exists a polynomial function* $f(\cdot, \cdot)$ *such that for all* λ*, depth bound* d*, circuit* $\mathcal{C} : \{0,1\}^k \to \{0,1\}$ *of depth at most* d*, and* $\mu_i \in \{0,1\}$ *for* $i \in [k]$*, the following holds: for* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda, 1^d)$*,* $\mathsf{ct}_i \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, \mu_i)$ *for* $i \in [k]$*,* $\hat{\mathsf{ct}} \leftarrow \mathsf{FHE.Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)$*, the bit-length of* $\hat{ct}$ *is at most* $f(\lambda, d)$*.*

Definition 6 (Correctness). *We say that an* FHE *scheme is correct if for all* λ*, depth bound* d*, circuit* $\mathcal{C} : \{0,1\}^k \to \{0,1\}$ *of depth at most* d*, and* $\mu_i \in \{0,1\}$ *for* $i \in [k]$*, the following holds: for* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda, 1^d)$*,* $\mathsf{ct}_i \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, \mu_i)$ *for* $i \in [k]$*,* $\hat{\mathsf{ct}} \leftarrow \mathsf{FHE.Eval}(\mathsf{pk}, \mathcal{C}, \mathsf{ct}_1, \ldots, \mathsf{ct}_k)$*, we have*

$$\Pr[\mathsf{FHE.Dec}(\mathsf{pk}, \mathsf{sk}, \hat{\mathsf{ct}}) = \mathcal{C}(\mu_1, \ldots, \mu_k)] = 1 - \lambda^{-\omega(1)}.$$

Definition 7 (Security). *We say that an* FHE *scheme is secure if for all* λ *and depth bound* d*, the following holds: for any adversary* $\mathcal{A}$ *with run-time* $2^{o(\lambda)}$*, the following experiment outputs* 1 *with probability* $2^{-\Omega(\lambda)}$*:*

1. *On input the security parameter* λ *and a depth bound* d*, the challenger runs* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda, 1^d)$ *and* $\mathsf{ct} \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, b)$ *for* $b \leftarrow \{0,1\}$*; it provides* $(\mathsf{pk}, \mathsf{ct})$ *to* $\mathcal{A}$*.*
2. *Adversary* $\mathcal{A}$ *outputs a guess* b'*; the challenger outputs* 1 *if* $b = b'$*.*

In our application, we will also need the FHE scheme to support circuit privacy, as defined below.

Definition 8 (Circuit Privacy). *We say that the homomorphic encryption scheme* FHE *is semi-honest circuit private if for* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda, 1^d)$*, any circuit* $\mathcal{C} : \{0,1\}^k \to \{0,1\}$ *of depth at most* d*,* $\mu_i \in \{0,1\}$ *for* $i \in [k]$*, no* $2^{o(\lambda)}$*-time adversary can distinguish between the following distributions with* $2^{-o(\lambda)}$ *advantage:*

$$\Big(\mathsf{FHE.Eval}(\mathsf{pk}, \mathcal{C}, \{\mathsf{ct}_i\}_{i \le k}), \{\mathsf{ct}_i\}_{i \le k}, \mathsf{pk}, \mathsf{sk}\Big)$$
$$\text{and } \Big(\mathsf{FHE.Eval}(\mathsf{pk}, \mathcal{C}^0, \{\mathsf{ct}'_i\}_{i \le k}), \{\mathsf{ct}_i\}_{i \le k}, \mathsf{pk}, \mathsf{sk}\Big),$$

where $\mathsf{ct}_i \leftarrow \mathsf{FHE.Enc}(\mathsf{pk}, \mu_i)$ *for* $i \in [k]$*,* $\mathsf{ct}'_1 = \mathsf{FHE.Enc}(\mathsf{pk}, \mathcal{C}(\mu_1, \ldots, \mu_k))$*,* $\mathsf{ct}'_i = \mathsf{FHE.Enc}(\mathsf{pk}, 0)$ *for* $1 < i \le k$ *and* $\mathcal{C}^0 : \{0,1\}^k \to \{0,1\}$ *is the trivial circuit of depth* d *that outputs its first input and ignores the others.*

We say that FHE *is maliciously circuit private if the above holds even if the keys* $(\mathsf{pk}, \mathsf{sk})$ *and ciphertexts* ct_i *for* $i \in [k]$ *are not necessarily generated honestly. In this case, the* μ_i*'s are defined as* $\mu_i = \mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{ct}_i)$ *for* $i \in [k]$*, where we assume without loss of generality that decryption always outputs a bit (even when the ciphertext is not well-formed).*

2.3 Non-Interactive Zero Knowledge Arguments

Definition 9 (Non Interactive Zero Knowledge Argument). *A non-interactive zero-knowledge (*NIZK*) argument system* Π *for an NP relation* R *consists of three PPT algorithms* $(\mathsf{Gen}, \mathsf{P}, \mathsf{V})$ *with the following syntax:*

- $\mathsf{Gen}(1^\lambda) \to \mathsf{crs}$: *On input a security parameter* λ*, the* Gen *algorithm outputs a common reference string* crs*; in the random oracle model, this algorithm may be skipped, since the* crs *can be generated by* P *and* V *by querying the random oracle on some fixed value.*
- $\mathsf{P}(\mathsf{crs}, x, w) \to \pi$: *On input the common reference string* crs*, a statement* $x \in \{0,1\}^{\mathrm{poly}(\lambda)}$*, a witness* w *such that* $(x, w) \in R$*, the prover* P *outputs a proof* π*.*
- $\mathsf{V}(\mathsf{crs}, x, \pi) \to \mathsf{accept}/\mathsf{reject}$: *On input a common reference string* crs*, a statement* $x \in \{0,1\}^{\mathrm{poly}(\lambda)}$ *and a proof* π*, the verifier* V *outputs* accept *or* reject*.*

The argument system Π *should satisfy the following properties.*

- ***Completeness:*** *For any* $(x, w) \in R$*, we have*

$$\Pr[\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda), \pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w) : \mathsf{V}(\mathsf{crs}, x, \pi) = 1] \ge 1 - \lambda^{-\omega(1)}.$$

- ***Soundness:*** *For any* $x \in \{0,1\}^{\mathrm{poly}(\lambda)}$ *and any* $2^{o(\lambda)}$ *time prover* P^**, we have*

$$\Pr[\mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda), \pi \leftarrow \mathsf{P}^*(\mathsf{crs}, x) : \mathsf{V}(\mathsf{crs}, x, \pi) = 1] \le 2^{-\Omega(\lambda)}.$$

- ***Honest Verifier Zero Knowledge:*** *There is a PPT simulator* Sim *such that, for all statements* x *for which there exists* w *with* $R(x, w) = 1$*, for any* $2^{o(\lambda)}$ *time adversary* $\mathcal{A}$*, we have:*

$$\Big|\ \Pr\left[1 \leftarrow \mathcal{A}\left((\mathsf{crs}, x, \pi)\ :\ \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda),\ \pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w)\right)\right]$$
$$-\Pr\left[1 \leftarrow \mathcal{A}\left((\mathsf{crs}, x, \pi)\ :\ (\mathsf{crs}, \pi) \leftarrow \mathsf{Sim}(1^\lambda, x)\right)\right]\ \Big| \le 2^{-\Omega(\lambda)}.$$

Definition 10 (Argument of Knowledge). *The argument system* $(\mathsf{Gen}, \mathsf{P}, \mathsf{V})$ *is called an argument of knowledge for the relation* R *if it is complete and knowledge-sound as defined below.*

- ***Knowledge Sound:*** *For any* $2^{o(\lambda)}$ *time prover* P^**, there exists an extractor* $\mathcal{E}$ *with expected run-time polynomial in* λ *and the run-time of* P^**, such that for all PPT adversaries* $\mathcal{A}$

$$\Pr\left[\begin{array}{r} \mathsf{crs} \leftarrow \mathsf{Gen}(1^\lambda), (x,s) \leftarrow \mathcal{A}(\mathsf{crs}), \\ \pi^* \leftarrow \mathsf{P}^*(\mathsf{crs}, x, s), b \leftarrow \mathsf{V}(\mathsf{crs}, x, \pi^*), \\ w \leftarrow \mathcal{E}^{\mathsf{P}^*(\mathsf{crs},x,s)}(\mathsf{crs}, x, \pi^*, b) \end{array} \middle| (x,w) \notin R \wedge b = \mathsf{accept} \right] \leq 2^{-\Omega(\lambda)}.$$

If an argument of knowledge is also non-interactive zero knowledge, it is termed as a non-interactive zero knowledge argument of knowledge, abbreviated as NIZKAoK.

3 Warmup: Very Honest Blind Signature from FHE

In Figure 1, we describe a two-round construction of blind signatures based on FHE. Our construction uses the following building blocks:

- A UF-CMA signature scheme $\mathsf{Sig} = (\mathsf{Sig.KeyGen}, \mathsf{Sig.Sign}, \mathsf{Sig.Verify})$. Let $\mathsf{Sig.Sign}_{\mathsf{Sig.sk},\rho}$ denote the circuit that takes μ as input and returns $\mathsf{Sig.Sign}(\mathsf{Sig.sk}, \mu; \rho)$ (i.e., with the key $\mathsf{Sig.sk}$ and the $\mathsf{Sig.Sign}$ randomness ρ being hardwired).
- An IND-CPA secure public key encryption scheme $\mathsf{PKE} = (\mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ having the property that the public key generated by $\mathsf{PKE.KeyGen}$ is indistinguishable from a uniformly sampled value from its range. We observe that this is true for most lattice based PKE schemes e.g. [GPV08].
- A maliciously circuit private fully homomorphic encryption scheme (see Definition 8) $\mathsf{FHE} = (\mathsf{FHE.KeyGen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec}, \mathsf{FHE.Eval})$ that is capable of homomorphically evaluating circuits with depth up to the depth of $\mathsf{Sig.Sign}_{\mathsf{Sig.sk},\rho}$.
- A NIZK argument system for the statements of the form as given in equation 3.1 (see Figure 1).
- A hash function $H' : \{0,1\}^* \to \mathcal{PK}$ modeled as random oracle. Here, $\mathcal{PK}$ is the sapce of public keys of PKE.

The construction is described in Figure 1.

Completeness: The completeness follows from the completeness of the underlying primitives, NIZK, FHE and Sig. The completeness of the blind signature is argued as follows. From the completeness of NIZK, the signer accepts π_{sk} (and does not abort) if the user computed the FHE public key pk, encryption of sk, $\mathsf{ct}_{\mathsf{sk}}$ and the proof π_{sk} correctly. From the correctness of FHE, the element $\mathsf{ct}_\sigma = \mathsf{FHE.Eval}(\mathsf{Sig.Sign}_{\mathsf{BS.sk},\rho}, \mathsf{ct}_\mu)$ decrypts to $\mathsf{Sig.Sign}_{\mathsf{BS.sk},\rho}(\mu) = \mathsf{Sig.Sign}(\mathsf{BS.sk}, \mu; \rho) = \sigma$. Hence $\mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{ct}_\sigma)$ outputs a signature $\sigma = \mathsf{Sig.Sign}(\mathsf{BS.sk}, \mu)$. It passes verification, by correctness of Sig.

We now argue that the FHE-based blind signature satisfies one more unforgeability.

Theorem 1 (One-More Unforgeability). *Assume that the underlying signature scheme* Sig *satisfies* UF-CMA *security, the fully homomorphic encryption scheme* FHE *satisfies malicious circuit privacy,* NIZK *is sound and* PKE *is correct. Then the construction in Figure 1 satisfies one more unforgeability.*

Proof. The argument proceeds via the following hybrids.

Hybrid_0: This is the genuine one-more unforgeability experiment.

Hybrid_1: In this hybrid, the challenger answers random oracle queries for H' differently. On any input a, it first samples a random value from range of H' and then programs $H'(a)$ to be that value. In effect, the challenger now samples ppk uniformly from its range and programs $H'(\alpha) = \mathsf{ppk}$.

Hybrid_2: In this hybrid, the challenger first samples $(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and then programs $H'(\alpha) = \mathsf{ppk}$ and stores psk.

Setup. $\mathsf{Gen}(1^\lambda)$: Upon input the security parameter λ, do the following:
- Invoke $(\mathsf{Sig.sk}, \mathsf{Sig.vk}) \leftarrow \mathsf{Sig.KeyGen}(1^\lambda)$.
- Set $\mathsf{ppk} = H'(\alpha)$, where α is any arbitrary publicly known value (for e.g. $\alpha = 1^\lambda$).
- Output $\mathsf{BS.sk} = \mathsf{Sig.sk}$, $\mathsf{BS.vk} = \{\mathsf{Sig.vk}, \alpha, \mathsf{ppk}\}$.

Signing. $\langle \mathcal{S}(\mathsf{BS.sk}), \mathcal{U}(\mathsf{BS.vk}, \mu) \rangle$:

1. **User:** Given the key $\mathsf{BS.vk} = \{\mathsf{Sig.vk}, \alpha, \mathsf{ppk}\}$ and a message μ, user $\mathcal{U}$ first verifies if $\mathsf{ppk} = H'(\alpha)$ and continues only if the verification passes. It then does the following:
   - It first samples $\mathsf{FHE.KeyGen}$ randomness r and an FHE key-pair $(\mathsf{pk}, \mathsf{sk}) = \mathsf{FHE.KeyGen}(1^\lambda; r)$.
   - It computes $\mathsf{ct}_\mu = \mathsf{FHE.Enc}(\mathsf{pk}, \mu)$.
   - It samples a $\mathsf{PKE.Enc}$ randomness r_e and computes $\mathsf{ct}_{\mathsf{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{sk}; r_e)$.
   - It then generates a NIZK proof π_{sk} for the following statement:
     Given $\mathsf{pk}, \mathsf{ppk}, \mathsf{ct}_{\mathsf{sk}}$, there exists a tuple (sk, r, r_e) such that
     $$(\mathsf{pk}, \mathsf{sk}) = \mathsf{FHE.KeyGen}(1^\lambda; r) \wedge \mathsf{ct}_{\mathsf{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{sk}; r_e). \tag{3.1}$$
   - It sends $(\mathsf{pk}, \pi_{\mathsf{sk}}, \mathsf{ct}_\mu, \mathsf{ct}_{\mathsf{sk}})$ to signer $\mathcal{S}$.
2. **Signer:** Upon receiving $(\mathsf{pk}, \pi_{\mathsf{sk}}, \mathsf{ct}_\mu, \mathsf{ct}_{\mathsf{sk}})$, signer $\mathcal{S}$ does the following:
   - It verifies π_{sk} and outputs $\perp$ if it fails.
   - It samples $\mathsf{Sig.Sign}$ randomness ρ and computes a ciphertext $\mathsf{ct}_\sigma = \mathsf{FHE.Eval}(\mathsf{pk}, \mathsf{Sig.Sign}_{\mathsf{BS.sk}, \rho}, \mathsf{ct}_\mu)$ that decrypts to $\mathsf{Sig.Sign}(\mathsf{BS.sk}, \mu; \rho)$.
   - It sends ct_σ to user $\mathcal{U}$.
3. **User:** Upon receiving ct_σ, user $\mathcal{U}$ computes and outputs $\sigma = \mathsf{FHE.Dec}(\mathsf{sk}, \mathsf{ct}_\sigma)$.

Verifying. The verifier computes $\mathsf{Sig.Verify}(\mathsf{BS.vk}, \mu, \sigma)$ and outputs the result.

Fig. 1 HE-based Round-Optimal Blind Signature.

Hybrid_3: This hybrid differs from the previous one in that the challenger recovers sk_i by decrypting $\mathsf{ct}_{\mathsf{sk},i}$ using psk, and aborts if it fails to do so.

Hybrid_4: In this hybrid, the challenger uses sk_i to compute $\mu_i = \mathsf{FHE.Dec}(\mathsf{sk}_i, \mathsf{ct}_{\mu_i})$ for all $i \in [Q]$. It then changes the way the Q ciphertexts $\mathsf{ct}_{\sigma_1} \ldots, \mathsf{ct}_{\sigma_Q}$ are generated. It first signs the messages $\mu_1, \ldots, \mu_Q$ to obtain signatures $\sigma_1, \ldots, \sigma_Q$ (in the clear). Then, it encrypts $\sigma_1, \ldots, \sigma_Q$ using FHE to obtain $\mathsf{ct}'_{\sigma_1}, \ldots, \mathsf{ct}'_{\sigma_Q}$. It runs $\mathsf{FHE.Eval}(\mathsf{pk}_i, \mathcal{C}^0, (\mathsf{ct}'_{\sigma_i}, \star))$ for all $i \in [Q]$ to obtain $\mathsf{ct}_{\sigma_1} \ldots, \mathsf{ct}_{\sigma_Q}$. Here $\mathcal{C}^0$ is a dummy circuit that has the same depth and number of inputs as the $\mathsf{Sig.Sign}$ signing circuit and outputs its first input, and $\star$ represents as many independent encryptions of 0 as required (see Definition 8).

