Group Signatures without NIZK: From Lattices in the Standard Model

Shuichi Katsumata[1,2] and Shota Yamada[2]

[1] The University of Tokyo
[2]National Institute of Advanced Industrial Science and Technology (AIST)
shuichi.katsumata@it.k.u-tokyo.ac.jp, yamada-shota@aist.go.jp

April 25, 2019

Abstract

In a group signature scheme, users can anonymously sign messages on behalf of the group they belong to, yet it is possible to trace the signer when needed. Since the first proposal of lattice-based group signatures in the random oracle model by Gordon, Katz, and Vaikuntanathan (ASIACRYPT 2010), the realization of them in the standard model from lattices has attracted much research interest, however, it has remained unsolved. In this paper, we make progress on this problem by giving the first such construction. Our schemes satisfy CCA-selfless anonymity and full traceability, which are the standard security requirements for group signatures proposed by Bellare, Micciancio, and Warinschi (EUROCRYPT 2003) with a slight relaxation in the anonymity requirement suggested by Camenisch and Groth (SCN 2004). We emphasize that even with this relaxed anonymity requirement, all previous group signature constructions rely on random oracles or NIZKs, where currently NIZKs are not known to be implied from lattice-based assumptions. We propose two constructions that provide tradeoffs regarding the security assumption and efficiency:

- Our first construction is proven secure assuming the standard LWE and the SIS assumption. The sizes of the public parameters and the signatures grow linearly in the number of users in the system.
- Our second construction is proven secure assuming the standard LWE and the subexponential hardness of the SIS problem. The sizes of the public parameters and the signatures are independent of the number of users in the system.

Technically, we obtain the above schemes by combining a secret key encryption scheme with additional properties and a special type of attribute-based signature (ABS) scheme, thus bypassing the utilization of NIZKs. More specifically, we introduce the notion of *indexed* ABS, which is a relaxation of standard ABS. The above two schemes are obtained by instantiating the indexed ABS with different constructions. One is a direct construction we propose and the other is based on previous work.

1 Introduction

1.1 Background

Group signatures, originally proposed by Chaum and van Heyst [Cv91], allow members of a group to sign on behalf of the group while guaranteeing the properties of authenticity, anonymity, and

traceability. The signatures do not reveal the particular identity of the group member who issued it, however, should the need arise, a special entity called the group manager can trace the signature back to the signer using some secret information, thus holding the group members accountable for their signatures. Due to the appealing properties group signatures offer, they have proven to be useful in many real-life applications including privacy-protecting mechanisms, anonymous online communication, e-commerce systems, and trusted hardware attestation such as Intel's SGX.

Since their introduction, numerous constructions of group signatures have been proposed with different flavors: in the random oracle model [BBS04, CL04, GKV10] or standard model [BMW03, BW06, Gro07], supporting static groups [BMW03] or dynamic groups [BSZ05, BCC+16], and constructions based on various number theoretical assumptions such as strong RSA [ACJT00, CL02], paring-based [BW06, Gro07], and lattice-based [GKV10, LLLS13]. Despite the vast amount of research concerning group signatures, in essence all constructions follow the *encrypt-then-prove* paradigm presented by Bellare, Micciancio, and Warinschi [BMW03]. To sign on a message, a group member encrypts its certificate provided by the group manager and then proves in (non-interactive) zero-knowledge of the fact that the ciphertext is an encryption of a valid certificate while also binding the message to the zero-knowledge proof.

Thus far, all group signature schemes have relied on non-interactive zero-knowledge (NIZK) proofs in the proving stage of the encrypt-then-prove paradigm. Since NIZKs for general languages are implied from (certified doubly enhanced) trapdoor permutations [FLS90, BY93] and from bilinear maps [GOS06, GS08], group signatures in the standard model are known to exist from factoring-based and pairing-based assumptions [BMW03, BW06, BW07, Gro07]. In contrast, constructions of lattice-based group signatures in the standard model have shown to be considerably difficult. Since the first lattice-based group signature in the random oracle model (ROM) proposed by Gordon et al. [GKV10], there has been a rich line of subsequent works [LLLS13, NZZ15, LNW15, LLM+16a, LLNW16, LNWX18, PLS18], however, all schemes are only provably secure in the ROM. This situation stems from the notorious fact that lattices are ill-fit with NIZKs. Although more than a decade has passed since the emergence of lattices, there is still only one construction of NIZK known in the standard model [PV08], where the language supported by [PV08] seems unsuitable to devise group signatures. Notably, the open problem of constructing lattice-based group signatures in the standard model, which has explicitly been stated in Laguillaumie et al. [LLLS13] for example, has not made any progress in the past decade or so. Taking prior works on group signatures into consideration, it seems we would require a breakthrough result for lattice-based NIZKs or to come up with a different approach than the encrypt-then-prove paradigm to obtain a lattice-based group signature in the standard model.

1.2 Our Contribution

In this paper, we make progress on this problem and give the first construction of group signatures from lattices in the standard model. Our main result can be stated informally as follows:

Theorem 1 (Informal)**.** *Under the hardness of the LWE and SIS problems with polynomial approximation factors*[1]*, there exists a group signature scheme with full-traceability and CCA-selfless anonymity in the standard model.*

We explain the statement in more details in the following. Here, we basically adopt the syntax and the security notions of the group signatures defined by Bellare, Micciancio, and Warin-

[1] By LWE and SIS problems with polynomial approximation factors, we mean they are problems which are as hard as certain worst case lattice problems with polynomial approximation factor.

schi [BMW03], which are presumably one of the most widely accepted definitions. Our construction satisfies the standard notion of full-traceability, which asserts that an adversary cannot forge a valid signature that can be opened to an uncorrupted user or that cannot be traced to anyone. As for anonymity, our construction satisfies CCA-selfless anonymity introduced by Camenisch and Groth [CG05]. The notion of CCA-selfless anonymity is a relaxation of CCA-full anonymity defined by Bellare et al. [BMW03]. Informally, full-anonymity requires that the adversary cannot distinguish signatures from two different members even if *all the signing keys of the members of the system are exposed* and it has access to an open oracle. On the other hand, CCA-selfless anonymity requires anonymity to hold only when *the signing keys of the two members in question are not exposed* and it has access to an open oracle. While the latter definition is weaker, as discussed by Camenisch and Groth [CG05], it is sufficient for some natural situations. For example, consider a situation where an adversary can adaptively corrupt users while the parties cannot erase the data. In this setting, the former security notion does not buy any more security than the latter. We emphasize that even with this relaxed security notion, no group signature from lattices is known in the standard model prior to our work. In particular, regardless of what the security notion we consider for anonymity, all prior lattice-based constructions required random oracles.