Indistinguishability of hybrids: We show that consecutive hybrids are indistinguishable.

The only difference between Hybrid_0 and Hybrid_1 is that in Hybrid_1, $H'(\alpha)$ is set in the reverse order, i.e. first a random value from the the range of H' is sampled uniformly and then $H'(\alpha)$ is programmed to be that value. The two hybrids are therefore indistinguishable in random oracle model.

The only difference between Hybrid_1 and Hybrid_2 is that ppk is a uniformly random value in Hybrid_1, while in Hybrid_2, it is an output of $\mathsf{PKE.KeyGen}(1^\lambda)$ algorithm. Since, ppk generated using $\mathsf{PKE.KeyGen}(1^\lambda)$ is computationally indistinguishable from a random value (by design), the two hybrids are also computationally indistinguishable.

Claim 2 *Assume that* PKE *is correct and* NIZK *is sound. Then* Hybrid_2 *and* Hybrid_3 *are indistinguishable.*

Proof *(Claim 2).* The two hybrids differ only in that in Hybrid_3, the challenger decrypts $\mathsf{ct}_{\mathsf{sk}_i}$ to recover sk_i and aborts if it fails to do so. From the soundness of NIZK, $\mathsf{ct}_{\mathsf{sk}_i}$ must be a valid PKE ciphertext (otherwise π_{sk_i} does not verify and the game is aborted in both the hybrids). Hence, from the correctness of PKE the challenger successfully recovers sk_i and does not abort. Note that the extracted value is not used in any computation and hence the adversary's view does not change. □

Claim 3 *Assume that* NIZK *is sound and* FHE *satisfies malicious circuit privacy. Then* Hybrid_3 *and* Hybrid_4 *are computationally indistinguishable.*

Proof *(Claim 3).* For all $i \in [Q]$, the adversary is given $\mathsf{ct}_{\sigma_i} = \mathsf{FHE.Eval}(\mathsf{pk}_i, \mathsf{Sig.Sign}_{\mathsf{Sig.sk},\rho_i}, \mathsf{ct}_{\mu_i})$ in Hybrid_3. In Hybrid_4, the adversary is given $\mathsf{ct}_{\sigma_i} = \mathsf{FHE.Eval}(\mathsf{pk}_i, \mathcal{C}^0, \mathsf{ct}'_{\sigma_i})$, where $\mathsf{ct}'_{\sigma_i} = \mathsf{FHE.Enc}(\mathsf{pk}_i, \sigma_i)$ and $\sigma_i = \mathsf{Sig.Sign}(\mathsf{Sig.sk}, \mu_i; \rho)$.

From the soundness of NIZK, for all $i \in [Q]$, sk_i recovered by the challenger indeed correspond to the public key pk_i. Hence, from the correctness of FHE.Dec, the direct signature σ_i computed by the challenger in Hybrid_4 is indeed equal to $\mathsf{Sig.Sign}_{\mathsf{Sig.sk},\rho}(\mu_i)$, where μ_i is the message encrypted by ct_i, in Hybrid_3. As a result, the adversary's views:

$$\begin{aligned} &(\mathsf{FHE.Eval}(\mathsf{pk}_i, \mathsf{Sig.Sign}_{\mathsf{Sig.sk},\rho}, \mathsf{ct}_i), \mathsf{ct}_i, \mathsf{pk}_i, \mathsf{sk}_i) \quad \text{in } \mathsf{Hybrid}_3 \\ \text{and } &(\mathsf{FHE.Eval}(\mathsf{pk}_i, \mathcal{C}^0, \mathsf{ct}'_{\sigma_i}), \mathsf{ct}_i, \mathsf{pk}_i, \mathsf{sk}_i) \quad \text{in } \mathsf{Hybrid}_4 \end{aligned}$$

are indistinguishable due to malicious circuit privacy of FHE.

Note here that semi-honest circuit privacy would not suffice, as we are not certain that the ciphertexts are properly generated (or even that sk_i is properly generated, as the NIZK only guarantees its well-formedness). □

The theorem statement follows from Lemma 1, which shows that if Sig satisfies UF-CMA security, then the adversary can win in Hybrid_4 only with negligible probability.

Lemma 1. *Assume that* Sig *satisfies* UF-CMA *security. Then the advantage of the adversary in the one more unforgeability game is negligible in* Hybrid_4.

Proof *(Lemma 1).* Assume there exists an adversary $\mathcal{U}^*$ that wins the one more unforgeability game in Hybrid_4. Then we build an adversary $\mathcal{B}$ against the UF-CMA security of the underlying signature scheme Sig. Adversary $\mathcal{B}$ does the following:

1. It obtains $\mathsf{Sig.vk}$ from the signature challenger, samples $(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$, programs $H'(\alpha) = \mathsf{ppk}$ and defines $\mathsf{BS.vk} = \{\mathsf{Sig.vk}, \alpha, \mathsf{ppk}\}$ and forwards it to $\mathcal{U}^*$.
2. It runs the signing protocol with adversary $\mathcal{U}^*$, simulating the BS signer as follows:
   (a) Adversary $\mathcal{U}^*$ outputs the FHE public keys $\mathsf{pk}_1, \ldots, \mathsf{pk}_Q$, their PKE encryptions $\mathsf{ct}_{\mathsf{sk}_1}, \ldots, \mathsf{ct}_{\mathsf{sk}_Q}$ along with proofs $\pi_{\mathsf{sk}_1}, \ldots, \pi_{\mathsf{sk}_Q}$. Adversary $\mathcal{B}$ verifies the proofs and if all of them verifies, it proceeds further and recovers $\mathsf{sk}_i = \mathsf{PKE.Dec}(\mathsf{psk}, \mathsf{ct}_{\mathsf{sk}_i})$ for all $i \in [Q]$.
   (b) Adversary $\mathcal{U}^*$ also outputs Q ciphertexts $\mathsf{ct}_{\mu_1}, \ldots, \mathsf{ct}_{\mu_Q}$. Adversary $\mathcal{B}$ uses sk_i to decrypt ct_{μ_i} to obtain μ_i for all $i \in [Q]$.
   (c) Adversary $\mathcal{B}$ sends $\mu_1, \ldots, \mu_Q$ to the UF-CMA signature challenger and obtains signatures $\sigma_1, \ldots, \sigma_Q$.
   (d) It constructs $\mathsf{ct}_{\sigma_1} \ldots, \mathsf{ct}_{\sigma_Q}$ from $\sigma_1, \ldots, \sigma_Q$ as in the previous hybrid and returns these to $\mathcal{U}^*$.
   (e) When $\mathcal{U}^*$ outputs $Q+1$ message-signature pairs (μ_i, σ_i) for $i \in [Q+1]$ that pass verification and such that the μ_i's are distinct, it outputs any of these for which μ_i had not been queried to the UF-CMA signature challenger.

The success of $\mathcal{U}^*$ translates to the success of $\mathcal{B}$ in the UF-CMA game.

□

□

We now argue that the FHE-based blind signature satisfies very honest signer blindness.

Theorem 4 (Very Honest Signer Blindness). *Assume that the* NIZK *is zero knowledge,* FHE *is correct and secure and* PKE *is secure. Then the construction in Figure 1 satisfies very honest signer blindness.*

Proof. The argument proceeds via a sequence of hybrids.

Hybrid_0: This is the genuine very honest signer blindness experiment.

1. The challenger generates $(\mathsf{BS.sk} = \mathsf{Sig.sk}, \mathsf{BS.vk} = \{\mathsf{Sig.vk}, \alpha, \mathsf{ppk}\})$ and returns these to the adversary $\mathcal{S}^*$.
2. The signer $\mathcal{S}^*$ outputs two messages μ_0 and μ_1. The challenger picks a uniform bit b.
3. The signer $\mathcal{S}^*$ interacts concurrently with $\mathcal{U}_0(\mathsf{BS.vk}, \mu_b)$ and $\mathcal{U}_1(\mathsf{BS.vk}, \mu_{\bar{b}})$, played by the challenger as follows:
   (a) User $\mathcal{U}_0$ (resp. $\mathcal{U}_1$) generates the FHE public and secret keys $(\mathsf{pk}_0, \mathsf{sk}_0)$ (resp. $(\mathsf{pk}_1, \mathsf{sk}_1)$) honestly, computes $\mathsf{ct}_{\mathsf{sk}_0} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{sk}_0)$ (resp. $\mathsf{ct}_{\mathsf{sk}_1} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{sk}_1)$) along with proof π_{sk_0} (resp. π_{sk_1}) for statement in equation (3.1).
   (b) Additionally, users $\mathcal{U}_0$ and $\mathcal{U}_1$ provide their respective FHE ciphertexts $\mathsf{ct}_{\mu_b} = \mathsf{FHE.Enc}(\mathsf{pk}_0, \mu_b)$ and $\mathsf{ct}_{\mu_{\bar{b}}} = \mathsf{FHE.Enc}(\mathsf{pk}_1, \mu_{\bar{b}})$.
   (c) The challenger evaluates the signing algorithm homomorphically on ciphertexts ct_{μ_b} and $\mathsf{ct}_{\mu_{\bar{b}}}$ to obtain ct'_{μ_b} and $\mathsf{ct}'_{\mu_{\bar{b}}}$, respectively. (recall that this step is performed by the adversary in the honest signer blindness experiment, but by the challenger in the very honest signer blindness experiment) ; the challenger gives ct'_{μ_b} and $\mathsf{ct}'_{\mu_{\bar{b}}}$ to the signer $\mathcal{S}^*$, as well as all the intermediate values of their computation.
   (d) After obtaining the evaluated FHE ciphertexts, ct'_{μ_b} and $\mathsf{ct}'_{\mu_{\bar{b}}}$ users $\mathcal{U}_0$ and $\mathcal{U}_1$ decrypt them using the FHE secret keys sk_0 and sk_1 to obtain signatures σ_b and $\sigma_{\bar{b}}$ respectively.
   (e) The signer $\mathcal{S}^*$ is given σ_0, σ_1.
   (f) The signer $\mathcal{S}^*$ outputs its guess bit b'.

Hybrid_1: In this Hybrid, the challenger answers the oracle queries for H' differently - for any input a, it first samples a random value from the range of H' and then programs $H'(a)$ to be that value. In effect, the challenger now samples ppk uniformly from its range and programs $H'(\alpha) = \mathsf{ppk}$.

Hybrid_2: In this Hybrid, the challenger sets $H'(\alpha)$ differently - it first runs $(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and then programs $H'(\alpha) = \mathsf{ppk}$.

Hybrid_3: In this hybrid, the proofs π_{sk_0} and π_{sk_1} are replaced with simulated proofs.

Hybrid_4: In this hybrid, at Step 3(d), the signatures σ_b and $\sigma_{\bar{b}}$ are generated by using Sig.Sign directly on μ_b and $\mu_{\bar{b}}$, respectively.

Hybrid_5: In this hybrid, $\mathsf{ct}_{\mathsf{sk}_0}$ and $\mathsf{ct}_{\mathsf{sk}_1}$ are replaced with encryptions of zero.

Hybrid_6: In this hybrid, we replace $\mathsf{FHE.Enc}(\mathsf{pk}_0, \mu_b)$ by $\mathsf{FHE.Enc}(\mathsf{pk}_0, 0)$ and $\mathsf{FHE.Enc}(\mathsf{pk}_1, \mu_{\bar{b}})$ by $\mathsf{FHE.Enc}(\mathsf{pk}_1, 0)$.

Indistinguishability of hybrids.

1. The only difference between Hybrid_0 and Hybrid_1 is that in the latter hybrid, $H'(a)$ on any input a is programmed to be uniformly random value. Hence, the two hybrids are indistinguishable in ROM.
2. Hybrid_1 and Hybrid_2 differ only in the value of $H'(\alpha)$. It is a random value in the former hybrid while a PKE public key in the latter one. Hence, the two hybrids are indistinguishable by design, since we use a PKE scheme in which the public key is computationally indistinguishable from a uniformly random value in its range.
3. The indistinguishability between Hybrid_2 and Hybrid_3 follows from the zero-knowledge property of the underlying NIZKAoK.
4. The only difference between Hybrid_3 and Hybrid_4 is that in the former, we have $\sigma_b = \mathsf{FHE.Dec}(\mathsf{sk}_0, \mathsf{ct}'_{\mu_b})$ (resp. $\sigma_{\bar{b}} = \mathsf{FHE.Dec}(\mathsf{sk}_1, \mathsf{ct}'_{\mu_{\bar{b}}})$) while in the latter, the signatures σ_b and $\sigma_{\bar{b}}$ are generated using Sig.Sign directly. Indistinguishability follows from the correctness of FHE.
5. The only difference between Hybrid_4 and Hybrid_5 is that in the former, $\mathsf{ct}_{\mathsf{sk}_\beta}$ is a PKE encryption of sk_β for $\beta \in \{0, 1\}$, while in the latter it encrypts zero. The indistinguishability follows from the CPA security of PKE. That is, if $\mathcal{S}^*$ can distinguish between the two hybrids with non-negligible advantage then there exists an adversary $\mathcal{B}$ (playing the role of BS challenger) against CPA security of PKE.
   Here, we show the reduction for an intermediate hybrid, $\mathsf{Hybrid}_{4.5}$ in which $\mathsf{ct}_{\mathsf{sk}_0}$ encrypts $\mathbf{0}$, while $\mathsf{ct}_{\mathsf{sk}_1}$ encrypts sk_1. We can then go from $\mathsf{Hybrid}_{4.5}$ to Hybrid_5 in the same way.
   The reduction $\mathcal{B}$ is defined as follows:

(a) Upon receiving the public key ppk from the PKE challenger, the reduction $\mathcal{B}$ (BS challenger) samples $(\mathsf{Sig.sk}, \mathsf{Sig.vk}) \leftarrow \mathsf{Sig.KeyGen}(1^\lambda)$, programs $H'(\alpha) = \mathsf{ppk}$, sets $\mathsf{BS.sk} = \mathsf{Sig.sk}$, $\mathsf{BS.vk} = (\mathsf{Sig.vk}, \alpha, \mathsf{ppk})$ and invokes S^*.

(b) The signer $\mathcal{S}^*$ outputs two messages μ_0 and μ_1. $\mathcal{B}$ picks a uniform bit b.

(c) The signer $\mathcal{S}^*$ interacts concurrently with $\mathcal{U}_0(\mathsf{BS.vk}, \mu_b)$ and $\mathcal{U}_1(\mathsf{BS.vk}, \mu_{\bar{b}})$, played by $\mathcal{B}$ as follows:

i. User $\mathcal{U}_0$ (resp. $\mathcal{U}_1$) generates the FHE public and secret keys $(\mathsf{pk}_0, \mathsf{sk}_0)$ (resp. $(\mathsf{pk}_1, \mathsf{sk}_1)$). $\mathcal{U}_1$ computes $\mathsf{ct}_{\mathsf{sk}_1} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{sk}_1)$. $\mathcal{B}$ sends $\mathsf{sk}_0, \mathbf{0}$ as challenge messages to the to PKE challenger and sets the ciphertext returned by the PKE challenger to be $\mathsf{ct}_{\mathsf{sk}_0}$. It then generates simulated proofs π_{sk_0} and π_{sk_1}. $\mathcal{S}^*$ is given $(\mathsf{pk}_0, \mathsf{ct}_{\mathsf{sk}_0}, \pi_{\mathsf{sk}_0})$ and $(\mathsf{pk}_1, \mathsf{ct}_{\mathsf{sk}_1}, \pi_{\mathsf{sk}_1})$.

ii. Additionally, users $\mathcal{U}_0$ and $\mathcal{U}_1$ provide their respective ciphertexts $\mathsf{ct}_{\mu_b} = \mathsf{FHE.Enc}(\mathsf{pk}_0, \mu_b)$ and $\mathsf{ct}_{\mu_{\bar{b}}} = \mathsf{FHE.Enc}(\mathsf{pk}_1, \mu_{\bar{b}})$.

iii. $\mathcal{B}$ evaluates the signing algorithm homomorphically on ciphertexts ct_{μ_b} and $\mathsf{ct}_{\mu_{\bar{b}}}$ to obtain ct'_{μ_b} and $\mathsf{ct}'_{\mu_{\bar{b}}}$, respectively (recall that this step is performed by the adversary in the honest signer blindness experiment, but by the challenger in the very honest signer blindness experiment); $\mathcal{B}$ gives ct'_0 and ct'_1 to the signer $\mathcal{S}^*$, as well as all intermediate values of their computation.

iv. $\mathcal{B}$ computes signatures σ_b and $\sigma_{\bar{b}}$ on messages μ_b and $\mu_{\bar{b}}$ using $\mathsf{Sig.sk}$.

v. The signer $\mathcal{S}^*$ is given σ_0, σ_1.

vi. The signer $\mathcal{S}^*$ outputs its guess bit b'.

vii. $\mathcal{B}$ returns b' to the PKE challenger.