One potential drawback of the above construction may be that it has rather large public parameters and signatures, whose sizes grow linearly in the number of users in the system. A natural question would be whether we can make these sizes independent of the number of users. As a side contribution, we answer this question affirmatively under a stronger assumption:

Theorem 2 (Informal)**.** *Under the hardness of the LWE problem with polynomial approximation factors and the subexponential hardness of the SIS problem with polynomial approximation factors, there exists a group signature scheme with full-traceability and CCA-selfless anonymity whose sizes of the public parameters and signatures are independent of the number of users.*

These results are obtained by a generic construction of group signatures from one-time signatures (OTS), secret key encryptions (SKE), and a new primitive which we call *indexed attribute-based signatures* (indexed ABS). We require the standard notion of strong unforgeability for the OTS and it can be instantiated by any existing schemes such as [Moh11]. For the SKE, we require some special properties. Specifically, we require the SKE to be anonymous in addition to standard notions of hiding the message. We also require the SKE to have a decryption circuit with logarithmic depth and the property which we call key-robustness. Intuitively speaking, the key-robustness requires that the ciphertext spaces corresponding to two random secret keys to be disjoint with all but negligible probability. Such an SKE with special properties can be instantiated from the standard LWE assumption. The indexed ABS is a relaxation of the standard notion of ABS, where the setup and key generation algorithms take additional inputs. We require it to satisfy the security notion that we call co-selective unforgeability and (perfect) privacy. We show two ways of instantiating the indexed ABS. As for the first instantiation, we provide a construction of an indexed ABS that is proven to have the required security properties under the standard hardness of the SIS assumption. This instantiation leads us to Theorem 1. As for the second instantiation, we view the constrained signature scheme by Tsabary [Tsa17] as an indexed ABS scheme. Using this we obtain Theorem 2. We note that unlike our first instantiation, since the constrained signature scheme in [Tsa17] does not offer sufficient security properties for our purpose, we need to utilize complexity leveraging that incurs a subexponential reduction loss to when constructing our group signature.

1.3 Overview of Our Technique

Preprocessing NIZKs. The starting point of our work is the recent breakthrough result of *preprocessing* NIZK for **NP** from lattices in the standard model by Kim and Wu [KW18]. In a preprocessing NIZK [DMP88], a trusted third party generates a proving key k_P and a verification key k_V independently of the statement to be proven and provides k_P to the prover and k_V to the verifier. The prover can construct proofs using k_P and the verifier can validate the proofs using k_V. Preprocessing NIZKs can be seen as a general form of NIZKs; if both k_P and k_V need not be secret, then it corresponds to NIZKs in the common reference string (CRS) model; if k_P can be public but k_V needs to be secret, then it corresponds to designated verifier NIZKs [PsV06, DFN06]. The lattice-based preprocessing NIZK of Kim and Wu [KW18] can be viewed as a *designated prover* NIZK (DP-NIZK), where the proving key k_P needs to be kept secret but the verification key k_V can be made public.[2] Here, the zero-knowledge property of DP-NIZKs crucially relies on the fact that the verifier does not know the proving key k_P.

At first glance, DP-NIZKs seem to be all that we require to construct group signatures. The trusted group manager provides the user a (secret) proving key k_P on time of joining the group and publicly publishes the verification key k_V. This meets the criteria of DP-NIZKs since k_P will be kept secret by the group members and the proofs (i.e., signatures) can be publicly verified. Therefore, one might be tempted to substitute NIZKs in the CRS model with lattice-based DP-NIZKs to obtain a lattice-based group signature in the standard model. Unfortunately, this naive approach is trivially insecure. Specifically, the anonymity will be broken the moment a single group member becomes corrupt. If the group manager provides the same proving key k_P to the group members, then in case any of the group members become corrupt, k_P will be in the hands of the adversary. As we mentioned above, the zero-knowledge property of DP-NIZKs will break if the proving key k_P is known. An easy fix may be to instead provide proving keys $(\mathsf{k}_{P_i})_{i\in I}$ respectively to each group members $i \in I$ and publicly publish the corresponding verification keys $(\mathsf{k}_{V_i})_{i\in I}$. In this case, even if some of the group members become corrupt, their proving keys will not affect the zero-knowledge property of the other non-corrupt members using an independent proving key. However, the problem with this approach is that each proof constructed by a proving key k_{P_i} is implicitly associated with a unique verification key k_{V_i}. Since each verification key k_{V_i} is associated to a group member $i \in I$, the adversary can simply check which verification key accepts the proof (i.e., signature) to break anonymity. Therefore, although DP-NIZKs seem to be somewhat useful for constructing group signatures, it itself is not sufficient to be a substitute for NIZKs in the CRS model.

Viewing Attribute-Based Signatures as DP-NIZKs. The problem with the approach using DP-NIZKs is the following: if we give the same proving key k_P to every group member, then the scheme will be insecure against collusion attacks and if we give different proving keys k_{P_i} individually to each group members, then the scheme will lose anonymity. Therefore, the primitive we require for constructing group signatures is something akin to DP-NIZKs that additionally provides us with both collusion resistance and anonymity.

At this point, we would like to draw the attention to attribute-based signatures (ABS) [MPR11]. In ABS, a signer assigned with an attribute $\mathbf{y}$ is provided a signing key $\mathsf{sk}_{\mathbf{y}}$ from the authority and the signer can anonymously sign a message associated with a policy C using $\mathsf{sk}_{\mathbf{y}}$ if and only if $C(\mathbf{y}) = 1$. In addition, using the master public key mpk, anybody can verify

[2] As mentioned in Section 4 of [KW18], their scheme is only publicly verifiable when considering a slightly weaker notion of zero-knowledge than the standard notion of zero-knowledge for preprocessing NIZKs. In our work, the weaker notion suffices.