Clearly if the PKE challenger returns encryption of sk_0, then the view of $\mathcal{S}^*$ corresponds to Hybrid_4 else, $\mathsf{Hybrid}_{4.5}$. Hence, if $\mathcal{S}^*$ wins then so does $\mathcal{B}$ against the PKE challenger.

6. The indistinguishability between Hybrid_5 and Hybrid_6 follows from the semantic security of FHE. The proof is a standard reduction to the semantic security. Note that due to the previous hybrids, the blindness challenger does not need the FHE secret key for any operations and can obtain the FHE public keys and ciphertexts from the FHE challenger. The arguments π_{sk} are simulated, and the signatures σ_b and $\sigma_{\bar{b}}$ are computed directly using the signing key $\mathsf{Sig.sk}$.

To complete the proof of the theorem, it suffices to note that the advantage of the adversary in Hybrid_6 is 0 since the bit b is information theoretically hidden. □

Attack for Honest Signer Blindness. Note that the scheme of Figure 1 may not satisfy the stronger notion of honest signer blindness, which differs from very honest signer blindness in that the signer can deviate from the protocol during the signing phase. For example, assume the signature scheme Sig is obtained by derandomizing a probabilistic signature scheme Sig_0, as follows: at key generation, one samples a PRF key k and appends it to the signing key; when signing a message, one first creates a pseudo-random string using k and the message to be signed, and then runs $\mathsf{Sig}_0.\mathsf{Sign}$ with this pseudo-random string. Then, to break honest signer blindness, the signer could use two distinct PRF keys k_0 and k_1 for the two executions of the signing protocol. At the end, it could compute the four possible signatures from (μ_0, μ_1) and (k_0, k_1) and assess which PRF key has been used in which execution, by matching σ_0 and σ_1 with those four signatures.

4 Round-Optimal Schnorr based Blind Signature

In this section, we provide a round optimal blind signature based on the Schnorr signature and fully homomorphic encryption FHE. We first recap the Schnorr signature scheme.

4.1 Schnorr Signature Scheme

We recap the Schnorr signature scheme $\mathsf{Sch} = (\mathsf{Sch.KeyGen}, \mathsf{Sch.Sign}, \mathsf{Sch.Verify})$ in Figure 2. The algorithm $\mathsf{Sch.Sign}$ is divided into two subroutines, namely, $(\mathsf{Sch.Sign}^r, \mathsf{Sch.Sign}^s)$.

Correctness follows from the equalities $g^s = g^{k+\mathsf{Sch.sk}\cdot H(\mu,r)} = r \cdot \mathsf{Sch.vk}^{H(\mu,r)}$. For security, please see [PS00].

$\mathsf{Sch.KeyGen}(1^\lambda)$: Upon input the security parameter λ, set a cyclic group $G = \langle g \rangle$ of prime order q and a hash function $H : \{0,1\}^* \to \mathbb{Z}_q^*$ such that the scheme is secure; then do the following:

- $\mathsf{sk} \leftarrow \mathbb{Z}_q^*$.
- Output $\mathsf{Sch.vk} = g^{\mathsf{sk}} \in G$, $\mathsf{Sch.sk} = \mathsf{sk}$.

$\mathsf{Sch.Sign}(\mathsf{Sch.sk}, \mu)$. Upon input the signing key $\mathsf{Sch.sk}$ and a message μ, do the following:

- $k \leftarrow \mathbb{Z}_q^*$.
- $\mathsf{Sch.Sign}^r(k)$: Output $r = g^k \in G$.
- $\mathsf{Sch.Sign}^s(\mathsf{Sch.sk}, \mu; k)$: For $r \leftarrow \mathsf{Sch.Sign}^r(k)$, output $s = k + \mathsf{Sch.sk} \cdot H(\mu, r)$.
- Output $\sigma = (r, s)$.

$\mathsf{Sch.Verify}(\mathsf{Sch.vk}, \mu, \sigma = (r, s))$. Upon input the verification key $\mathsf{Sch.vk}$, a message μ and a signature $\sigma = (r, s)$, check $g^s = r \cdot \mathsf{Sch.vk}^{H(\mu, r)}$. If the check passes, then accept, else reject.

Fig. 2 Schnorr Signature Scheme.

4.2 Two-Round Blind Signature from Schnorr and FHE

Our construction uses the following building blocks:

- A UF-CMA Schnorr signature scheme $\mathsf{Sch} = (\mathsf{Sch.KeyGen}, \mathsf{Sch.Sign}{=}(\mathsf{Sch.Sign}^r, \mathsf{Sch.Sign}^s), \mathsf{Sch.Verify})$.
- An IND-CPA secure public key encryption scheme $\mathsf{PKE} = (\mathsf{PKE.KeyGen}, \mathsf{PKE.Enc}, \mathsf{PKE.Dec})$ having the property that the public key generated by $\mathsf{PKE.KeyGen}$ is indistinguishable from a uniformly sampled value from its range.
- A maliciously circuit private fully homomorphic encryption scheme (see Def. 8) $\mathsf{FHE} = (\mathsf{FHE.KeyGen}, \mathsf{FHE.Enc}, \mathsf{FHE.Dec}, \mathsf{FHE.Eval})$ that is capable of homomorphically evaluating circuits with depth up to the depth of circuit $\mathcal{C}_{\mathsf{Sch.sk}, k_s}$, which has $\mathsf{Sch.sk}, k_s$ hardwired and takes a message μ and a group element r_u as inputs and returns $k_s + \mathsf{Sch.sk} \cdot H(\mu, r_u \cdot g^{k_s})$.
- A NIZK argument system $\mathsf{NIZK}_{\mathsf{FHE}}$ for the statements of the form given in equation 4.1.
- A NIZK argument system $\mathsf{NIZK}_{\mathsf{ct}}$ for the well-formedness of FHE ciphertexts, i.e., for statements of the form: Given ct, hpk, there exists μ, r such that $\mathsf{FHE.Enc}(\mathsf{hpk}, \mu; r) = \mathsf{ct}$.
- A NIZK argument of knowledge system $\mathsf{NIZKAoK}_{\mathsf{DL}}$ for the knowledge of discrete logarithm of an element in a group, i.e., for statements of the form : Given a group G and generator g and $r \in G$, there exists k such that $r = g^k \in G$.
- A hash function $H' : \{0,1\}^* \to \mathcal{PK}$ modeled as random oracle. Here, $\mathcal{PK}$ is the space of public keys of PKE.

We provide our construction in Figure 3.

Correctness. The completeness follows from the completeness of the underlying primitives. In an honest execution of the scheme, from the completeness of $\mathsf{NIZK}_{\mathsf{FHE}}$, the signer accepts π_{sk} (and does not abort). Similarly, from the completeness of the $\mathsf{NIZK}_{\mathsf{ct}}$ proof system, the proofs π_{r_u} and π_μ verify correctly if they were computed correctly. From the correctness of FHE, the signing circuit evaluated on the encryptions ct_μ and ct_{r_u}, of the message and the user side randomness respectively, yields a ciphertext $\mathsf{ct}_s = \mathsf{FHE.Eval}(\mathsf{hpk}, \mathcal{C}_{\mathsf{Sch.sk}, k_s}, \mathsf{ct}_\mu, \mathsf{ct}_{r_u})$ which decrypts to $s = k_s + \mathsf{Sch.sk} \cdot H(\mu, r_u \cdot g^{k_s})$ computed as $\mathsf{FHE.Dec}(\mathsf{hsk}, \mathsf{ct}_s)$. Thus $k_u + s = k_u + k_s + \mathsf{Sch.sk} \cdot H(\mu, r_u \cdot g^{k_s}) = \mathsf{Sch.Sig}^s(\mathsf{Sch.sk}, \mu; k_u + k_s)$. As $r = \mathsf{Sch.Sign}^r(k_u + k_s)$ as before, by correctness of the Schnorr signature scheme Sch, the signature $\sigma = (r, k_u + s)$ passes verification.

One More Unforgeability. We now argue that the blind Schnorr signature scheme satisfies one more unforgeability.

Setup. $\mathsf{Gen}(1^\lambda)$: Upon input the security parameter λ, do the following:
- Invoke $(\mathsf{Sch.sk}, \mathsf{Sch.vk}) \leftarrow \mathsf{Sch.KeyGen}(1^\lambda)$.
- Set $\mathsf{ppk} = H'(\alpha)$, where α is any arbitrary publicly known value (for e.g. $\alpha = 1^\lambda$).
- Output $\mathsf{BS.sk} = \mathsf{Sch.sk}$, $\mathsf{BS.vk} = \{\mathsf{Sch.vk}, \alpha, \mathsf{ppk}\}$.

Signing. $\langle \mathcal{S}(\mathsf{BS.sk}), \mathcal{U}(\mathsf{BS.vk}, \mu)\rangle$:

1. **User:** Given the key $\mathsf{BS.vk}$ and a message μ, user $\mathcal{U}$ first verifies if $\mathsf{ppk} = H'(\alpha)$ and continues only if the verification passes. It then does the following:
   - It first samples $\mathsf{FHE.KeyGen}$ randomness ρ and an FHE key-pair $(\mathsf{hpk}, \mathsf{hsk})$=$\mathsf{FHE.KeyGen}(1^\lambda; \rho)$.
   - It then samples a $\mathsf{PKE.Enc}$ randomness r_e and computes $\mathsf{ct_{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{hsk}; r_e)$.
   - It then generates a $\mathsf{NIZK_{FHE}}$ proof π_{sk} for the following statement:
     Given $\mathsf{hpk}, \mathsf{ppk}, \mathsf{ct_{sk}}$, there exists a tuple $(\mathsf{hsk}, \rho, r_e)$ such that
     $$(\mathsf{hpk}, \mathsf{hsk}) = \mathsf{FHE.KeyGen}(1^\lambda; \rho) \wedge \mathsf{ct_{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{hsk}; r_e). \tag{4.1}$$
   - It samples $\mathsf{Sch.Sign}$ partial randomness $k_u \leftarrow \mathbb{Z}_q^*$ and computes $r_u = g^{k_u}$, $\mathsf{ct}_{r_u} = \mathsf{FHE.Enc}(\mathsf{hpk}, r_u)$ and $\mathsf{ct}_\mu = \mathsf{FHE.Enc}(\mathsf{hpk}, \mu)$.
   - It computes NIZK proofs π_{r_u} and π_μ that ct_{r_u} and ct_μ are well-formed, using $\mathsf{NIZK_{ct}}$.
   - It sends $(\mathsf{hpk}, \mathsf{ct_{sk}}, \pi_{\mathsf{sk}}, \mathsf{ct}_{r_u}, \mathsf{ct}_\mu, \pi_{r_u}, \pi_\mu)$ to signer $\mathcal{S}$.
2. **Signer:** Upon receiving $(\mathsf{hpk}, \mathsf{ct_{sk}}, \pi_{\mathsf{sk}}, \mathsf{ct}_{r_u}, \mathsf{ct}_\mu, \pi_{r_u}, \pi_\mu)$, signer $\mathcal{S}$ does the following:
   - It verifies π_{sk}, π_{r_u} and π_μ and outputs $\perp$ if any of these fails.
   - It samples $\mathsf{Sch.Sign}$ partial randomness $k_s \leftarrow \mathbb{Z}_q$ and computes $\mathsf{Sch.Sign}^r(k_s)$, i.e. $r_s = g^{k_s}$; it then generates a $\mathsf{NIZKAoK_{DL}}$ proof π_{k_s} that r_s is well-formed with respect to k_s, i.e., there exists k_s such that $r_s = g^{k_s}$.
   - It computes a ciphertext $\mathsf{ct}_s = \mathsf{FHE.Eval}(\mathsf{hpk}, \mathcal{C}_{\mathsf{BS.sk}, k_s}, \mathsf{ct}_\mu, \mathsf{ct}_{r_u})$, which returns the encryption of $k_s + \mathsf{BS.sk} \cdot H(\mu, r_u \cdot g^{k_s})$.
   - It sends $\mathsf{ct}_s, r_s, \pi_{k_s}$ to user $\mathcal{U}$.
3. **User:** Upon receiving $\mathsf{ct}_s, r_s, \pi_{k_s}$, user $\mathcal{U}$ does the following:
   - It first verifies π_{k_s} and outputs $\perp$ if it fails.
   - It computes $r = g^{k_u} \cdot r_s$ and $s = \mathsf{FHE.Dec}(\mathsf{hsk}, \mathsf{ct}_s)$.
   - It computes $\mathsf{Sch.Verify}(\mathsf{BS.vk}, \mu, (r, k_u + s))$ and outputs $\perp$ if it fails.
   - It outputs $\sigma = (r, k_u + s)$.

Verifying. The verifier computes $\mathsf{Sch.Verify}(\mathsf{BS.vk}, \mu, \sigma = (r, k_u + s))$ and outputs the result.

Fig. 3 $\mathsf{BS}^{\mathsf{Sch}}$: HE-based Round-Optimal Blind Schnorr Signature.

Theorem 5. *Assume that the underlying signature scheme* Sch *satisfies* $\mathsf{UF\text{-}CMA}$ *security,* FHE *and* PKE *are correct and* FHE *satisfies malicious circuit privacy,* $\mathsf{NIZK_{FHE}}$ *and* $\mathsf{NIZK_{ct}}$ *are sound and* $\mathsf{NIZKAoK_{DL}}$ *is zero knowledge. Then the construction in Figure 3 satisfies one more unforgeability.*

Proof. The proof is argued via the following hybrids:

Hybrid_0: This is the genuine one-more unforgeability experiment between the challenger (simulated signer $\mathcal{S}$) and the adversary (malicious user $\mathcal{U}^*$).

1. $\mathsf{BS.KeyGen}(1^\lambda)$ outputs $(\mathsf{BS.sk}, \mathsf{BS.vk})$, and $\mathcal{U}^*$ is given $\mathsf{BS.vk}$.
2. $\mathcal{U}^*$ interacts concurrently with Q instances $\mathcal{S}^1_{\mathsf{BS.sk}}, \ldots, \mathcal{S}^Q_{\mathsf{BS.sk}}$. Each instance $i \in [Q]$ with $\mathcal{S}^i_{\mathsf{BS.sk}}$ consists of transcripts $(\mathsf{hpk}_i, \mathsf{ct_{sk}}_i, \pi_{\mathsf{sk}_i}, \mathsf{ct}_{r_u,i}, \mathsf{ct}_{\mu_i}, \pi_{r_u,i}, \pi_{\mu_i})$ generated by $\mathcal{U}^*$ itself, $(\mathsf{ct}_{s,i}, r_{s,i}, \pi_{k_s,i})$ sent by $\mathcal{S}^i_{\mathsf{BS.sk}}$, and $\sigma_i = (r_i, k_{u,i} + s_i)$ computed by $\mathcal{U}^*$.
3. $\mathcal{U}^*$ outputs $(\mu_1, \sigma_1, \ldots, \mu_{Q+1}, \sigma_{Q+1})$.

Hybrid_1: In this hybrid, the challenger answers random oracle queries for H' differently. On any input a, it first samples a random value from range of H' and then programs $H'(a)$ to be that value. In effect, in this hybrid the challenger samples ppk randomly from its range and then programs $H'(\alpha) = \mathsf{ppk}$.

Hybrid_2: In this hybrid, the challenger first samples $(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$ and then programs $H'(\alpha) = \mathsf{ppk}$ and stores psk.

Hybrid_3: In this hybrid, the proof $\pi_{k_{s,i}}$ is now a simulated proof for all $i \in [Q]$.