the signature regardless of who signed it. The first requirement of an ABS, which captures unforgeability, is that any collusion of signers with attributes $(\mathbf{y}_i)_{i\in I}$ cannot forge a signature on a message associated with a policy C if $C(\mathbf{y}_i) = 0$ for all $i \in I$. The second requirement, which captures anonymity, is that given a valid signature on a message associated with a policy C, the attribute $\mathbf{y}$ that was used to sign the message must remain anonymous. Namely, signatures generated by $\mathsf{sk}_{\mathbf{y}_0}$ and $\mathsf{sk}_{\mathbf{y}_1}$ are indistinguishable if $C(\mathbf{y}_0) = C(\mathbf{y}_1) = 1$. Looking at the similarity between DP-NIZKs and ABS, it is tempting to view a witness $\mathbf{w}$ as an attribute $\mathbf{y}$ and to set the proving key k_P as the ABS signing key $\mathsf{sk}_{\mathbf{w}}$. To prove that $\mathbf{w}$ is a valid witness to the statement $\mathbf{x}$, i.e., $(\mathbf{x}, \mathbf{w}) \in \mathcal{R}$ for the **NP** relation $\mathcal{R}$, the prover first prepares a circuit $C_{\mathbf{x}}(\mathbf{w}) := \mathcal{R}(\mathbf{x}, \mathbf{w})$ that has the statement $\mathbf{x}$ hard-wired to it. Then the prover signs some message associated with the policy $C_{\mathbf{x}}$ using its proving key $\mathsf{k}_P = \mathsf{sk}_{\mathbf{w}}$ and outputs the signature as the proof π. The verifier can publicly verify the proof π by checking whether or not the signature is valid. At a high level, the soundness of the proof system would follow from the unforgeability of ABS and the zero-knowledge property would follow from the anonymity of ABS. Furthermore, our initial motivation of satisfying collusion resistance and anonymity is met by the properties of ABS; even if the proving keys $(\mathsf{k}_{P_i} = \mathsf{sk}_{\mathbf{w}_i})_{i\in I}$ are compromised, it cannot be used to prove a statement $\mathbf{x}$ such that $\mathcal{R}(\mathbf{x}, \mathbf{w}_i) = 0$ for all $i \in I$ and the proofs constructed by different proving keys are indistinguishable from one another since the single mpk can be used to check the validity of all proofs (unlike the above case where unique verification keys k_{V_i} were assigned to each proving keys k_{P_i}).

Constructing Groups Signatures from ABS. While the idea of viewing ABS as some variant of DP-NIZK seems to be a great step forward, the question of how to use it to construct a group signature remains. Let us come back to the basic but powerful encrypt-then-prove paradigm of Bellare et al. [BMW03]. Recall that with this approach, the group manager issues a certificate to each group member $i \in I$ and publishes a public key for a public-key encryption scheme. To sign, a group member i encrypts its certificate as ct_i under the public key of the group manager and creates a NIZK proof of the fact that ct_i encrypts the certificate. Observe that each group member i implicitly constructs a member-specific statement $\mathbf{x}_i = \mathsf{ct}_i$ when generating the NIZK proof and sets the pair of certificate and the randomness used to create ct_i as the witness $\mathbf{w}_i$. Traceability follows since each statement $\mathbf{x}_i$ encrypts the identity of the signer and the group manager who holds the secret key can decrypt them. Anonymity of the group signature is also intact even though the statement $\mathbf{x}_i$ used by each group member is different, due to the semantic security of the underlying public-key encryption scheme. Now, let us look at the above approach through the lens of NIZK-like ABSs: The group manager issues a certificate *and an ABS signing key* $\mathsf{sk}_{\mathbf{w}_i}$ for some witness $\mathbf{w}_i$ to each group member $i \in I$, and to sign, a group member i encrypts its certificate as ct_i under the public key of the group manager and uses the ABS signing key $\mathsf{sk}_{\mathbf{w}_i}$ to *create an ABS signature for some policy* $C_{\mathbf{x}_i}$ which serves as a NIZK proof of the fact that ct_i encrypts the certificate. In order for this approach to work, the witness (i.e., attribute) embedded to the ABS signing key $\mathsf{sk}_{\mathbf{w}_i}$ must be an accepting input to the policy $C_{\mathbf{x}_i}$ which has the statement $\mathbf{x}_i = \mathsf{ct}_i$ hard-wired. Although it may be not obvious at first glance, as a matter of fact, this approach is impossible! Notably, the group manager cannot prepare in advance a witness $\mathbf{w}_i$ to a statement $\mathbf{x}_i$ that will be chosen by the group member at the time of signing. Recall that the witness $\mathbf{w}_i$ to $\mathbf{x}_i = \mathsf{ct}_i$ was the certificate *and* the randomness used to create ct_i. The group manager can embed in the ABS signing key a certificate but not the randomness since there is no way to not know what kind of randomness will be used to generate the ciphertext by the group member beforehand. Therefore, to use the ABS as a type of NIZK proof system, we must devise a mechanism for constructing statements $\mathbf{x}_i$ while keeping the witness $\mathbf{w}_i$ fixed once

and for all at the time of preparation of the ABS signing key.

This brings us to our final idea. To overcome the above problem, we embed the group member identifier $i \in I$ and a key K_i of a *secret key encryption* scheme to the ABS signing key $\mathsf{sk}_{i||\mathsf{K}_i}$. We then construct the statements $\mathbf{x}_i$ so that i and K_i can be reused as the fixed witness.[3] The following is the high-level construction of our group signature.

-GS.KeyGen: The group manager provides user $i \in I$ with a key K_i of an SKE scheme and an ABS signing key $\mathsf{sk}_{i||\mathsf{K}_i}$ where the string $i||\mathsf{K}_i$ is interpreted as an attribute.

-GS.Sign: To sign on a message M, the group member $i \in I$ prepares a ciphertext $\mathsf{ct}_i \leftarrow \mathsf{SKE.Enc}(\mathsf{K}_i, i)$, views the statement $\mathbf{x}_i$ as ct_i, and prepares a circuit $C_{\mathbf{x}_i}$ with the statement $\mathbf{x}_i$ hard-wired such that $C_{\mathbf{x}_i}(i||\mathsf{K}_i) := (i \in I) \wedge (i = \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}_i))$. Then using $\mathsf{sk}_{i||\mathsf{K}_i}$, it runs the ABS signing algorithm on message M with $C_{\mathbf{x}_i}$ as the policy. The signature is $\Sigma = (\sigma_{\mathsf{ABS}}, \mathsf{ct}_i)$.

-GS.Vrfy: To verify a signature $\Sigma = (\sigma_{\mathsf{ABS}}, \mathsf{ct})$, it prepares the circuit $C_{\mathsf{ct}}(z||y) := (z \in I) \wedge (z = \mathsf{SKE.Dec}(y, \mathsf{ct}))$ and runs the ABS verification algorithm with message M, signature σ_{ABS} and policy C_{ct}.

-GS.Open: To trace a signer from a signature $\Sigma = (\sigma_{\mathsf{ABS}}, \mathsf{ct})$, the group manager uses the secret keys $(\mathsf{K}_i)_{i \in I}$ to extract the group member identifier from the ciphertext ct.

It can be checked that the scheme is correct. If the ciphertext ct_i encrypts $i \in I$, then $\mathsf{sk}_{i||\mathsf{K}_i}$ can be used to construct a signature for the policy $C_{\mathbf{x}_i}$ where $\mathbf{x}_i = \mathsf{ct}_i$. We briefly sketch the traceability and anonymity of our group signature. First, traceability holds from the key robustness of the SKE scheme and the unforgeability of the ABS scheme. The former property states that the ciphertext space of a different set of secret keys must be disjoint. In particular, this implies that the set of statements $\mathbf{x}_i = \mathsf{ct}_i$ (i.e., languages) constructed by each group member will be disjoint. Therefore, since this also implies that the set of policies $C_{\mathbf{x}_i}$ used by each group members will be disjoint, it allows us to reduce the problem of traceability to the unforgeability of the underlying ABS scheme. We note that although key robustness may be a non-standard property to consider for SKE schemes, it is an easy property to satisfy. Second, anonymity holds from the anonymity and semantic security of the SKE scheme and the anonymity of the ABS scheme. Here, anonymity of an SKE scheme informally states that the ciphertext does not leak what secret key was used to construct it. Specifically, if there were two ciphertexts, it must be difficult to tell whether they are an encryption under the same key or two different keys. These two properties allow us to argue that the ciphertext ct_i leaks no information of the group member identity. Furthermore, the anonymity of the ABS scheme ensures that σ_{ABS} does not leak the group member identity as well. Hence the signature $\sigma = (\sigma_{\mathsf{ABS}}, \mathsf{ct}_i)$ remains anonymous.

Interestingly, our construction does not need to explicitly rely on "certificates" anymore as was done in prior constructions. This is because the signing key $\mathsf{sk}_{i||\mathsf{K}_i}$ is not only a proving key for the NIZK proof system, but also implicitly a certificate. In particular, since the ABS can be viewed as a variant of *designated prover* NIZKs, the fact that a signer was able to construct a valid signature implicitly implies that the signer was certified by the group manager. Therefore, there is no need for adding another layer of certificate to our construction as was done in previous group signature constructions. Finally, we point out in advance that our actual construction

[3] Our core idea of fixing the witness can also be realized by instead embedding $i \in I$ and a (weak) PRF seed into the ABS signing key, and using a *public key* encryption scheme. We provide detailed discussions on our choice of using SKEs in Remark 5.

in Section 4 is more complicated than the above high-level structure due to the fact that we additionally capture CCA anonymity rather than only CPA anonymity. In CCA anonymity, the adversary is further provided with an open oracle that opens (i.e., traces) a signature to a signer. Since in the security proof, the reduction algorithm will no longer hold the opening key and must simulate the open oracle on its own, extra complications are incurred compared to the CPA anonymity setting where there is no such open oracle. This situation is analogous to the difference between CPA and CCA-encryption schemes.

To the knowledgeable readers, we remark that the above idea is similar to those of Kim and Wu [KW18] for constructing DP-NIZKs. In particular, the way we embed a key of an SKE scheme, rather than the witness, to the ABS signing key is analogous to the way [KW18] embeds the key of an SKE scheme to a signature of a homomorphic signature scheme [GVW15]. Notably, both schemes crucially rely on the fact that once some private information has been embedded into an ABS signing key (resp. a homomorphic signature), the signing key (resp. signature) can be reused to generate proofs for arbitrary statements.

Constructing ABS with the Desired Properties. We now change the discussion on how to instantiate the above generic construction. Since we can instantiate SKEs through a combination of relatively standard techniques, we focus on how to instantiate ABSs from lattices in this overview. A natural way of instantiating the ABS required in our GS construction would be to use the ABS scheme proposed by Tsabary [Tsa17] proven secure under the SIS assumption, which is the only known ABS construction from lattices.[4] In their paper, two ABS schemes are proposed. The first scheme is constructed from homomorphic signatures and the second is a direct construction. We focus on the second construction here, because the anonymity notion achieved by the first scheme is not sufficient for our purpose.[5] In fact, even the latter scheme does not provide a sufficient security notion that is required for our purpose, namely, for the proof of full-traceability. While Tsabary's ABS scheme achieves selective unforgeability where the adversary is forced to declare its target policy with respect to which it will forge a signature at the beginning of the security game, we require the ABS to be unforgeable even if the adversary is allowed to *adaptively* choose its target policy. The necessity of the adaptiveness of the target policy can be seen by recalling that a forgery in the full-traceability game is of the form $\Sigma^\star = (\sigma^\star_{\mathsf{ABS}}, \mathsf{ct}^\star)$, where $\mathsf{ct}^\star$ is an adaptively chosen ciphertext that specifies the target policy $C_{\mathsf{ct}^\star}$. An easy way to resolve this discrepancy is to assume the subexponential hardness of the SIS problem and prove that Tsabary's scheme is adaptively unforgeable via complexity leveraging [BB04b]. This approach leads us to Theorem 2.

Though the above approach works, it incurs a subexponential security loss, which is not desirable. At first glance, one may think that the underlying ABS must be adaptively unforgeable to be used in our generic GS construction; an adversary can adaptively make arbitrary many key queries and signing queries, and generate a forgery depending on the answers which it gets from these queries. Unfortunately, the only known construction of a lattice-based ABS scheme in the standard model with such a strong security property is provided by complexity leveraging as described above. However, a more careful observation reveals that we do not actually require the full power of adaptive unforgeability. First, the ABS scheme does not have to support an

[4] Actually, the paper proposes constructions of constrained signature (CS), which is a slightly different primitive from ABS. However, this primitive readily implies ABS.

[5] More specifically, the first scheme only achieves a so-called weakly-hiding property, where the key attribute is not leaked from a signature, but two signatures that are signed by the same user can be linked. Translated into the setting of group signature, this allows an adversary to link two different signatures by the same user, which trivially breaks anonymity.

unbounded number of signing keys since the number of members in the group signature is fixed at setup in the static setting. Furthermore, we can relax the syntax of the ABS so that the key generation algorithm takes a user index i as an additional input, since each signing key in the group signature is associated with a user index. Finally, we can relax the unforgeability requirement of the ABS so that the adversary is forced to make all the key queries at the beginning of the security game while the target policy associated with the forgery can be chosen adaptively. We call this security notion *co-selective unforgeability*, since this is somewhat dual to the selective unforgeability notion where the key queries can be adaptive but the target policy is required to be declared at the beginning of the game.