Hybrid_4: In this hybrid, the ciphertexts $\mathsf{ct}_{s,i}$ are computed differently as follows: for all $i \in [Q]$. The challenger extracts hsk_i as $\mathsf{PKE.Dec}(\mathsf{psk}, \mathsf{ct}_{\mathsf{sk}_i})$. Then it computes $r_{u,i}^{\mathsf{Dec}} = \mathsf{FHE.Dec}(\mathsf{hsk}_i, \mathsf{ct}_{r_{u,i}})$ and $\mu_i^{\mathsf{Dec}} = \mathsf{FHE.Dec}(\mathsf{hsk}_i, \mathsf{ct}_{\mu_i})$. Then it computes $s_i^{\mathsf{Dec}} = k_{s,i} + \mathsf{BS.sk} \cdot H(\mu_i^{\mathsf{Dec}}, r_{u,i}^{\mathsf{Dec}} \cdot g^{k_{s,i}})$ and replaces $\mathsf{ct}_{s,i}$ by the encryptions of s_i^{Dec} computed as follows: it first computes $\mathsf{ct}'_{s,i} = \mathsf{FHE.Enc}(\mathsf{hpk}_i, s_i^{\mathsf{Dec}})$ and then performs homomorphic evaluation on $\mathsf{ct}'_{s,i}$ as: $\mathsf{ct}_{s,i} = \mathsf{FHE.Eval}(\mathsf{hpk}_i, \mathcal{C}^0, (\mathsf{ct}'_{s,i}, \star))$. Here, $\mathcal{C}^0$ is a dummy circuit of the same depth and arity as that of $\mathcal{C}_{\mathsf{BS.sk},k_{s,i}}$ which outputs its first input, ignoring the rest and $\star$ represents as many encryptions of 0's as required.

Indistinguishability of hybrids: The argument for indistinguishability between Hybrid_0, Hybrid_1 and Hybrid_2 is same as that in the proof of Theorem 1. They are indistinguishable in random oracle model and from the choice of a PKE scheme in which the public key ppk is indistinguishable from uniform. Indistinguishability of Hybrid_2 and Hybrid_3 relies on $\mathsf{NIZKAoK}_{\mathsf{DL}}$ being zero knowledge, while indistinguishability of Hybrid_3 and Hybrid_4 relies on soundness of $\mathsf{NIZK}_{\mathsf{ct}}$ and $\mathsf{NIZK}_{\mathsf{FHE}}$, correctness of PKE and FHE and malicious circuit privacy of FHE. We argue indistinguishability between Hybrid_2, Hybrid_3 and Hybrid_4 via the following claims.

Claim 6 *Assume that* $\mathsf{NIZKAoK}_{\mathsf{DL}}$ *is zero knowledge. Then* Hybrid_2 *and* Hybrid_3 *are indistinguishable.*

Proof *(Claim 6).* The only difference between Hybrid_2 and Hybrid_3 is that the proofs $\pi_{k_{s,i}}$ for $i \in [Q]$ are real in the former and simulated in the latter. Hence, by zero knowledge property of $\mathsf{NIZKAoK}_{\mathsf{DL}}$, the claim follows. □

Claim 7 *Assume that* $\mathsf{NIZK}_{\mathsf{ct}}$ *and* $\mathsf{NIZK}_{\mathsf{FHE}}$ *are sound and* PKE *and* FHE *are correct and* FHE *satisfies malicious circuit privacy. Then* Hybrid_3 *and* Hybrid_4 *are indistinguishable.*

Proof *(Claim 7).* The two hybrids differ in the computation of $\mathsf{ct}_{s,i}$ for $i \in [Q]$. In addition, in Hybrid_4 the continuation of the game is also conditioned on successful extraction of hsk_i by the challenger. Firstly, we note that for all $i \in [Q]$, if the proof π_{sk_i} verifies (which should happen for the game to continue further), then by the soundness of $\mathsf{NIZK}_{\mathsf{FHE}}$, $\mathsf{ct}_{\mathsf{sk}_i}$ must be a valid PKE encryption of a valid hsk_i and hence from the correctness of PKE, the challenger successfully recovers correct hsk_i and does not abort.

Next we observe that in Hybrid_3, $\mathsf{ct}_{s,i} = \mathsf{FHE.Eval}(\mathsf{hpk}_i, \mathcal{C}_{\mathsf{BS.sk},k_s}, (\mathsf{ct}_{\mu_i}, \mathsf{ct}_{r_{u,i}}))$ and in Hybrid_4, $\mathsf{ct}_{s,i} = \mathsf{FHE.Eval}(\mathsf{hpk}_i, \mathcal{C}^0, (\mathsf{ct}'_{s,i}, \star))$, where $\mathsf{ct}'_{s,i} = \mathsf{FHE.Enc}(\mathsf{hpk}_i, s_i^{\mathsf{Dec}})$ and $s_i^{\mathsf{Dec}} = \mathcal{C}_{\mathsf{BS.sk},k_{s,i}}(\mu_i^{\mathsf{Dec}}, r_u^{\mathsf{Dec}})$. The soundness of $\mathsf{NIZK}_{\mathsf{ct}}$ ensures that ct_{μ_i} and $\mathsf{ct}_{r_{u,i}}$ are a valid ciphertexts and correctness of FHE ensures that underlying plaintexts are recovered correctly. This implies that s_i^{Dec} computed directly in Hybrid_4 is indeed equal to $\mathcal{C}_{\mathsf{BS.sk},k_{s,i}}(\mu_i, r_{u,i})$, where μ_i and $r_{u,i}$ are the underlying plaintexts in ct_{μ_i} and $\mathsf{ct}_{r_{u,i}}$, respectively. As a result, the adversary's views:

$$(\mathsf{FHE.Eval}(\mathsf{hpk}_i, \mathcal{C}_{\mathsf{BS.sk},k_{s,i}}, (\mathsf{ct}_{\mu_i}, \mathsf{ct}_{r_{u,i}})), \mathsf{ct}_{\mu_i}, \mathsf{ct}_{r_{u,i}}, \mathsf{hpk}_i, \mathsf{hsk}_i) \quad \text{in } \mathsf{Hybrid}_3$$
$$\text{and} \quad (\mathsf{FHE.Eval}(\mathsf{hpk}_i, \mathcal{C}^0, (\mathsf{ct}'_{s,i}, \star)), \mathsf{ct}_{\mu_i}, \mathsf{ct}_{r_{u,i}}, \mathsf{hpk}_i, \mathsf{hsk}_i) \quad \text{in } \mathsf{Hybrid}_4$$

are indistinguishable due to malicious circuit privacy of FHE. □

The theorem statement follows from Lemma 2.

Lemma 2. *Assume that* Sch *satisfies* UF-CMA *security. Then the advantage of the adversary in the one more unforgeability game is negligible in* Hybrid_4.

Proof *(Lemma 2).* Assume there exists an adversary $\mathcal{U}^*$ that wins the one more unforgeability game in Hybrid_4. Then we build an adversary $\mathcal{B}$ against the UF-CMA security of the Schnorr signature Sch:

1. $\mathcal{B}$ obtains $\mathsf{Sch.vk}$ from the Schnorr signature challenger $\mathcal{C}$; it samples $(\mathsf{ppk}, \mathsf{psk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$, programs $H'(\alpha) = \mathsf{ppk}$ and defines $\mathsf{BS.vk} = \{\mathsf{Sch.vk}, \alpha, \mathsf{ppk}\}$ and forwards it to $\mathcal{U}^*$.

2. It runs the signing protocol with adversary $\mathcal{U}^*$ simulating the BS signer by reprogramming the random oracle as follows:
   (a) Adversary $\mathcal{U}^*$ outputs FHE public keys $\{\mathsf{hpk}_i\}_{i\in[Q]}$, ciphertexts $\{\mathsf{ct}_{\mathsf{sk}_i}\}_{i\in[Q]}$ and $\mathsf{NIZK}_{\mathsf{FHE}}$ proofs $\{\pi_{\mathsf{sk}_i}\}_{i\in[Q]}$. Then $\mathcal{B}$ extracts $\{\mathsf{hsk}_i\}_{i\in[Q]}$ as $\mathsf{PKE.Dec}(\mathsf{psk}, \mathsf{ct}_{\mathsf{sk}_i})$ for $i \in [Q]$.
   (b) $\mathcal{B}$ reprograms the random oracle (denote the assigned hash values from the Schnorr signature challenger by $H_{\mathcal{C}}$ and the reprogrammed hash values by $H_{\mathcal{B}}$). When $\mathcal{U}^*$ outputs a random oracle query $H_{\mathcal{B}}$ on (x, y) to $\mathcal{B}$, then $\mathcal{B}$ assigns a hash value for $H_{\mathcal{B}}(x, y)$ by itself, or by requesting a random oracle query $H_{\mathcal{C}}$ to the Schnorr signature challenger - if $(x, y) = (m_i, r_i \cdot r_{u,i})$ and the i th execution is already done for some $i \in [Q]$ then $\mathcal{B}$ assigns the value $H_{\mathcal{C}}(m_i, r_i)$ to $H_{\mathcal{B}}(x, y)$; else if the value $H_{\mathcal{C}}(x, y)$ is already assigned to $H_{\mathcal{B}}$ then $\mathcal{B}$ assigns a uniform and fresh value to $H_{\mathcal{B}}(x, y)$; else $\mathcal{B}$ assigns the value $H_{\mathcal{C}}(x, y)$ to $H_{\mathcal{B}}(x, y)$.
   (c) $\mathcal{U}^*$ outputs Q signing queries consisting of the ciphertexts $\mathsf{ct}_{r_{u,i}}$ and ct_{μ_i} and their proofs $\pi_{r_{u,i}}, \pi_{\mu_i}$ for $i \in [Q]$. Then $\mathcal{B}$ verifies the proofs and if valid, obtains $r_{u,1}, \cdots, r_{u,Q}$ and $\mu_1, \cdots, \mu_Q$ by decrypting the ciphertexts with $\mathsf{hsk}_1, \ldots, \mathsf{hsk}_Q$, respectively.
   (d) $\mathcal{B}$ sends $\mu_1, \cdots, \mu_Q$ to the Schnorr challenger $\mathcal{C}$ and obtains signatures $\sigma_1, \cdots, \sigma_Q$ where $\sigma_i = (r_i, s_i)$. $\mathcal{B}$ assigns the hash value $H_{\mathcal{C}}(\mu_i, r_i)$ to $H_{\mathcal{B}}(\mu_i, r_i \cdot r_{u,i})$, and outputs $\perp$ if the value $H_{\mathcal{C}}(\mu_i, r_i)$ is already assigned to $H_{\mathcal{B}}$.
   (e) $\mathcal{B}$ sets $r_{s,i} = r_i$ and simulates the proofs $\pi_{k_{s,i}}$ for all $i \in [Q]$. Note that for this, the challenger does not need the witness $dlog(r_{s,i})$. To generate encryptions of s_i, $\mathcal{B}$ first computes $\mathsf{ct}'_{s,i} = \mathsf{FHE.Enc}(\mathsf{hpk}_i, s_i)$ and then $\mathsf{ct}_{s,i} = \mathsf{FHE.Eval}(\mathsf{hpk}_i, \mathcal{C}^0, (\mathsf{ct}'_{s,i}, \star))$ for $i \in [Q]$ and forwards them along with $r_{s,1}, \cdots, r_{s,Q}$ and $\pi_{k_{s,1}}, \cdots, \pi_{k_{s,Q}}$ to $\mathcal{U}^*$.
   (f) $\mathcal{U}^*$ outputs $Q + 1$ message-signature pairs $(\mu_j, (r_j, s'_j))_{j\in J}$ that pass verification with respect to $H_{\mathcal{B}}$ and such that the μ_j's are distinct, where $|J| = Q + 1$.
3. $\mathcal{B}$ outputs any of these for which μ_j had not been queried to the Schnorr challenger.

Note that $s_i = \mathsf{Sch.Sign}^s(\mathsf{Sch.sk}, \mu; dlog(r_{s,i})) = dlog(r_{s,i}) + \mathsf{Sch.sk} \cdot H_{\mathcal{C}}(\mu_i, r_{s,i}) = dlog(r_{s,i}) + \mathsf{Sch.sk} \cdot H_{\mathcal{B}}(\mu_i, r_{u,i} \cdot r_{s,i}) = \mathcal{C}_{\mathsf{Sch.sk},k_{s,i}}(\mu_i, r_{u,i})$ with respect to $H_{\mathcal{B}}$. We also note that the last output message-signature pair $(\mu_j, (r_j, s'_j))$ by $\mathcal{B}$ satisfies $g^{s'_j} = r_j \cdot \mathsf{BS.vk}^{H_{\mathcal{B}}(\mu_j, r_j)} = r_j \cdot \mathsf{BS.vk}^{H_{\mathcal{C}}(\mu_j, r_j)}$ since μ_j has not been queried to the Schnorr challenger. Hence the message-signature pair output by $\mathcal{B}$ passes verification with respect to $H_{\mathcal{C}}$, which is a forgery of the Schnorr signature.

The aborting probability in 2.(d) can be bounded by considering all hash queries and all sign queries. The probability that one of the sign queries with μ_i outputs r_i such that $r_i \cdot r_{u,i}$ corresponds to one of the hash/sign queries is bounded by

$$Q_S \cdot (Q_S + Q_H + 1)/q,$$

where Q_S and Q_H are the number of sign/hash queries since r_i is uniform independent of the rest.

The advantage of the adversary $\mathcal{U}^*$ is translated to an advantage of the adversary $\mathcal{B}$. □

□

Blindness. We now argue that the FHE-based blind Schnorr signature scheme satisfies honest signer blindness. This can be upgraded to malicious signer blindness by adding $\mathsf{NIZKAoK}$ proofs to show that the public are honestly generated.

Theorem 8. *Assume that* FHE *and* PKE *are secure and correct,* $\mathsf{NIZK}_{\mathsf{ct}}$ *and* $\mathsf{NIZK}_{\mathsf{FHE}}$ *are zero knowledge and* $\mathsf{NIZKAoK}_{\mathsf{DL}}$ *is knowledge sound. Then the construction in Figure 3 satisfies honest signer blindness.*

Note that all assumptions can be conjectured to hold against a quantum adversary. In particular, note that the $\mathsf{NIZKAoK}_{\mathsf{DL}}$ is trivially sound for a quantum adversary.

Proof. The argument proceeds via the following hybrids.

Hybrid_0 : This is the genuine honest signer blindness experiment. The challenger (simulated user) and the adversary (malicious signer) S^* interacts in the following game.

1. The challenger samples $\mathsf{BS.sk} = \mathsf{Sch.sk}$, $\mathsf{BS.vk} = (\mathsf{Sch.vk}, \alpha, \mathsf{ppk})$ and sends $\mathsf{BS.sk}, \mathsf{BS.vk}$ to the signer S^*.
2. S^* outputs messages μ_0 and μ_1. The challenger picks a bit $b \leftarrow \{0,1\}$ uniformly randomly.
3. S^* interacts concurrently with users $\mathcal{U}(\mathsf{BS.vk}, \mu_b)$ and $\mathcal{U}'(\mathsf{BS.vk}, \mu_{\bar{b}})$, simulated by the challenger as follows:
   (a) $\mathcal{U}$ (resp. $\mathcal{U}'$) runs $\mathsf{FHE.KeyGen}(1^\lambda)$ to generate $(\mathsf{hsk}, \mathsf{hpk})$ (resp. $(\mathsf{hsk}', \mathsf{hpk}')$), computes $\mathsf{ct}_{\mathsf{sk}}$ (resp. $\mathsf{ct}'_{\mathsf{sk}}$) and $\mathsf{NIZK}_{\mathsf{FHE}}$ proof π_{sk} (resp. π'_{sk}) and sends $\mathsf{hpk}, \mathsf{ct}_{\mathsf{sk}}, \pi_{\mathsf{sk}}$ (resp. $\mathsf{hpk}', \mathsf{ct}'_{\mathsf{sk}}, \pi'_{\mathsf{sk}}$) to S^*.
   (b) $\mathcal{U}$ (resp. $\mathcal{U}'$) randomly chooses $k_u \leftarrow \mathbb{Z}_q^*$ and computes $r_u = g^{k_u}$ (resp. $k'_u \leftarrow \mathbb{Z}_q^*$ and computes $r'_u = g^{k'_u}$) and sends the encryptions ct_{r_u} and ct_μ along with proofs π_{r_u}, π_μ (resp. ct'_{r_u}, ct'_μ, π'_{r_u}, π'_μ) to S^*.
   (c) $\mathcal{U}$ (resp. $\mathcal{U}'$) receives r_s, π_{k_s} and ct_s (resp. r'_s, π'_{k_s} and ct'_s) from S^*.
   (d) $\mathcal{U}$ (resp. $\mathcal{U}'$) verifies the proof and aborts if it fails. Otherwise, $\mathcal{U}$ (resp. $\mathcal{U}'$) computes $r = g^{k_u} r_s$ (resp. $r' = g^{k'_u} r'_s$). Then $\mathcal{U}$ (resp. $\mathcal{U}'$) decrypts ct_s (resp. ct'_s) and obtains s (resp. s').
   (e) $\mathcal{U}$ (resp. $\mathcal{U}'$) verifies signature (r, s) (resp. (r', s')) and aborts if the check fails.
4. If the game is aborted at any stage then the challenger sends $(\perp, \perp), (\perp, \perp)$ to S^*. Otherwise, it sets $(r_b, s_b) = (r, s)$ and $(r_{\bar{b}}, s_{\bar{b}}) = (r', s')$ and sends (r_0, s_0) and (r_1, s_1) to S^*.
5. The adversary S^* outputs its guess for bit b.