Indeed, co-selective unforgeability is enough for instantiating our generic GS construction, because, in the construction the attributes hardwired to the signing keys of the ABS are $\{i \| \mathsf{K}_i\}$ independent from the public parameter of the ABS and can be chosen at the outset of the security game. With this observation in mind, we define a relaxed version of ABS which we call *indexed* ABS and provide a construction which does not resort to complexity leveraging.

Constructing Indexed ABS. Our starting point is the observation made by Tsabary [Tsa17], who showed that a homomorphic signature scheme can be viewed as a very weak form of an ABS scheme. In light of this observation, we can view the fully homomorphic signature scheme by Gorbunov, Vaikuntanathan, and Wichs [GVW15] as a single-user ABS scheme. In the scheme, the master public key is of the form $\mathsf{mpk} = (\mathbf{A}, \vec{\mathbf{B}} = [\mathbf{B}_1 \| \cdots \| \mathbf{B}_k])$ where $\mathbf{A}$ and $\mathbf{B}_i$ are random matrices over $\mathbb{Z}_q^{n \times m}$ and a secret key sk_x for an attribute $x \in \{0,1\}^k$ is a matrix with small entries $\vec{\mathbf{R}} = [\mathbf{R}_1 \| \cdots \| \mathbf{R}_k]$ such that $\vec{\mathbf{B}} = \mathbf{A}\vec{\mathbf{R}} + x \otimes \mathbf{G}$, where $\mathbf{G}$ is the special gadget matrix whose trapdoor is publicly known. To sign on a policy $F : \{0,1\}^k \to \{0,1\}$ and a message M, the signer uses the homomorphic evaluation algorithms [BGG+14, GV15] to compute matrices $\mathbf{R}_F$ and $\mathbf{B}_F$ such that $\mathbf{B}_F = \mathbf{A}\mathbf{R}_F + F(x)\mathbf{G}$ from sk_x, where $\mathbf{R}_F$ is a matrix with small entries and $\mathbf{B}_F$ is a publicly computable matrix. When $F(x) = 1$, the signer can compute the trapdoor for the matrix $[\mathbf{A} \| \mathbf{B}_F]$ from $\mathbf{R}_F$ using the technique of [ABB10a, MP12] and sample a short vector $\mathbf{e}_F$ from a Gaussian distribution such that $[\mathbf{A} \| \mathbf{B}_F]\mathbf{e}_F = \mathbf{0}$ using the trapdoor. The signature on (F, M) is the vector $\mathbf{e}_F$. It can be seen that $\mathbf{e}_F$ does not leak information of x, since the distribution from which it is sampled only depends on the master public key and F. Furthermore, the scheme satisfies a relaxed version of the co-selective unforgeability, where the adversary can corrupt a single user but is *not* allowed to make signing queries. To see this, let us assume that there is an adversary who chooses x at the beginning of the game and generates a forgery $\mathbf{e}_{F^\star}$ for $F^\star$ such that $F^\star(x) = 0$ given $(\mathsf{mpk}, \mathsf{sk}_x)$. Then, we can solve the SIS problem using this adversary. The reduction algorithm is given a matrix $\mathbf{A}$ as the problem instance of SIS and x from the adversary. It then sets $\vec{\mathbf{B}} = \mathbf{A}\vec{\mathbf{R}} + x \otimes \mathbf{G}$ and gives $\mathsf{sk}_x := \vec{\mathbf{R}}$ to the adversary at the beginning of the game. For the forgery $\mathbf{e}_{F^\star}$ output by the adversary, we have $[\mathbf{A} \| \mathbf{B}_{F^\star}]\mathbf{e}_{F^\star} = \mathbf{0}$. Since $\mathbf{B}_{F^\star} = \mathbf{A}\mathbf{R}_{F^\star}$, we can extract a short vector $\mathbf{z} := [\mathbf{I} \| \mathbf{R}_{F^\star}]\mathbf{e}$ such that $\mathbf{A}\mathbf{z} = \mathbf{0}$, which is a solution to the SIS problem.

There are two problems with this scheme. First, the scheme can only support a single user, whereas we need a scheme to support multiple users. It can be seen that the security of the above scheme can be broken in case the adversary obtains the keys of two different users. Second, the unforgeability of the scheme is broken once the adversary is given an access to a signing oracle. Indeed, a valid signature for a policy-message pair (F, M) is also valid for (F, M') with different $\mathsf{M}' \neq \mathsf{M}$, since the above signing and verification algorithms simply ignore the messages M. In other words, the message is not bound to the signature.

We first address the former problem. In order to accommodate multiple users in the system, we change the master public key of the scheme to be $(\mathbf{A}, \{\vec{\mathbf{B}}^{(i)}\}_{i \in [N]})$, where N is the number of

users. The secret key for a user i and an attribute $x^{(i)}$ is $\mathbf{R}^{(i)}$ such that $\vec{\mathbf{B}}^{(i)} = \mathbf{A}\vec{\mathbf{R}}^{(i)} + x^{(i)} \otimes \mathbf{G}$. To sign on a message, the user i first computes the trapdoor for $[\mathbf{A}\|\mathbf{B}_F^{(i)}]$ similarly to the above single-user construction. It then extends the trapdoor for the matrix $[\mathbf{A}\|\mathbf{B}_F^{(1)}\|\cdots\|\mathbf{B}_F^{(N)}]$ using the trapdoor extension technique [CHKP10]. Then, it samples a short vector $\mathbf{e}_F$ from a Gaussian distribution such that $[\mathbf{A}\|\mathbf{B}_F^{(1)}\|\cdots\|\mathbf{B}_F^{(N)}]\mathbf{e}_F = \mathbf{0}$. It can be observed that $\mathbf{e}_F$ does not reveal the attribute x nor the user index i since the distribution from which it is sampled only depends on the master public key and F. Note that the trapdoor extension step is essential for hiding the user index i. We can prove unforgeability for the scheme similarly to the single-user case. A key difference here is that, since there are now N matrices in the master public key, we can embed up to N user attributes $\{x^{(i)}\}_{i\in[N]}$ into the master public key as $\mathbf{B}^{(i)} = \mathbf{A}\mathbf{R}^{(i)} + x^{(i)} \otimes \mathbf{G}$.

Next, we address the latter problem. We apply the classic OR-proof technique [FLS90] and show that a scheme that is unforgeable only when the adversary cannot make signing queries can be generically converted into a scheme that is unforgeable even when the adversary can make signing queries. To do so, we introduce a dummy user that is not used in the real system. In the security proof, the signing queries are answered using the signing key of the dummy user. In order to enable this proof strategy, a naïve approach would be to change the scheme so that in order to sign on (F, M), the signer signs on a modified new policy F', which on input $x \in \{0,1\}^k$ outputs $F(x)$ and outputs 1 on input a special symbol. Then, we associate the attribute of the dummy user with the special symbol. By the privacy property of the original (no signing query) ABS, the fact that the signing queries are answered using the dummy key instead of the key specified by the adversary will be unnoticed. A problem with this approach is that since the reduction algorithm has the secret key associated with the special symbol, it can sign on *any* message and policy. Namely, any forgery output by the adversary will not be useful for the reduction algorithm since it could have constructed it on its own to begin with. To resolve this problem, we partition the space of all possible message-policy pairs into two sets, the challenge set and the controlled set, using an admissible hash [BB04a, FHPS13]. Then, we associate the dummy key with an attribute that can sign on any pair in the controlled set, but not on the challenge set. We then hope that the adversary outputs the pair that falls into the challenge set, which allows us to successfully finish the reduction. By the property of the admissible hash, this happens with noticeable probability and we can prove the security of the resulting scheme.

1.4 Related Works

Different Models of Group Signatures. One can classify group signatures in a few types of models. The largest distinctions are whether the group signature is static or dynamic and whether it achieves selfless-anonymity or full-anonymity. If the group signature allows the members of a group to dynamically change, then it is dynamic. Otherwise, it is static. Furthermore, it is called fully-anonymous when a member's signing key is exposed and it is still impossible to tell which signatures are made by the member in question. It is called selfless-anonymous when anonymity of the signature holds only when the signing key of the member in question is not exposed.

The first proper security model introduced by Bellare et al. [BMW03] defines a static group signature with full-anonymity. Later Camenisch and Groth [CG05] introduced the weaker notion of selfless-anonymity to the model of [BMW03]. Soon after, Bellare et al. [BSZ05] and Kiayias and Yung [KY06] independently extended the prior definitions to the dynamic setting. Other than these two distinctions, group signatures are enriched with many other useful properties such as verifier local revocation [BS04], message-dependent-opening [SEH+13], and opening soundness [SSE+12]. Although there are various types of models of group signatures, so far, there exists no

lattice-based group signature in the standard model for any of them.

Group Signatures from Different Assumptions. The first group signature in the standard model was proposed by Bellare et al. [BMW03]. They showed that (doubly enhanced) trapdoor permutations suffice for constructing group signatures; (doubly enhanced) trapdoor permutations is known to imply NIZKs [FLS90, BY93, Gol04]. In particular their results imply group signatures from factoring-based assumptions. From a theoretical aspect, Camenisch and Groth [CG05], showed that group signatures schemes are implied from the existence of one-way functions and NIZKs for general NP languages.[6] Since such NIZKs are implied from the existence of one-way functions in the ROM [Rom90, Nao91, PsV06], their results show that one-way functions imply group signatures in the ROM. Constructions of pairing-based group signatures in the standard model were proposed by Boyen and Waters [BW06, BW07] and Groth [Gro07], where they relied on the Groth-Sahai methodology [GS08] for designing NIZKs in the standard model for specific languages involving elements over bilinear groups. The first construction of lattice-based group signatures in the ROM from lattices were proposed by Gordon et al. [GKV10]. Subsequent works [LLLS13, NZZ15, LNW15, LLNW16, PLS18] proposed simpler and more efficient solutions. The recent work of Ling et al. [LNWX18] constructs a group signature where the parameters of the scheme are independent of the number of group members. Group signatures in the dynamic setting were proposed by Libert et al. [LLM+16b] and extended by Ling et al. [LNWX17]. Lattice-based group signatures with other types of properties such as verifier local revocation [LLNW14] and message-dependent opening [LLNW16] have also been constructed. All the lattice-based group signatures are proved secure in the ROM using the Fiat-Shamir heuristic [FS87]. Finally, Bellare and Fuchsbauer [BF14] proposed the notion of policy-based signatures and showed that policy-based signatures supporting NP-relations imply group signatures. We note that although Tsabaray [Tsa17] constructs policy-based signatures, they cannot be used to construct group signatures since their scheme only supports relations in P and not NP.

1.5 Independent Work and Open Problems

After our paper was accepted to Eurocrypt 2019, Peikert and Shiehian [PS19] posted a paper on ePrint Archive that finally closed the long-standing open problem of constructing NIZK proof system for any NP language from the LWE assumption. Their idea builds on the recent line of works that realize the Fiat-Shamir paradigm [FS87] in the standard model [CCR16, KRR17, CCRR18, HL18, CCH+19] using correlation-intractable hash functions. Notably, the result of [PS19] combined with [CG05] immediately recovers our main result, that is, the construction of selfless anonymous group signature from lattices. Moreover, [PS19] combined with [BMW03] yields a fully anonymous group signature scheme, whereas, we do not see how to realize this with our framework.

However, on the other hand, our framework enables constructions of selfless-anonymous group signatures from several assumptions that are not covered by a simple combination of [PS19] and [CG05]. For example, we obtain selfless-anonymous group signatures from (1) the LPN assumption with constant noise rate and the SIS assumption with polynomial approximation factors or (2) the SIS assumption with subexponential approximation factors. We refer to Sec. 7 for the details. The main difference between our framework and the generic approach of using NIZK system of [PS19] is that the former can avoid the potentially stronger LWE assumption whereas the latter

[6] We note that the security model of the achieved group signature is slightly weaker than that of [BMW03], i.e., it satisfies selfless-anonymity but not full-anonymity.

cannot. We leave it as an open problem to construct fully anonymous group signature solely from the SIS assumption.

2 Preliminaries

Notations. For an algorithm A that takes as input x and randomness r, "$y \in \mathsf{A}(x)$" means $\Pr_r[y' = y : y' \leftarrow \mathsf{A}(x;r)] > 0$. We denote by "$\oplus$" the (bit-wise) exclusive-or operation. For matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ that have the same number of rows, $[\mathbf{A}_1 \| \mathbf{A}_2]$ denotes their horizontal concatenation. For an integer q, we assume that an element in $\mathbb{Z}_q$ is represented by an integer in $(-q/2, q/2]$. A function $f(\cdot) : \mathbb{N} \to [0,1]$ is said to be *negligible* if for all polynomials $p(\cdot)$ and all sufficiently large $\kappa \in \mathbb{N}$, we have $f(\kappa) < 1/p(\kappa)$. The function f is *noticeable* when there exists a polynomial $p(\cdot)$ such that we have $f(\kappa) \geq |1/p(\kappa)|$ for all sufficiently large κ. Throughout the paper, we use "κ" to denote a security parameter (which is given to algorithms always in the unary form 1^κ). We denote by "$\mathrm{poly}(\cdot)$" an unspecified integer-valued positive polynomial of κ and by "$\mathrm{negl}(\kappa)$" an unspecified negligible function of κ.