Hybrid_1 and Hybrid_2 are same as in the unforgeability proof (Theorem 5).

Hybrid_3 : In this hybrid, the $\mathsf{NIZK}_{\mathsf{FHE}}$ proofs π_{sk} and π'_{sk} and $\mathsf{NIZK}_{\mathsf{ct}}$ proofs π_{r_u}, π_μ, π'_{r_u} and π'_μ are replaced with simulated proofs.

Hybrid_4: In this hybrid, the PKE ciphertexts $\mathsf{ct}_{\mathsf{sk}}$ and $\mathsf{ct}'_{\mathsf{sk}}$ encrypt $\mathbf{0}$.

Now, consider any PPT adversary (signer) S^*. We show via the following arguments that the advantage of S^* in Hybrid_0, i.e. in the real world is negligible.

Indistinguishability of Hybrids: Hybrid_0, Hybrid_1 and Hybrid_2 are indistinguishable by the same argument as that in the proof of unforgeability (Theorem 5). Hybrid_2 and Hybrid_3 are indistinguishable assuming $\mathsf{NIZK}_{\mathsf{FHE}}$ and $\mathsf{NIZK}_{\mathsf{ct}}$ are zero knowledge. Hybrid_3 and Hybrid_4 are indistinguishable from the security of PKE. We argue indistinguishability between Hybrid_2, Hybrid_3 and Hybrid_4 via the following claims.

Claim 9 *Assume* $\mathsf{NIZK}_{\mathsf{FHE}}$ *and* $\mathsf{NIZK}_{\mathsf{ct}}$ *are zero knowledge. Then* Hybrid_2 *and* Hybrid_3 *are indistinguishable.*

Proof *(Claim 9).* The only difference between Hybrid_2 and Hybrid_3 is that the $\mathsf{NIZK}_{\mathsf{FHE}}$ proofs $\pi_{\mathsf{sk}}, \pi'_{\mathsf{sk}}$ and $\mathsf{NIZK}_{\mathsf{ct}}$ proofs $\pi_{r_u}, \pi_\mu, \pi'_{r_u}, \pi'_\mu$ are real in the former and simulated in the latter. Hence indistinguishability follows from the zero-knowledge property of $\mathsf{NIZK}_{\mathsf{FHE}}$ and $\mathsf{NIZK}_{\mathsf{ct}}$, respectively. □

Claim 10 *Assume* PKE *is secure. Then* Hybrid_3 *and* Hybrid_4 *are indistinguishable.*

Proof *(Claim 10).* The two hybrids differ only in PKE ciphertexts $\mathsf{ct}_{\mathsf{sk}}$ and $\mathsf{ct}'_{\mathsf{sk}}$ which encrypt FHE keys in Hybrid_3 and $\mathbf{0}$ in Hybrid_4. Hence, the indistinguishability between the two hybrids can be shown via standard reduction to PKE security. □

Next, we show that under the assumptions in Theorem 8, advantage of S^* in Hybrid_4 is negligible.

Lemma 3. *Assume the conditions given in Theorem 8 are true. Then, the advantage of* S^* *in* Hybrid_4 *is negligible.*

Proof *(Lemma 3).* To prove the claim, we partition the set of all adversarial signers S^*, and prove the claim for each part separately. Consider the following classification of (possibly malicious) signers:

- Type_{hon} : the class of those adversaries who always perform homomorphic evaluation so that the ciphertexts ct_s and ct'_s decrypt to correct values, namely signatures that pass Schnorr verification.
- Type_{mal} : the class of adversaries who always perform homomorphic evaluation of at least one of the two sessions incorrectly, so that ct_s or ct'_s decrypt to a value that fails Schnorr verification.

– Type_{mix} : the class of those adversaries who are not in Type_{hon} or Type_{mal}. Thus, such adversaries sometimes compute both ciphertexts correctly, and sometimes compute at least one of them incorrectly depending on their own or the challenger's random coins.

Claim 11 *Assume* FHE *is secure. Then, if* S^* *is of* Type_{mal}*, it has a negligible advantage in* Hybrid_4*.*

Intuitively, if the signer S^* computes (either of) the ciphertexts ct_s (resp. ct'_s) incorrectly, then at least one of them does not decrypt to the desired value, and signature verification fails. In this case the challenger outputs $\perp$ for both the signatures. Thus, the only information that S^* can learn is via the first round messages, which contain simulated proofs, a PKE ciphertext in each session encrypting zero, and FHE ciphertexts. Hence, the claim follows by FHE security.

Proof *(Claim 11).* We observe that such a S^* receives $\mathsf{hpk}, \mathsf{ct}_{\mathsf{sk}}, \mathsf{ct}_{r_u}, \mathsf{ct}_{\mu_b}$ and simulated proofs from $\mathcal{U}$ (resp. $\mathsf{hpk}', \mathsf{ct}'_{\mathsf{sk}}, \mathsf{ct}'_{r_u}, \mathsf{ct}_{\mu_{\bar{b}}}$ and simulated proofs from $\mathcal{U}'$) in its first round of interaction with the challenger. Note that of all the values returned to S^*, all the values other than ct_{μ_b} and $\mathsf{ct}_{\mu_{\bar{b}}}$ carry no information about the challenge bit b. Also, ct_{μ_b} and $\mathsf{ct}_{\mu_{\bar{b}}}$ hide the bit b due to FHE security.

Thus, we can show that if S^* has a non negligible advantage in Hybrid_4, then there exists an adversary $\mathcal{B}$ who breaks FHE security with non negligible advantage. The reduction $\mathcal{B}$ is defined as follows:
Upon receiving FHE public key, hpk_{ch} from the FHE challenger, $\mathcal{B}$ does the following:

1. Samples $(\mathsf{BS.vk}, \mathsf{BS.sk})$ as defined for Hybrid_4, and invokes S^* with $(\mathsf{BS.vk}, \mathsf{BS.sk})$.
2. S^* outputs two messages μ_0 and μ_1.
3. $\mathcal{B}$ samples a pair of FHE keys, $(\mathsf{hpk}_{\mathcal{B}}, \mathsf{hsk}_{\mathcal{B}}) \leftarrow \mathsf{FHE.KeyGen}(1^\lambda)$ and computes $\mathsf{ct}_{\mathcal{B}} = \mathsf{FHE.Enc}(hpk_{\mathcal{B}}, \mu_0)$ and samples a bit $b \leftarrow \{0,1\}$.
4. $\mathcal{B}$ sends μ_0 and μ_1 as challenge messages to the FHE challenger. The FHE challenger returns $\mathsf{ct}_{ch} = \mathsf{FHE.Enc}(\mathsf{hpk}_{ch}, \mu_\beta)$, for $\beta \leftarrow \{0,1\}$.
5. If $b = 0$, $\mathcal{B}$ sets $\mathsf{hpk} = \mathsf{hpk}_{\mathcal{B}}$, $\mathsf{hpk}' = \mathsf{hpk}_{ch}$, $\mathsf{ct}_\mu = \mathsf{ct}_{\mathcal{B}}$ and $\mathsf{ct}'_\mu = \mathsf{ct}_{ch}$. Else, if $b = 1$, $\mathcal{B}$ sets $\mathsf{hpk}' = \mathsf{hpk}_{\mathcal{B}}$, $\mathsf{hpk} = \mathsf{hpk}_{ch}$, $\mathsf{ct}'_\mu = \mathsf{ct}_{\mathcal{B}}$ and $\mathsf{ct}_\mu = \mathsf{ct}_{ch}$.
6. $\mathcal{B}$ computes $\mathsf{ct}_{r_u}, \mathsf{ct}'_{r_u}, \mathsf{ct}_{\mathsf{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathbf{0})$, $\mathsf{ct}'_{\mathsf{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathbf{0})$ and simulated proofs $\pi_{\mathsf{sk}}, \pi'_{\mathsf{sk}}, \pi_\mu, \pi'_\mu, \pi_{r_u}, \pi_{r'_u}$.
7. $\mathcal{B}$ simulates $\mathcal{U}$ with $\mathsf{hpk}, \mathsf{ct}_{\mathsf{sk}}, \mathsf{ct}_\mu, \mathsf{ct}_{r_u}, \pi_{\mathsf{sk}}, \pi_{r_u}, \pi_\mu$ and $\mathcal{U}'$ with $\mathsf{hpk}', \mathsf{ct}'_{\mathsf{sk}}, \mathsf{ct}'_\mu, \mathsf{ct}'_{r_u}, \pi'_{\mathsf{sk}}, \pi'_{r_u}, \pi'_\mu$.
8. In the end, $\mathcal{B}$ returns signatures $\perp, \perp$ to S^*.
9. S^* outputs its guess bit b'.
10. If $b' = b$, $\mathcal{B}$ returns 1, else 0 to the FHE challenger.

Analysis of $\mathcal{B}$*'s advantage:* Observe that if $\beta = 1$, then $\mathcal{B}$ simulated Hybrid_4 with S^* with challenge bit b. On the other hand, if $\beta = 0$, then both $\mathcal{U}$ and $\mathcal{U}'$ are simulated with μ_0 (and its ciphertexts) only, thus hiding bit b information theoretically. In this case S^* can do no better than making a random guess.

Hence, we can analyze the advantage of $\mathcal{B}$ as

$$\begin{aligned}\mathsf{Adv}_{\mathcal{B}} &= \Pr(\mathcal{B} \to 1 | \beta = 1) - \Pr(\mathcal{B} \to 1 | \beta = 0) \\ &= \Pr(b' = b | \beta = 1) - \Pr(b' = b | \beta = 0) \\ &= 1/2 + \epsilon - 1/2 = \epsilon.\end{aligned}$$

Lemma 4. *Assume* $\mathsf{NIZKAoK}_{\mathsf{DL}}$ *is knowledge sound and* FHE *is correct and secure. Then, if* S^* *is of* Type_{hon}*, i.e. always computes* ct_s *and* ct'_s *correctly, then its advantage in* Hybrid_4 *is negligible.*

Proof *(Lemma 4).* The proof is argued via the following hybrids.

Hybrid_5 : This is same as Hybrid_4 except that in this hybrid, the decrypted signature components, s_0, s_1 are replaced with the components that are generated by the challenger as follows:

1. The challenger extracts the witnesses k_s^{ext} and $k_s'^{\,\mathsf{ext}}$ from the proofs π_{k_s} and π'_{k_s} using $\mathsf{NIZKAoK}_{\mathsf{DL}}$ extractor. Using these, it computes $k^{\mathsf{ext}} = k_u + k_s^{\mathsf{ext}}$, $r_s^{\mathsf{ext}} = g^{k_s^{\mathsf{ext}}}$ and $r^{\mathsf{ext}} = g^{k^{\mathsf{ext}}}$. Similarly it computes $k'^{\,\mathsf{ext}} = k'_u + k_s'^{\,\mathsf{ext}}$, $r_s'^{\,\mathsf{ext}} = g^{k_s'^{\,\mathsf{ext}}}$ and $r'^{\,\mathsf{ext}} = g^{k'^{\,\mathsf{ext}}}$.

2. The challenger checks whether the randomnesses satisfy $r_s = r_s^{\mathsf{ext}}$ and $r'_s = r'_s{}^{\mathsf{ext}}$. If not, then it aborts. Else, it computes $r = g^{k_u} r_s$, $r' = g^{k'_u} r'_s$. Then it computes $s^{\mathsf{ext}} = k^{\mathsf{ext}} + \mathsf{BS.sk} \cdot H(\mu_b, r)$, $s'^{\mathsf{ext}} = k'^{\mathsf{ext}} + \mathsf{BS.sk} \cdot H(\mu_{\bar{b}}, r')$, sets $(r_b, s_b) = (r, s^{\mathsf{ext}})$, $(r_{\bar{b}}, s_{\bar{b}}) = (r', s'^{\mathsf{ext}})$ and returns the signatures as (r_0, s_0), (r_1, s_1) to S^*.

Hybrid_6 : In this hybrid, all the FHE encryptions (i.e., $\mathsf{ct}_{r_u}, \mathsf{ct}_\mu, \mathsf{ct}'_{r_u}, \mathsf{ct}'_\mu$) are replaced with encryptions of zero.

We first show that for such an adversary, Hybrid_4 and Hybrid_5 are indistinguishable.

Claim 12 *Assume the conditions in Lemma 4 hold. Then* Hybrid_4 *and* Hybrid_5 *are indistinguishable.*

Proof *(Claim 12).* Observe that if the proofs π_{k_s} and π'_{k_s} do not verify then the game aborts in both the hybrids. On the other hand, if the proofs π_{k_s} and π'_{k_s} verify, then the computation of s_b and $s_{\bar{b}}$ in Hybrid_5 is further conditioned on the successful extraction of k_s and k'_s after which they are computed directly without having to perform any decryption. In more detail, if the proofs π_{k_s} and π'_{k_s} verify, then the challenger does not abort at this stage in Hybrid_4 by definition. In Hybrid_5, from the knowledge soundness of $\mathsf{NIZKAoK}_{\mathsf{DL}}$, the challenger succeeds in extracting the witnesses k_s^{ext} and $k'_s{}^{\mathsf{ext}}$ with overwhelming probability and hence does not abort at this stage.

Next we show that the signatures generated in both hybrids are identical. Since r_b and $r_{\bar{b}}$ are computed in the same way in both the hybrids, we focus on s_b and $s_{\bar{b}}$. Here we give the argument to prove that s_b computed in Hybrid_5 is same as that computed in Hybrid_4. The same argument applies to $s_{\bar{b}}$.

In Hybrid_4, from the correctness of $\mathsf{FHE.Eval}$ and the assumption that S^* computes ct_s and ct'_s honestly, s_b obtained by computing $\mathsf{FHE.Dec}(\mathsf{FHE.sk}, \mathsf{ct}_s)$ passes and Sch verification and the challenger does not abort. From $\mathsf{Sch.Verify}$, we have $g^{s_b} = r_b \cdot \mathsf{BS.vk}^{H(\mu_b, r_b)}$. From the uniqueness of discrete log, we get that: given $r_b, \mathsf{BS.vk}$ and μ_b, s_b is uniquely determined and must be equal to $k_u + k_s + \mathsf{BS.sk} \cdot H(\mu_b, r_b)$, where $g^{k_s} = r_s$.

In Hybrid_5, since discrete log is unique, $k_s^{\mathsf{ext}} = k_s$. Hence, $s_b^{\mathsf{ext}} = k_u + k_s^{\mathsf{ext}} + \mathsf{BS.sk} \cdot H(\mu_b, r_b) = k_u + k_s + \mathsf{BS.sk} \cdot H(\mu_b, r_b) = s_b$ (returned in Hybrid_4). This proves the claim. □

Claim 13 *Assume* FHE *is secure. Then for any* S^* Hybrid_5 *and* Hybrid_6 *are indistinguishable.*

Proof *(Claim 13).* The two hybrids differ only in FHE ciphertexts ct_{r_u}, ct_μ, ct'_{r_u} and ct'_μ which encrypt genuine values in Hybrid_5 and $\mathbf{0}$ in Hybrid_6. Hence, the indistinguishability between the two hybrids can be argued via standard reduction to FHE security. □

We note that in Hybrid_6, the challenge bit b is information theoretically hidden. Hence the advantage of the adversary in Hybrid_4 is negligible, which concludes the proof of Lemma 4. □

Lemma 5. *Assume* $\mathsf{NIZKAoK}_{\mathsf{DL}}$ *is knowledge sound and* FHE *is correct and secure. Then, if* S^* *is of* Type_{mix}*, then its advantage in* Hybrid_4 *is negligible.*

Proof *(Lemma 5).* We define Hybrid_5 and Hybrid_6 as in the proof of Lemma 4. We prove in Claim 14, that if advantage of S^* in Hybrid_4 is non-negligible, then there exists an adversary whose advantage in Hybrid_5 is non-negligible. But since, we have shown that Hybrid_5 and Hybrid_6 are indistinguishable and advantage of all adversaries in Hybrid_6 is zero, there cannot exist any adversary with non-negligible advantage in Hybrid_5. This proves that advantage of S^* in Hybrid_4 cannot be non-negligible.

Claim 14 *Assume the conditions in Lemma 5 hold. Then, if* S^* *is of* Type_{mix} *and has non-negligible advantage in* Hybrid_4*, then there exists a PPT adversary (signer) who has non-negligible advantage in* Hybrid_5.

Proof *(Claim 14).* First, we make following observations:

1. Conditioned on the event that S^* computes ct_s and ct'_s correctly, Hybrid_4 and Hybrid_5 are indistinguishable. (Proof is same as that of Claim 12).
2. Conditioned on the event that S^* computes at least one of the two ciphertexts - ct_s and ct'_s - incorrectly, its advantage in Hybrid_4 is negligible (Proof is same as that of Claim 11).