2.1 Group Signature

Here, we adopt the definition of group signature schemes from the work of Bellare, Micciancio, and Warinschi [BMW03], with the relaxation regarding the anonymity suggested by Camenisch and Groth [CG05].

Syntax. Let $\{\mathcal{M}_\kappa\}_{\kappa \in \mathbb{N}}$ be a family of message spaces. In the following, we occasionally drop the subscript and simply write $\mathcal{M}$ when the meaning is clear. A group signature (GS) scheme is defined by the following algorithms:

$\mathsf{GS.KeyGen}(1^\kappa, 1^N) \to (\mathsf{gpk}, \mathsf{gok}, \{\mathsf{gsk}_i\}_{i \in [N]})$: The key generation algorithm takes as input the security parameter κ and the number of users N both in the unary form and outputs the group public key gpk, the opening key gok, and the set of user secret keys $\{\mathsf{gsk}_i\}_{i \in [N]}$.

$\mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, \mathsf{M}) \to \Sigma$: The signing algorithm takes as input the group public key gpk, the i-th user's secret key gsk_i (for some $i \in [N]$), and a message $\mathsf{M} \in \mathcal{M}_\kappa$ and outputs a signature Σ.

$\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}, \Sigma) \to \top$ or $\bot$: The verification algorithm takes as input the group public key gpk, the message M, and a signature Σ and outputs $\top$ if the signature is deemed valid and $\bot$ otherwise.

$\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}, \Sigma) \to i$ or $\bot$: The opening algorithm takes as input the group public key gpk, the opening key gok, a message M, a signature Σ and outputs an identity i or the symbol $\bot$.

For GS, we require correctness, CCA-selfless anonymity, and full traceability.

Correctness. We require that for all κ, $N \in \mathrm{poly}(\kappa)$, $(\mathsf{gpk}, \mathsf{gok}, \{\mathsf{gsk}_i\}_{i \in [N]}) \in \mathsf{GS.KeyGen}(1^\kappa, 1^N)$, $i \in [N]$, $\mathsf{M} \in \mathcal{M}_\kappa$, and $\Sigma \in \mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, \mathsf{M})$, $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}, \Sigma) = \top$ holds.

Full Traceability. We now define the full traceability for GS scheme. This security notion is defined by the following game between a challenger and an adversary A. During the game, the challenger maintains lists $\mathcal{Q}$ and $\mathcal{T}$, which are set to be empty at the beginning of the game.

Setup: At the beginning of the game, the challenger runs $\mathsf{GS.KeyGen}(1^\kappa, 1^N) \to (\mathsf{gpk}, \mathsf{gok}, \{\mathsf{gsk}_i\}_{i\in[N]})$ and gives $(1^\kappa, \mathsf{gpk}, \mathsf{gok})$ to A.

Queries: During the game, A can make the following two kinds of queries unbounded polynomially many times.

- **Corrupt Query:** Upon a query $i \in [N]$ from A, the challenger returns gsk_i to A. The challenger also adds i to $\mathcal{T}$.
- **Signing Queries:** Upon a query $(i, \mathsf{M}) \in [N]\times\mathcal{M}_\kappa$ from A, the challenger runs $\mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_i, \mathsf{M}) \to \Sigma$ and returns Σ to A. The challenger adds (i, M) to $\mathcal{Q}$.

Forgery: Eventually, A outputs $(\mathsf{M}^\star, \Sigma^\star)$ as the forgery. We say that A wins the game if:

1. $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}^\star, \Sigma^\star) \to \top$, and
2. either of the following conditions (a) or (b) is satisfied:
   (a) $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star) = \perp$,
   (b) $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star) = i^\star \notin \mathcal{T} \ \wedge \ (i^\star, \mathsf{M}^\star) \notin \mathcal{Q}$.

We define the advantage of an adversary to be the probability that the adversary A wins, where the probability is taken over the randomness of the challenger and the adversary. A GS scheme is said to satisfy full traceability if the advantage of any PPT adversary A in the above game is negligible for any $N = \mathrm{poly}(\kappa)$.

CCA-Selfless Anonymity. We now define CCA-selfless anonymity for a GS scheme. This security notion is defined by the following game between a challenger and an adversary A.

Setup: At the beginning of the game, the adversary A is given 1^κ as input and sends $i_0^\star, i_1^\star \in [N]$ to the challenger. Then the challenger runs $\mathsf{GS.KeyGen}(1^\kappa, 1^N) \to (\mathsf{gpk}, \mathsf{gok}, \{\mathsf{gsk}_i\}_{i\in[N]})$ and gives $(\mathsf{gpk}, \{\mathsf{gsk}_i\}_{i\in[N]\setminus\{i_0^\star, i_1^\star\}})$ to A.

Queries: During the game, A can make the following two kinds of queries unbounded polynomially many times.

- **Signing Queries:** Upon a query $(b, \mathsf{M}) \in \{0,1\} \times \mathcal{M}_\kappa$ from A, the challenger runs $\mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_{i_b^\star}, \mathsf{M}) \to \Sigma$ and returns Σ to A.
- **Open Queries:** Upon a query (M, Σ) from A, the challenger runs $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}, \Sigma)$ and returns the result to A.

Challenge Phase: At some point, A chooses its target message $\mathsf{M}^\star$. The challenger then samples a secret coin $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and computes $\mathsf{GS.Sign}(\mathsf{gpk}, \mathsf{gsk}_{i_{\mathsf{coin}}^\star}, \mathsf{M}^\star) \to \Sigma^\star$. Finally, it returns $\Sigma^\star$ to A.

Queries: After the challenge phase, A may continue to make signing and open queries unbounded polynomially many times. Here, we add a restriction that A cannot make an open query for $(\mathsf{M}^\star, \Sigma^\star)$.

Guess: Eventually, A outputs $\widehat{\mathsf{coin}}$ as a guess for coin.

We say that the adversary A wins the game if $\widehat{\mathsf{coin}} = \mathsf{coin}$. We define the advantage of an adversary to be $|\Pr[\mathsf{A} \text{ wins}] - 1/2|$, where the probability is taken over the randomness of the challenger and the adversary. A GS scheme is said to be CCA-selfless anonymous if the advantage of any PPT adversary A is negligible in the above game for any $N = \mathrm{poly}(\kappa)$.