Now we analyze the overall advantage of S^* in the two hybrids. Let $\mathsf{Corr\text{-}ct}$ be defined as the event that S^* computes ct_s and ct'_s correctly, i.e. they decrypt to values that pass Schnorr verification. Let Adv_{i,S^*} be the advantage of S^* in Hybrid_i. Then[6],

$$\mathsf{Adv}_{4,S^*} = (\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) + (\mathsf{Adv}_{4,S^*}|\overline{\mathsf{Corr\text{-}ct}})\Pr(\overline{\mathsf{Corr\text{-}ct}}) \tag{4.2}$$

From Observations 1 and 2 above, we have

$$(\mathsf{Adv}_{4,S^*}|\overline{\mathsf{Corr\text{-}ct}}) = \mathrm{negl} \tag{4.3}$$
$$(\mathsf{Adv}_{5,S^*}|\mathsf{Corr\text{-}ct}) = (\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct}) + \mathrm{negl} \tag{4.4}$$

This gives us

$$\mathsf{Adv}_{4,S^*} = (\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) + \mathrm{negl} \tag{4.5}$$
$$\mathsf{Adv}_{5,S^*} = (\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) + \delta^*\Pr(\overline{\mathsf{Corr\text{-}ct}}) + \mathrm{negl} \tag{4.6}$$

where $\delta^* := (\mathsf{Adv}_{5,S^*}|\overline{\mathsf{Corr\text{-}ct}})$.

Let us now define an adversary S^{**} as follows: S^{**} behaves exactly as S^* except that in the event of $\overline{\mathsf{Corr\text{-}ct}}$, it flips the bit b' returned by S^*. Then,

$$\begin{aligned}(\mathsf{Adv}_{5,S^{**}}|\mathsf{Corr\text{-}ct}) &= (\mathsf{Adv}_{5,S^*}|\mathsf{Corr\text{-}ct}),\\ (\mathsf{Adv}_{5,S^{**}}|\overline{\mathsf{Corr\text{-}ct}}) &= -(\mathsf{Adv}_{5,S^*}|\overline{\mathsf{Corr\text{-}ct}})\end{aligned} \tag{4.7}$$

$$\begin{aligned}\mathsf{Adv}_{5,S^{**}} &= (\mathsf{Adv}_{5,S^{**}}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) + (\mathsf{Adv}_{5,S^{**}}|\overline{\mathsf{Corr\text{-}ct}})\Pr(\overline{\mathsf{Corr\text{-}ct}})\\ &= (\mathsf{Adv}_{5,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) - (\mathsf{Adv}_{5,S^*}|\overline{\mathsf{Corr\text{-}ct}})\Pr(\overline{\mathsf{Corr\text{-}ct}})\\ &= (\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) - \delta^*\Pr(\overline{\mathsf{Corr\text{-}ct}}) + \mathrm{negl}\end{aligned} \tag{4.8}$$

If $\mathsf{Sign}(\delta^*) = \mathsf{Sign}((\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct}))$, then

$$\begin{aligned}|\mathsf{Adv}_{5,S^*}| &= |(\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) + \delta^*\Pr(\overline{\mathsf{Corr\text{-}ct}}) + \mathrm{negl}\,|\\ &\geq |(\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) + \mathrm{negl}\,|\\ &= |\mathsf{Adv}_{4,S^*}|\end{aligned} \tag{4.9}$$

If $\mathsf{Sign}(\delta^*) = -\mathsf{Sign}((\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct}))$, then

$$\begin{aligned}|\mathsf{Adv}_{5,S^{**}}| &= |(\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) - \delta^*\Pr(\overline{\mathsf{Corr\text{-}ct}}) + \mathrm{negl}\,|\\ &\geq |(\mathsf{Adv}_{4,S^*}|\mathsf{Corr\text{-}ct})\Pr(\mathsf{Corr\text{-}ct}) + \mathrm{negl}\,|\\ &= |\mathsf{Adv}_{4,S^*}|\end{aligned} \tag{4.10}$$

Thus, either S^* or S^{**} has absolute advantage at least as good as the absolute advantage of S^* in Hybrid_4. This proves that if S^* has non-negligible advantage in Hybrid_4, then there exists an adversary with non-negligible advantage in Hybrid_5, which in turn would imply an adversary with non-negligible advantage in Hybrid_6 – a contradiction. *(Claim 14)* □

(Lemma 5) □

(Lemma 3) □

Thus, from Lemma 3, we conclude that no PPT adversary can have non-negligible advantage in Hybrid_0. This concludes the proof of Theorem 8. □

[6] for events A and B, we let $\Pr(\mathsf{A}|\mathsf{B}) = 1$, if $\Pr(\mathsf{B}) = 0$

5 Instantiating Components for Blind Schnorr Signature

Next, we provide an instantiation of the HE-based blind Schnorr signature over an elliptic curve group $G \subset E(\mathbb{F}_{2^d})$ over a binary field. We choose to use this variant for two efficiency reasons: (i) the field size is small compared to the finite field variants and (ii) the binary field with characteristic 2 is useful for the hash and the following process. Further, we use the division-free variant binary EC-Scnorr signature [NSS04] to detour a costly finite field division.

5.1 Underlying Building Blocks

In this section, we provide the underlying building blocks for our blind Schnorr signature. We define $\mathcal{R} \coloneqq \mathbb{Z}[x]/(\Phi_m(x))$, where $\Phi_m(x)$ is the m th cyclotomic polynomial for an odd positive integer m, and denote $\mathcal{R}_N \coloneqq \mathcal{R}/N\mathcal{R}$ for a positive integer N. We denote by $\mathbb{F}_{p^r}$ a finite field of order p^r with a prime characteristic p.

BGV FHE and Packing Methods. Consider the BGV scheme $\mathsf{FHE.Enc} : \mathcal{M} \times \mathcal{K} \to \mathcal{C}$ for plaintext space $\mathcal{M} = \mathcal{R}_2$, key space $\mathcal{K}$ and ciphertext space $\mathcal{C} = \mathcal{R}_Q^2$. The ciphertext modulus Q is one of the positive integers $Q_0 < Q_1 < \cdots < Q_L$ depending on the level of the ciphertext, which decreases over the computations from L to 0. The bootstrapping restores the level of the ciphertext from L_{low} to $L_{\text{high}} > L_{\text{low}}$ and enables further homomorphic multiplications. The secret key $\mathsf{hsk} = (\mathsf{s}, 1)$ is chosen from $\mathcal{R}^2$ with ternary coefficients.

The plaintext space can be decomposed into multiple slots by the Chinese Remainder Theorem: $\mathcal{R}_2 \cong \prod_{i=1}^{\ell} \mathbb{Z}_2[x]/(f_i(x))$, where $f_1, \cdots, f_\ell$ s are the d_0-degree distinct irreducible polynomial factors of $\Phi_m(x)$ in $\mathbb{Z}_2[x]$. Each component $\mathbb{Z}_2[x]/(f_i)$ is isomorphic to a finite field $\mathbb{F}_{2^{d_0}}$ and is called a slot. Packing methods in BGV make it possible to encrypt ℓ elements of $\mathbb{F}_{2^{d_0}}$ as a single ciphertext, using multiple plaintext slots.

Elliptic Curve Schnorr Signature. We consider the Schnorr signature with a group defined on an binary elliptic curve. More concretely, let E be an elliptic curve over $\mathbb{F}_{2^d}$, and consider a subgroup $G \subset E(\mathbb{F}_{2^d})$ of prime order q and a generator $P \in G$. We denote $x[R]$, $y[R]$ the x, y-coordinates of the EC point R in affine coordinates and denote $[k]P$ the EC point addition of k copies of the point P, i.e. $P + P + \cdots + P$. We omit $[R]$ if there is no confusion. Let H be a hash function. Then the EC-Schnorr signature of a message μ under a secret key $\mathsf{sk} \in \mathbb{Z}_q$ is

$$\sigma = (R,\ s = k + H(\mu, x[R], y[R]) \cdot \mathsf{sk}),$$

where $R = [k]P$ for some $k \in \mathbb{Z}_q^*$. Since R is uniquely determined by the x-coordinate and one bit, so $y[R]$ can be compressed to the bit. This point compression is used when output a final blind signature, but not in the signing protocol executions. For a verification key $\mathsf{vk} = Q = [\mathsf{sk}]P$ and a signature (R, s) for the message m, the verification algorithm will check whether $[s]P = R + [H(\mu,\ x[R], y[R])]\, Q$.

Division-Free Elliptic Curve Schnorr Signature. We will use the division-free variant of the EC-Schnorr signature with the randomized projective coordinates [NSS04] over a group defined on a binary elliptic curve. Division-free indicates that there is no finite field division, when computing the first element in the signature. More concretely, let E be an elliptic curve over $\mathbb{F}_{2^d}$, and consider a subgroup $G \subset E(\mathbb{F}_{2^d})$ of prime order q and a generator $P \in G$. We denote by $[k]P$ the EC point addition of k copies of the point P, i.e. $P + P + \cdots + P$. Let H be a hash function. Note that the point R can be represented in two forms; affine coordinates (x, y), or projective coordinates $[X;\ Y;\ Z]$ where $x = X/Z$ and $y = Y/Z$, and $x,\ y,\ X,\ Y,\ Z \in \mathbb{F}_{2^d}$. Then the division-free EC-Schnorr signature of a message μ under a secret key $\mathsf{sk} \in \mathbb{Z}_q$ is

$$\sigma = (X,\ Y,\ Z,\ s = k + H(\mu,\ X,\ Y,\ Z) \cdot \mathsf{sk}) \in (\mathbb{F}_{2^d})^3 \times\ \mathbb{Z}_q,$$

where the coordinates are of the point $R = [k]P$ for some $k \in \mathbb{Z}_q^*$. For security, the coordinates $X,\ Y,\ Z$ have to be randomized with a random $\epsilon \leftarrow \mathbb{F}_{2^d}$ as ϵX, ϵY, ϵZ. For a verification key $\mathsf{vk} = Q = [\mathsf{sk}]P$ and a signature $(X,\ Y,\ Z, s)$ for the message m, the verification algorithm will check whether $R = (X/Z, Y/Z) \in G$

and whether $[s]P = R + [H(\mu, X, Y, Z)]Q$. Since R is uniquely determined by the x-coordinate and one bit, so $y[R]$ can be compressed to this bit. This point compression is used when outputting a final blind signature, but not in the signing protocol executions. The resulting signature size will be 6λ-bit for λ bit security[7]. In our case, we will have $k = k_u + k_s \in \mathbb{Z}_q$ and $R = R_u + [k_s]P \in G$, and signing will be homomorphically evaluated with encrypted R_u (with projective coordinates) and μ; and unencrypted k_s, sk and P.

Edwards Curve. We select Edward curves to benefit from "complete addition", i.e., addition which does not depend on whether the summands are infinity points. In contrast, elliptic curves have a "conditional" addition algorithm in general which depends on the summands, and necessitate adding an "if" statement in homomorphic evaluation, impacting efficiency.

5.2 Parameter Selection

We explicit our choice of the parameters for components for homomorphic signing.

- **FHE Parameters.** Currently, the only available library for bootstrappable BGV is HElib [HHS+21, HS21]. We take parameters as follows: $m = 42799 = 127 \cdot 337$, $\phi(m) = 42336 = 21 \cdot 2016$ and the largest modulus Q_L is 946 bits which gives 127.6-bit security (as estimated by the HElib library). With this choice, we get 2016 slots isomorphic to $\mathbb{F}_{2^{d_0}}$ with $d_0 = 21$.
- **Elliptic Curve.** We choose to use a binary Edwards curve E defined over $\mathbb{F}_{2^d}$ for $d = 273 = 21 \cdot 13$, which gives $d/2 = 136.5$-bit security for the base Schnorr signature[8]. We look for a cardinality q that is a prime in $[2^d + 1 - 2^{d/2+1}, 2^d + 1 + 2^{d/2+1}]$, and we aim at minimizing its Hamming weight. Among the two primes of Hamming weight 3 in the interval, we choose $q = 2^{273} + 2^{55} + 1$. Then we can find an elliptic curve E using the CM method, whose heuristic analysis in [BS07] shows that we can find it with fixed order q in time polynomial in d.
- **SHA-3 Hash Function.** For the hash evaluation, we choose the standard SHA-3 hash function and analyze it precisely for performance[9]. For a maximum 1086-bit input, consisting of three 273-bit projective coordinates of an elliptic curve point and a 267-bit message, it outputs a 256-bit hash value. The inner KECCAK-p function has a 1600-bit input with extra 2-bit suffix and a $c = 512$-bit capacity. Then its bit security is 128 and the number of rounds is $n_r = 24$.

5.3 Instantiating the Blind Signature Protocol

The resulting blind signature scheme $\mathsf{BS}^{\mathsf{DFSch}}$ based on the construction in Figure 3 and on division-free Schnorr has the following differences:

- In Setup, the setup for the division-free Schnorr DFSch outputs a group $G = \langle P \rangle$ with respect to a base point P on an Edwards curve $E(\mathbb{F}_{2^d})$.
- In Signing,
  1. $\mathcal{U}$ computes $(x_u, y_u) = R_u = [k_u]P$ for a randomly chosen k_u. $\mathcal{U}$ encrypts x_u, y_u and μ to ct_{x_u}, ct_{y_u} and ct_μ and sends these with the proofs.
  2. $\mathcal{S}$ computes the projective coordinates X_s, Y_s and Z_s such that $[X_s; Y_s; Z_s] = R_s = [k_s]P$ for a randomly chosen k_s and randomizes these by multiplying with $\epsilon \in \mathbb{F}_{2^d}$ for a randomly chosen ϵ. $\mathcal{S}$ generates the proof of the discrete log of $R_s = (X_s/Z_s, Y_s/Z_s)$. It computes $(\mathsf{ct}_X, \mathsf{ct}_Y, \mathsf{ct}_Z) \leftarrow \mathsf{FHE.Eval}(\mathsf{EC.Add}_{X_s,Y_s,Z_s}, (\mathsf{ct}_{x_u}, \mathsf{ct}_{y_u}))$ and computes $\mathsf{ct}_s = \mathsf{FHE.Eval}(\mathcal{C}^{\mathsf{DFSch}}_{\mathsf{BS.sk},k_s}, (\mathsf{ct}_\mu, \mathsf{ct}_X, \mathsf{ct}_Y, \mathsf{ct}_Z))$. $\mathcal{S}$ sends X_s, Y_s, Z_s and its proof and ct_s.

[7] In case of Schnorr signature with division, the signature size will be 4λ-bit (or 3λ-bit when using shorter hash) for λ bit security.

[8] There is no attack on the ECC parameter we choose, but there may be some more possibilities (or concern) than other standardized parameters. However, due to the lack of the flexibility in choosing FHE parameters, it is not easy to use the standardized ECC parameters for now.

[9] We need our hash function to be modeled as a random oracle, for which SHA-3 is suitable (see for instance [ABRB+19]).

3. $\mathcal{U}$ checks whether $(X_s/Z_s, Y_s/Z_s) \in G$ and verifies the proof. $\mathcal{U}$ computes $(X, Y, Z) = \mathsf{EC.Add}_{X_s,Y_s,Z_s}(x_u, y_u)$ and $s = \mathsf{FHE.Dec}(\mathsf{hsk}, \mathsf{ct}_s)$. $\mathcal{U}$ verifies the signature $\sigma = (X, Y, Z, k_u + s)$.

- In Verification, it is just a division-free Schnorr verification.

The circuit $\mathsf{EC.Add}_{X_s,Y_s,Z_s}$ is for the Edwards curve point addition with mixed representations [BLRF08]. With the components X_s, Y_s, Z_s hard-wired, for inputs x_u, y_u it outputs a deterministic projective coordinates of a Edwards curve point $(X_s/Z_s, Y_s/Z_s) + (x_u, y_u)$ if the inputs are valid repesentations of Edwards curve points. The circuit $\mathcal{C}^{\mathsf{DFSch}}_{\mathsf{BS.sk},k_s}$, the division-free Schnorr signing with $\mathsf{BS.sk}, k_s$ hardwired, takes inputs μ, X, Y, Z and outputs $k_s + H(\mu, X, Y, Z) \cdot \mathsf{BS.sk} \mod q$. As the division-free Schnorr signature, the Y-coordinate can be omitted in the blind signature as $\sigma' = (X, Y_0, Z, k_u + s)$, where Y_0 is a bit for the choice of Y. We here give a full description of $\mathsf{BS}^{\mathsf{DFSch}}$ in Figure 4.

Setup. $\mathsf{Gen}(1^\lambda)$: Upon input the security parameter λ, do the following:

- Invoke division-free Schnorr $\mathsf{DFSch.sk}$ Setup, resulting a group $G = \langle P \rangle$ on an Edwards curve $E(\mathbb{F}_{2^d})$.
- Invoke $(\mathsf{DFSch.sk}, \mathsf{DFSch.vk}) \leftarrow \mathsf{DFSch.KeyGen}(1^\lambda)$.
- Set $\mathsf{ppk} = H'(\alpha)$, where $\alpha \in \{0,1\}^\lambda$ is any arbitrary publicly known value (for e.g. $\alpha = 1^\lambda$).
- Output $\mathsf{BS.sk} = \mathsf{DFSch.sk}$, $\mathsf{BS.vk} = \{\mathsf{DFSch.vk}, \alpha, \mathsf{ppk}\}$.

Signing. $\langle \mathcal{S}(\mathsf{BS.sk}), \mathcal{U}(\mathsf{BS.vk}, \mu) \rangle$:

1. **User:** Given the key $\mathsf{BS.vk}$ and a message μ, user $\mathcal{U}$ first verifies if $\mathsf{ppk} = H'(\alpha)$ and continues only if the verification passes. It then does the following:
   - It first samples $\mathsf{FHE.KeyGen}$ randomness ρ and an FHE key-pair $(\mathsf{hpk}, \mathsf{hsk}) = \mathsf{FHE.KeyGen}(1^\lambda; \rho)$
   - It then samples a $\mathsf{PKE.Enc}$ randomness r_e and computes $\mathsf{ct}_{\mathsf{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{hsk}; r_e)$.
   - It then generates a $\mathsf{NIZK}_{\mathsf{FHE}}$ proof π_{sk} for the following statement:
     Given $\mathsf{pk}, \mathsf{ppk}, \mathsf{ct}_{\mathsf{sk}}$, there exists a tuple $(\mathsf{hsk}, \rho, r_e)$ such that
     $$(\mathsf{hpk}, \mathsf{hsk}) = \mathsf{FHE.KeyGen}(1^\lambda; \rho) \wedge \mathsf{ct}_{\mathsf{sk}} = \mathsf{PKE.Enc}(\mathsf{ppk}, \mathsf{hsk}; r_e). \tag{5.1}$$
   - It randomly chooses $k_u \in \mathbb{Z}_q$, computes $(x_u, y_u) = R_u = [k_u]P$.
   - It encrypts x_u, y_u and μ to ct_{x_u}, ct_{y_u} and ct_μ with the proofs on the well-formedness of the ciphertexts.
   - It sends $\mathsf{hpk}, \mathsf{ct}_{\mathsf{sk}}$ with it's proof and ciphertexts with the proofs to signer $\mathcal{S}$.
2. **Signer:** Upon receiving $(\mathsf{hpk}, \mathsf{ct}_{\mathsf{sk}}, \pi_{\mathsf{sk}}, \mathsf{ct}_{x_u}, \mathsf{ct}_{y_u}, \mathsf{ct}_\mu, \pi_{x_u}, \pi_{y_u}, \pi_\mu)$, signer $\mathcal{S}$ does the following:
   - It verifies π_{sk}, π_{x_u}, π_{y_u} and π_μ and outputs $\perp$ if any of these fails.
   - It samples a partial randomness $k_s \leftarrow \mathbb{Z}_q$ and computes the projective coordinate X_s, Y_s and Z_s such that $[X_s; Y_s; Z_s] = R_s = g^{k_s}$ and randomize by multiplying $\epsilon \leftarrow \mathbb{F}_{2^d}$. It generates a $\mathsf{NIZKAoK}_{\mathsf{DL}}$ proof π_{k_s} that it knows the discrete log of $R_s = (X_s/Z_s, Y_s/Z_s)$.
   - It computes $(\mathsf{ct}_X, \mathsf{ct}_Y, \mathsf{ct}_Z) \leftarrow \mathsf{FHE.Eval}(\mathsf{EC.Add}_{X_s,Y_s,Z_s}, (\mathsf{ct}_{x_u}, \mathsf{ct}_{y_u})))$ and computes $\mathsf{ct}_s = \mathsf{FHE.Eval}(\mathcal{C}^{\mathsf{DFSch}}_{\mathsf{BS.sk},k_s}, (\mathsf{ct}_\mu, \mathsf{ct}_X, \mathsf{ct}_Y, \mathsf{ct}_Z))$.
   - It sends X_s, Y_s, Z_s with the proof and ct_s to user $\mathcal{U}$.
3. **User:** Upon receiving ct_s, X_s, Y_s, Z_s, π_{k_s}, user $\mathcal{U}$ does the following:
   - It first verifies π_{k_s} and outputs $\perp$ if it fails.
   - It verifies $(X_s/Z_s, Y_s/Z_s) \in G$ and outputs $\perp$ if it fails.
   - It computes $(X, Y, Z) = \mathsf{EC.Add}_{X_s,Y_s,Z_s}(x_u, y_u)$ and $s = \mathsf{FHE.Dec}(\mathsf{hsk}, \mathsf{ct}_s)$.
   - It verifies the DFSch signature $\sigma = (X, Y, Z, k_u + s)$ and outputs $\perp$ if it fails.
   - It outputs σ.

Verifying. The verifier computes $\mathsf{DFSch.Verify}(\mathsf{BS.vk}, \mu, \sigma)$ and outputs the result.

Fig. 4 $\mathsf{BS}^{\mathsf{DFSch}}$: HE-based Round-Optimal Blind Division-free Schnorr Signature.

We now instantiate the key components for the blind signing.

5.3.1 Instantiating Large Finite Field Operations with BGV The FHE scheme BGV with plaintext space $\mathcal{R}_2$ naturally supports finite field operations on $\mathbb{F}_{2^{d_0}}$. However, bootstrapping for BGV parameter with large $d_0 \geq 256$ is not supported by HElib library, because of its inefficiency [HS21]. For efficiency reasons, hence we could not use $d_0 = d$ for the elliptic curve base field $\mathbb{F}_{2^d}$.

In order to handle the finite field operations in the large extension field, we use a field isomorphism $\mathbb{F}_{2^{d_0}}[x]/(F(x)) \cong \mathbb{F}_{2^d}$ with $d = nd_0$ for $F(x) \in \mathbb{F}_{2^{d_0}}[x]$ of degree n. Using the isomorphism, we can represent the large finite field elements $a, b \in \mathbb{F}_{2^d}$ with elements in the small finite field $\mathbb{F}_{2^{d_0}}$ as $a = a_0 + a_1 x + \cdots + a_{n-1}x^{n-1}$ and $b = b_0 + b_1 x + \cdots + b_{n-1}x^{n-1}$, where $a_i, b_i \in \mathbb{F}_{2^{d_0}}$ for $0 \leq i \leq n-1$.

We encode $a(x) = \sum_{i=0}^{n-1} a_i x^i \in \mathbb{F}_{2^d}$ as a vector $\vec{a} = (a_0, \cdots, a_{n-1}) \in (\mathbb{F}_{2^{d_0}})^n$. The finite field addition of $a, b \in \mathbb{F}_{2^d}$ can be seen as an addition of two vectors. However, the finite field multiplication is more complicated. When calculating $a \times b = c \in \mathbb{F}_{2^d}$ with polynomial representations, we have $\sum_{i=0}^{n-1} c_i x^i = (\sum_{i=0}^{n-1} a_i x^i)(\sum_{i=0}^{n-1} b_i x^i) \mod F(x)$. Letting d_i be the i th coefficient of the two polynomials a and b without mod $F(x)$, and $x^{j+n-1} \mod F(x) = \sum_{i=0}^{n-1} M_{i,j} x^i$ for $1 \leq j \leq n-1$, we can compute the coefficients for c with the constant matrix $\mathbf{M} = (M_{i,j}) \in (\mathbb{F}_{2^{d_0}})^{n \times (n-1)}$ as $(c_0, c_1, \cdots, c_{n-1})^t = (d_0, d_1, \cdots, d_{n-1})^t + \mathbf{M} \cdot (d_n, d_{n+1}, \cdots, d_{2n-2})^t$.

5.3.2 Instantiating Zero Knowledge Proofs We describe how to efficiently instantiate the proof systems required for our protocol.

- $\mathsf{NIZK}_{\mathsf{FHE}}$ **proof for the FHE key pairs.** The FHE key pairs are basically ciphertexts with respect to the secret key hsk. The user needs to generate FHE keys and proofs on the key pairs, once before the execution starts. This will be included in the communication cost, however, in the scenario that multiple blind signatures are issued to a single user, the key pairs can be delivered in the off-line phase.
  The public key hpk consists of an encryption key enck, rotation keys rotk_i for i in an index set I, a recryption key reck for bootstrapping and a relinearlization key rlk. The proof we require is for the following statement: *for given an* FHE *public key* $\mathsf{hpk} = (\mathsf{enck}, (\mathsf{rotk}_i)_{i \in I}, \mathsf{reck}, \mathsf{rlk}) \in R_Q^{2(|I|+3)}$, *a* PKE *public key* $\mathsf{ppk} = (p_0, p_1)$, *a* PKE *ciphertext* $\mathsf{ct}_{\mathsf{sk}} = (c_0, c_1)$ *and an integer* Δ, *there exists a ternary secret* $\mathsf{s} \in \mathcal{R}$, *errors* e, $e_{\mathsf{rotk},i}$, e_{reck}, e_{rlk}, e_1, $e_2 \in \mathcal{R}$ *with small norms, and* $r \in \mathcal{R}$ *such that*

$$\begin{aligned} \mathsf{enck}_1 &= -\mathsf{enck}_0 \cdot \mathsf{s} + 2e, & \mathsf{rlk}_1 &= -\mathsf{rlk}_0 \cdot \mathsf{s} + \mathsf{s}^2 + 2e_{\mathsf{rlk}}, \\ \mathsf{rotk}_{i,1} &= -\mathsf{rotk}_{i,0} \cdot \mathsf{s} + \mathsf{s}(x^i) + 2e_{\mathsf{rotk},i}, & c_0 &= r \cdot p_0 + e_1, \\ \mathsf{reck}_1 &= -\mathsf{reck}_0 \cdot \mathsf{s} + \mathsf{s} + 2e_{\mathsf{reck}}, & c_1 &= r \cdot p_1 + \Delta \mathsf{s} + e_2, \end{aligned}$$

  for each $i \in I$, *in* $\mathcal{R}_Q$. Since the vector representations of $\mathsf{s}(x^i)$ can be obtained with linear transformations on $\vec{\mathsf{s}}$, we can rewrite the equations as $A\vec{u} = \vec{v}$, where the vector $\vec{u}$ is a concatenation of the vectors for s, the error terms and a message for rlk and the vector $\vec{v}$ is a concatenation of the vectors for second components of the keys and the components of $\mathsf{ct}_{\mathsf{sk}}$. The proof on the quadratic relation that rlk indeed encrypts s^2 is unnecessary, if the signer adds $\rho \cdot \mathsf{FHE.Eval}(\mathsf{hpk}, C(x) = x^2, \mathsf{reck})$ to ct_s at the end of the signer-side signing, for a randomly chosen $\rho \leftarrow \mathcal{R}$.

  On the unnecessity of the proof on the quadratic relation for rlk. The proof on the relation $\mu_{\mathsf{rlk}} = s^2$ cannot be directly proven using the tools in the literature, where μ_{rlk} is a message included in rlk when regarding it as a ciphertext. We instead introduce a technique to randomize the signer's output ciphertext if the rlk is not honestly generated. The signer adds $\rho \cdot \mathsf{ct}$ where $\mathsf{ct} \coloneqq \rho \cdot \mathsf{FHE.Eval}(\mathsf{hpk}, C(x) = x^2, \mathsf{reck})$ to ct_s at last, for a random $\rho \leftarrow \mathcal{R}$, then if the rlk is honestly generated then the result is $\mathsf{ct}_s + \mathsf{FHE.Enc}(0)$. Otherwise, let $\mathsf{reck}_0\mathsf{s} + \mathsf{reck}_1 = \mathsf{s} + 2e_{\mathsf{reck}} \in \mathcal{R}_Q$ and $\mathsf{rlk}_0 \cdot \mathsf{s} + \mathsf{rlk}_1 = \mathsf{s}^2 + t + 2e_t \in \mathcal{R}_Q$. Then $\mathsf{ct}_0 \cdot \mathsf{s} + \mathsf{ct}_1 = (\mathsf{reck}_0^2 - 1)t + 2((\mathsf{reck}_0^1 - 1)e_t + (\mathsf{reck}_0 + 1)\mathsf{s}e_{\mathsf{reck}} + \mathsf{reck}_1 e_{\mathsf{reck}}) \in \mathcal{R}_Q$. With the fact that the first components of the public keys are given with a PRNG seed, the first component and the message is randomized when adding $\rho \cdot \mathsf{ct}$ if $t \neq 0$, $\mathsf{ct}_0 \neq 0$ and $\mathsf{reck}_0^2 - 1 \neq 0$. The latter two conditions can be checked by the signer and if so the signer aborts. The error term will be randomized using the noise flooding technique in Section 5.3.3. Then the $\mathsf{NIZK}_{\mathsf{FHE}}$ proof can be efficiently proven using the exact lattice proofs [ENS20, LNS21] based on BDLOP commitment scheme.

- $\mathsf{NIZK}_{\mathsf{ct}}$ **proof for the** FHE **ciphertexts.** The user needs to prove that the fresh ciphertexts ct are indeed valid ciphertexts. More concretely, the user should generate a proof for the following statement: *for given* $\mathsf{ct} = (c_0, c_1) \in \mathcal{R}_Q^2$ *and* $\mathsf{enck} = (\mathsf{enck}_0, \mathsf{enck}_1) \in \mathcal{R}_Q^2$*, there exists* $\mu \in \mathcal{R}$ *with binary coefficients,* $e_1, e_2 \in \mathcal{R}$ *with small norms and* $r \in \mathcal{R}$*, such that* $(c_0, c_1) = (r \cdot \mathsf{enck}_0 + e_1, r \cdot \mathsf{enck}_1 + \mu + 2e_2)$. We can prove this similarly to the above $\mathsf{NIZK}_{\mathsf{FHE}}$, but with much smaller A and Q.

- $\mathsf{NIZKAoK}_{\mathsf{DL}}$ **proof for the knowledge of a discrete log.** The signer is required to add a $\mathsf{NIZKAoK}_{\mathsf{DL}}$ when sending X_s, Y_s, Z_s. It is a proof on that *for given* $G = \langle P \rangle$ *of order* q *and* $R = (X_s/Z_s, Y_s/Z_s) \in G$*, there exists* $k \in \mathbb{Z}_q$*, such that* $[k]P = R$. This can be proven with a Schnorr signature with a verification key R and a secret key k, and the proof is $\pi = (R' = [k']P, s' = k' + H(R', R) \cdot k)$, where $k' \leftarrow \mathbb{Z}_q^*$ [CS97].

5.3.3 Malicious Circuit Private FHE For semi-honest circuit privacy, there are two techniques in the literature that can be applied to BGV, the noise flooding of the ciphertext error term and the iterative bootstrappings [DS16]. Since the iterative bootstrapping technique requires about λ bootstrappings, we rather use the noise flooding technique. Recall that the BGV ciphertext $\mathsf{ct} = (c_0, c_1) \in \mathcal{R}_Q$ for a plaintext message $m(x) \in \mathcal{R}_2$ satisfies $\langle \mathsf{ct}, \mathsf{sk} \rangle = m(x) + 2e(x) \mod Q$ for the secret key sk and an error $e(x) \in \mathcal{R}$. If the coefficients of e are in the interval $[-B, B]$, then we can add the ciphertext ct by an encryption of zero with an error whose coefficients are uniformly randomly selected from the interval $[-2^\lambda B, 2^\lambda B]$. Then the statistical distance between the distribution of the resulting error and the uniform distribution is $2^{-\lambda}$, which gives the semi-honest circuit privacy with the security parameter λ. For this, the error bound B should have bit-size less than $log_2 Q - \lambda - 1$ for a correct decryption. Concretely, we use $B \leq log_2 Q - \lambda - 4$ for the following modulus switching as described in Section 5.3.4. This can be obtained when the *bit capacity* of the ciphertext is larger than $\lambda + 4$, when Q is larger than 370 bit, heuristically.

We can then modify BGV scheme to achieve a malicious circuit privacy in the following way as in [AKSY21]. First, for the FHE public keys the first elements should be generated using a hash function. In the random oracle model we can ensure that they are uniformly distributed. There also should be $\mathsf{NIZKAoK}$ proofs in the public key that the second elements are well-formed. Second, before the FHE computations are performed, each ciphertext needs to be bootstrapped. This can make every ciphertext in possibly malicious form into a properly formed ciphertext as in [AGM21].

Using ROM does not require any signing/communication cost and as a matter of fact, we can use a PRNG seed for a proper instantiation for the first components. Also, we already have a proof on the well-formedness of the public key pairs for the unforgeability of blind signature. The ciphertext modulus size could be remain small with the modulus switching technique at last. Hence, to make BGV maliciously circuit private, it requires only one more extra bootstrapping with an encryption and an HE addition for the noise flooding.

5.3.4 Modulus Switching, Down to the Limit We use modulus switching [BGV12, HS20] to reduce the signer-to-user communication cost. Recall that modulus switching converts a ciphertext with a large modulus Q into a ciphertext with a smaller modulus Q', such that correctness of decryption and the FHE operations are maintained. We use this technique but want to make the modulus Q' as small as possible, even smaller than the lowest level modulus of BGV, while maintaining only the correctness of decryption. For $\mathsf{ct} = (c_0, c_1) \in \mathcal{R}_Q$, we define $a_i = \lceil Q'/Q \cdot c_i \rfloor \in \mathcal{R}$, $b_i = Q'c_i - Qa_i \in \mathcal{R}$, and $d_i \in \mathcal{R}$ as $Qd_i = b_i \mod 2$ for $i = 0, 1$. Then the coefficients of the b_i are in $[-Q/2, Q/2]$ and the coefficients of d_i's can be chosen in $\{0, \pm 1\}$ with same sign with the corresponding coefficients in b_i. Then we can get the resulting ciphertext $\mathsf{ct}' = (c_0' = a_0 + d_0, c_1' = a_1 + d_1)$ which gives the following modular equation $(c_0 + c_1\mathsf{s} \mod Q) = (c_0' + c_1'\mathsf{s} \mod Q')$. The ciphertext ct' decrypts to the same message if and only if $\|Q'e/Q + (d_0 - b_0/Q) + (d_1 - b_1/Q)\mathsf{s}\|_{\mathsf{can}} \leq Q'/2D_m$, where $D_m \approx 2$ and $\|d_i - b_i/Q\|_{\mathsf{can}} \leq 1.5$. Recall that the resulting ciphertext from the previous Section 5.3.3 has $\|e\|_{\mathsf{can}} < Q/4$ and $\|(d_0 - b_0/Q) + (d_1 - b_1/Q)\mathsf{s}\|_{\mathsf{can}} < 2^8$ for s with hamming weight 120. Hence we can choose $Q' \approx 2^{11}$, and the resulting ciphertext size will be ≈ 116 KB.

5.4 Two Variants of $\mathsf{BS}^{\mathsf{Sch}}$

We also give two optional variants of our Blind Schnorr Signature to reduce the communication cost or the signing time. The first variant uses a symmetric cipher such as AES-256. The second variant uses a $\mathbb{Z}_p$-linear map as a hash function. This reduces the hash evaluation time, but its security is proven only in the GGM [CLMQ21].

5.4.1 Variant Using AES To lower the user-to-signer communication cost, we can use a symmetric block cipher which also maintains the conjectured quantum blindness. Typically, we can use AES. The user encrypts with AES and then sends it to the signer. The signer can homomorphically decrypt it if it has an FHE encryption of the AES symmetric keys. This procedure is very similar to the homomorphic evaluation of AES circuit in [GHS12].

To lower the large communication cost due to the BGV ciphertext, we rather send AES encryptions of the messages μ and k_u and then convert it to BGV ciphertexts. This conversion is indeed an homomorphic evaluation of AES decryption with homomorphically encrypted AES keys, and can be seen as a *recryption*. When the AES keys are fixed, the AES encryption and the decryption functions are bijections from $\{0,1\}^{128}$ to $\{0,1\}^{128}$ and are inverse to each other. Hence the NIZK proofs for the BGV ciphertexts ct_μ and ct_{k_u} can be replaced by the NIZK proofs for the BGV ciphertexts of the AES keys. In such scenario, the semantic security of FHE and AES ensures the indistinguishability between $(\mathsf{ct}_{\mathsf{AES.sk}} = \mathsf{FHE.Enc}_{\mathsf{hpk}}(\mathsf{AES.sk}), \mathsf{AES}_{\mathsf{AES.sk}}(\mu))$ and $(\mathsf{ct}_0 = \mathsf{FHE.Enc}_{\mathsf{hpk}}(0), \mathsf{rand} \leftarrow \{0,1\}^{128})$ and the soundness of the proofs guarantee the well-formedness of the (possibly malicious) $\mathsf{ct}_{\mathsf{AES.sk}}$. Note that the ciphertexts and the proofs can be given to the signer before the executions start, i.e., during the off-line phase.

We can parallelize the decryption circuit by using the CBC mode. In more details, the ith ciphertext ct_i can be decrypted as $\mathsf{AES\text{-}256\text{-}CBC.Dec}(\mathsf{sk}_i, \mathsf{ct}_i) = \mathsf{AES.Dec}(\mathsf{sk}_i, \mathsf{ct}_i) \oplus \mathsf{ct}_{i-1}$ so the AES.Dec computation can be parallelized using multiple slots, and the XOR operation can be applied after they are decrypted. Finally, we note that choosing 256-bit long AES keys is conjectured to provide 128-bit quantum security.

5.4.2 Variant Using $\mathbb{Z}_q$-Linear Hash in GGM To bypass the costly SHA-3 hash evaluation we introduce a variant of our blind Schnorr signature from FHE, using linear hash function. We recall that from [CLMQ21], when assuming the GGM model, Schnorr signature is secure even with a $\mathbb{Z}_q$-linear hash function. This could be one option to accelerate the homomorphic signing procedure.

Acknowledgments. We thank Sébastien Canard, Ilaria Chillotti, Léo Ducas and Adeline Roux-Langlois for insightful discussions. This work was partly supported by the DST "Swarnajayanti" fellowship, an IndoFrench CEFIPRA project, National Blockchain Project, the CCD Centre of Excellence, European Union Horizon 2020 Research and Innovation Program Grant 780701, BPI-France in the context of the national project RISQ (P141580), and the ANR AMIRAL project (ANR-21-ASTR-0016). Part of the research corresponding to this work was conducted while the first and last two authors were visiting the Simons Institute for the Theory of Computing.

References

ABB20a. Nabil Alkeilani Alkadri, Rachid El Bansarkhani, and Johannes Buchmann. BLAZE: practical lattice-based blind signatures for privacy-preserving applications. In *Financial Crypto*, 2020.

ABB20b. Nabil Alkeilani Alkadri, Rachid El Bansarkhani, and Johannes Buchmann. On lattice-based interactive protocols: An approach with less or no aborts. In *ACISP*, 2020.

Abe01. Masayuki Abe. A secure three-move blind signature scheme for polynomially many signatures. In *EUROCRYPT*, 2001.

ABRB+19. José Bacelar Almeida, Cécile Baritel-Ruet, Manuel Barbosa, Gilles Barthe, François Dupressoir, Benjamin Grégoire, Vincent Laporte, Tiago Oliveira, Alley Stoughton, and Pierre-Yves Strub. Machine-checked proofs for cryptographic standards: Indifferentiability of sponge and secure high-assurance implementations of sha-3. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 1607–1622, New York, NY, USA, 2019. Association for Computing Machinery.

AGM21. Shweta Agrawal, Shafi Goldwasser, and Saleet Mossel. Deniable fully homomorphic encryption from learning with errors. In *CRYPTO*, 2021.

AKSY21. Shweta Agrawal, Elena Kirshanova, Damien Stehlé, and Anshu Yadav. Can round-optimal lattice-based blind signatures be practical? *IACR ePrint Arch.*, 2021.

BCC+09. Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, and Hovav Shacham. Randomizable proofs and delegatable anonymous credentials. In *CRYPTO*, 2009.

BECE+20. Samuel Bouaziz-Ermann, Sébastien Canard, Gautier Eberhart, Guillaume Kaim, Adeline Roux-Langlois, and Jacques Traoré. Lattice-based (partially) blind signature without restart. *IACR ePrint Arch.*, 2020.

BGSS17. Olivier Blazy, Philippe Gaborit, Julien Schrek, and Nicolas Sendrier. A code-based blind signature. In *ISIT*, 2017.

BGV12. Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *ITCS*, 2012.

BLL+21. Fabrice Benhamouda, Tancrède Lepoint, Julian Loss, Michele Orrù, and Mariana Raykova. On the (in)security of ROS. In *EUROCRYPT*, 2021.

BLRF08. Daniel J Bernstein, Tanja Lange, and Reza Rezaeian Farashahi. Binary edwards curves. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 244–265. Springer, 2008.

BLS01. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *ASIACRYPT*, 2001.

BLS19. Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In *CRYPTO*, 2019.

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. *J. Cryptol.*, 2003.

Bol03. Alexandra Boldyreva. Threshold signatures, multisignatures and blind signatures based on the gap-Diffie-Hellman-group signature scheme. In *PKC*, 2003.

Bra93. Stefan Brands. Untraceable off-line cash in wallet with observers. In *CRYPTO*, 1993.

BS07. Reinier Bröker and Peter Stevenhagen. Efficient CM-constructions of elliptic curves over finite fields. *Math. Comput.*, 2007.

CGGI16. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *ASIACRYPT*, 2016.

CGGI17. Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. Faster packed homomorphic operations and efficient circuit bootstrapping for TFHE. In *ASIACRYPT*, 2017.

Cha82. David Chaum. Blind signatures for untraceable payments. In *CRYPTO*, 1982.

CL01. Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT*, 2001.

CLMQ21. Yilei Chen, Alex Lombardi, Fermi Ma, and Willy Quach. Does Fiat-Shamir require a cryptographic hash function? In *EUROCRYPT*, 2021.

CLOT21. Ilaria Chillotti, Damien Ligier, Jean-Baptiste Orfila, and Samuel Tap. Improved programmable bootstrapping with larger precision and efficient arithmetic circuits for tfhe. In Mehdi Tibouchi and Huaxiong Wang, editors, *Advances in Cryptology – ASIACRYPT 2021*, pages 670–699, Cham, 2021. Springer International Publishing.

CLPX18. Hao Chen, Kim Laine, Rachel Player, and Yuhou Xia. High-precision arithmetic in homomorphic encryption. In Nigel P. Smart, editor, *Topics in Cryptology – CT-RSA 2018*, pages 116–136, Cham, 2018. Springer International Publishing.

CP92. David Chaum and Torben Pryds Pedersen. Wallet databases with observers. In *CRYPTO*, 1992.

CS97. Jan Camenisch and Markus Stadler. Proof systems for general statements about discrete logarithms. *Technical Report/ETH Zurich, Dept. of Computer Science*, 1997.

dPK22. Rafael del Pino and Shuichi Katsumata. A new framework for more efficient round-optimal lattice-based (partially) blind signature via trapdoor sampling. In *Crypto*, 2022.

DS16. Léo Ducas and Damien Stehlé. Sanitization of FHE ciphertexts. In *EUROCRYPT*, 2016.

ENS20. Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In *ASIACRYPT*, 2020.

FHKS16. Georg Fuchsbauer, Christian Hanser, Chethan Kamath, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model from weaker assumptions. In *SCN*, 2016.

FHS15. Georg Fuchsbauer, Christian Hanser, and Daniel Slamanig. Practical round-optimal blind signatures in the standard model. In *CRYPTO*, 2015.

Fis06. Marc Fischlin. Round-optimal composable blind signatures in the common reference string model. In *CRYPTO*, 2006.

FPS20. Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind schnorr signatures and signed elgamal encryption in the algebraic group model. In *EUROCRYPT*, pages 63–95. Springer, 2020.

FS10. Marc Fischlin and Dominique Schröder. On the impossibility of three-move blind signature schemes. In *EUROCRYPT*, 2010.

GG14. Sanjam Garg and Divya Gupta. Efficient round optimal blind signatures. In *EUROCRYPT*, 2014.

GHS12. Craig Gentry, Shai Halevi, and Nigel P Smart. Homomorphic evaluation of the AES circuit. In *CRYPTO*, 2012.

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, 2008.

GRS+11. Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round optimal blind signatures. In *CRYPTO*, 2011.

HHS+21. Shai Halevi, Hamish Hunt, Victor Shoup, Oliver Masters, Flavio Bergamaschi, Jack Crawford, Fabian Boemer, et al. Helib (version 2.2.1), October 2021. Available at https://github.com/homenc/HElib.

HKKL07. Carmit Hazay, Jonathan Katz, Chiu-Yuen Koo, and Yehuda Lindell. Concurrently-secure blind signatures without random oracles or setup assumptions. In *TCC*, 2007.

HKL19. Eduard Hauck, Eike Kiltz, and Julian Loss. A modular treatment of blind signatures from identification schemes. In *EUROCRYPT*, 2019.

HKLN20. Eduard Hauck, Eike Kiltz, Julian Loss, and Ngoc Khanh Nguyen. Lattice-based blind signatures, revisited. In *CRYPTO*, 2020.

HS20. Shai Halevi and Victor Shoup. Design and implementation of HElib: a homomorphic encryption library. *IACR ePrint Arch.*, 2020.

HS21. Shai Halevi and Victor Shoup. Bootstrapping for HElib. *J. Cryptol.*, 34(1):1–44, 2021.

IKSA03. Subariah Ibrahim, Maznah Kamat, Mazleena Salleh, and Sh.R. Abdul Aziz. Secure E-voting with blind signature. In *NCTT*, 2003.

JLO97. Ari Juels, Michael Luby, and Rafail Ostrovsky. Security of blind digital signatures (extended abstract). In *CRYPTO*, 1997.

KLR21. Jonathan Katz, Julian Loss, and Michael Rosenberg. Boosting the security of blind signature schemes. In *ASIACRYPT*, 2021.

KLX20. Julia Kastner, Julian Loss, and Jiayu Xu. On pairing-free blind signature schemes in the algebraic group model. *IACR ePrint Arch.*, 2020.

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 *EUROCRYPT*, 2021.

LLK+22. Yongwoo Lee, Joon-Woo Lee, Young-Sik Kim, Yongjune Kim, Jong-Seon No, and HyungChul Kang. High-precision bootstrapping for approximate homomorphic encryption by error variance minimization. Springer-Verlag, 2022.

LLKN22. Joon-Woo Lee, Eunsang Lee, Young-Sik Kim, and Jong-Seon No. Hierarchical galois key management systems for privacy preserving aiaas with homomorphic encryption. *Cryptology ePrint Archive*, 2022.

LNP22. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Efficient lattice-based blind signatures via gaussian one-time signatures. In *PKC*, 2022.

LNS21. Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Shorter lattice-based zero-knowledge proofs via one-time commitments. In *PKC*, 2021.

LSK+19. Huy Quoc Le, Willy Susilo, Thanh Xuan Khuc, Minh Kim Bui, and Dung Hoang Duong. A blind signature from module lattices. In *DSC*, 2019.

NSS04. David Naccache, Nigel P. Smart, and Jacques Stern. Projective coordinates leak. In Christian Cachin and Jan L. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, pages 257–267, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

OA03. Miyako Ohkubo and Masayuki Abe. Security of some three-move blind signature schemes reconsidered. In *SCIS*, 2003.

Oka92. Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *CRYPTO*, 1992.

Pas11. Rafael Pass. Limits of provable security from standard assumptions. In *STOC*, 2011.

PHVBS19. Dimitrios Papachristoudis, Dimitrios Hristu-Varsakelis, Foteini Baldimtsi, and George Stephanides. Leakage-resilient lattice-based partially blind signatures. *IET Information Security*, 2019.

PS97. David Pointcheval and Jacques Stern. New blind signatures equivalent to factorization. In *CCS*, 1997.

PS00. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *J. Cryptol.*, 2000.

PSM17. Albrecht Petzoldt, Alan Szepieniec, and Mohamed Saied Emam Mohamed. A practical multivariate blind signature scheme. In *Financial Crypto*, 2017.

Rüc10. Markus Rückert. Lattice-based blind signatures. In *ASIACRYPT*, 2010.

Sch89. Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO*, 1989.

TZ22. Stefano Tessaro and Chenzhi Zhu. Short pairing-free blind signatures with exponential security. *IACR ePrint Arch.*, 2022.

Wat12. Brent Waters. Functional encryption for regular languages. In *CRYPTO*, 2012.

YAZ+19. Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In *CRYPTO*, 2019.

YL19. Xun Yi and Kwok-Yan Lam. A new blind ECDSA scheme for bitcoin transaction anonymity. In *Asia-CCS*, 2019.

ZZ21. Mark Zhandry and Cong Zhang. The relationship between idealized models under computationally bounded adversaries. *IACR ePrint Arch.*, 2021.