Remark 1. *Note that the above security definition is slightly different from the selfless anonymity notion defined by Camenisch and Groth [CG05]. In their definition, the adversary is allowed to adaptively choose the targets and corrupt the users other than the targets. Since the number of users N is polynomially bounded, it can easily be shown that these security definitions are actually equivalent by a straightforward reduction that simply guesses $i_0^\star$ and $i_1^\star$ at the beginning of the game and aborts if the guess turns out to be incorrect. We use the above definition because it is simpler and handy.*

2.2 Secret Key Encryption and Other Primitives

We will use some cryptographic primitives such as secret key encryptions (SKE) and one-time signatures (OTS) to construct a GS scheme. The definitions of these primitives will appear in Appendix A. Since we require an SKE to have some non-standard properties, we provide a brief explanation here. We require key robustness, which intuitively says that the ciphertext spaces corresponding to two random secret keys are disjoint with all but negligible probability. In addition, we require SKE to satisfy INDr-CCA security, which stipulates that a ciphertext is indistinguishable from a pseudorandom ciphertext that is publicly samplable, even if the distinguisher is equipped with a decryption oracle.

2.3 Admissible Hash Functions

Here, we define the notion of admissible hash, which was first introduced by [BB04a]. We follow the definition of [FHPS13, BV15] with minor changes.

Definition 1. *Let $\ell := \ell(\kappa)$ and $\ell' := \ell'(\kappa)$ be some polynomials. We define the function* $\mathsf{WldCmp} : \{0,1\}^\ell \times \{0,1\}^\ell \times \{0,1\}^\ell \to \{0,1\}$ *as*

$$\mathsf{WldCmp}(y,z,w) = 0 \Leftrightarrow \forall i \in [\ell] \ \big((y_i = 0) \vee (z_i = w_i)\big)$$

where y_i, z_i, and w_i denote the i-th bit of y, z, and w respectively. Intuitively, WldCmp *is a string comparison function with wildcards where it compares z and w only at those points where $y_i = 1$. Let $\{\mathsf{H}_\kappa : \{0,1\}^{\ell'(\kappa)} \to \{0,1\}^{\ell(\kappa)}\}_{\kappa\in\mathbb{N}}$ be a family of hash functions. We say that $\{\mathsf{H}_\kappa\}_\kappa$ is a family of admissible hash functions if there exists an efficient algorithm* AdmSmp *that takes as input 1^κ and $Q \in \mathbb{N}$ and outputs $(y,z) \in \{0,1\}^\ell \times \{0,1\}^\ell$ such that for every polynomial $Q(\kappa)$ and all $X^\star, X^{(1)}, \ldots X^{(Q)} \in \{0,1\}^{\ell'(\kappa)}$ with $X^\star \notin \{X^{(1)}, \ldots, X^{(Q)}\}$, we have*

$$\Pr_{(y,z)}\Big[\ \mathsf{WldCmp}(y,z,\mathsf{H}(X^\star)) = 0 \ \wedge \ \Big(\mathsf{WldCmp}(y,z,\mathsf{H}(X^{(j)})) = 1 \quad \forall j \in [Q]\Big)\ \Big] \geq \Delta_Q(\kappa),$$

for a noticeable function $\Delta_Q(\kappa)$, where the probability above is taken over the choice of $(y,z) \xleftarrow{\$} \mathsf{AdmSmp}(1^\kappa, Q)$.

As shown in previous works [Lys02, FHPS13], a family of error correcting codes $\{\mathsf{H}_\kappa : \{0,1\}^{\ell'(\kappa)} \to \{0,1\}^{\ell(\kappa)}\}_{\kappa\in\mathbb{N}}$ with constant relative distance $c \in (0, 1/2)$ is an admissible hash function. Explicit and efficient constructions of such codes are given in [SS96, Zém01, Gol08] to name a few.

3 Indexed Attribute-Based Signatures

In this section, we define the syntax and the security notion of indexed attribute-based signature (indexed ABS). We require indexed ABS to satisfy unforgeability and privacy. For the former, we consider two kinds of security notions that we call co-selective unforgeability and no-signing-query unforgeability. While the latter notion of unforgeability is weaker, we will show that an indexed ABS scheme that only satisfies this weaker security notion can be converted into a scheme with the stronger security notion without loosing privacy.

3.1 Indexed Attribute-Based Signature

Syntax. Let $\{\mathcal{C}_\kappa\}_{\kappa\in\mathbb{N}}$ be a family of circuits, where $\mathcal{C}_\kappa$ is a set of circuits with domain $\{0,1\}^{k(\kappa)}$ and range $\{0,1\}$, and the size of every circuit in $\mathcal{C}_\kappa$ is bounded by $\mathrm{poly}(\kappa)$. Let also $\{\mathcal{M}_\kappa\}_{\kappa\in\mathbb{N}}$ be a family of message spaces. In the following, we occasionally drop the subscript and simply write $\mathcal{C}$ and $\mathcal{M}$ when the meaning is clear. An indexed attribute-based signature (indexed ABS) scheme for the circuit class $\mathcal{C}$ is defined by the following algorithms:

$\mathsf{ABS.Setup}(1^\kappa, 1^N) \to (\mathsf{mpk}, \mathsf{msk})$: The setup algorithm takes as input the security parameter κ and the bound on the number of users N both in the unary form and outputs the master public key mpk and the master secret key msk.

$\mathsf{ABS.KeyGen}(\mathsf{msk}, i, x) \to \mathsf{sk}_x$: The key generation algorithm takes as input the master secret key msk, the user index $i \in [N]$, and the attribute $x \in \{0,1\}^k$ and outputs the user secret key sk_x. We assume that i and x are implicitly included in sk_x.

$\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_x, \mathsf{M}, C) \to \sigma$: The signing algorithm takes as input the master public key mpk, the secret key sk_x associated to x, a message $\mathsf{M} \in \mathcal{M}_\kappa$, and a policy $C \in \mathcal{C}_\kappa$ and outputs the signature σ.

$\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, C, \sigma) \to \top$ or $\bot$: The verification algorithm takes as input the master public key mpk, a message M, a policy C, and a signature σ. It outputs $\top$ if the signature is deemed valid and $\bot$ otherwise. We assume that the verification algorithm is deterministic.

We require correctness, privacy, and co-selective unforgeability.

Correctness. We require correctness: that is, for all κ, $N \in \mathrm{poly}(\kappa)$, $(\mathsf{mpk}, \mathsf{msk}) \in \mathsf{ABS.Setup}(1^\kappa, 1^N)$, $i \in [N]$, $x \in \{0,1\}^k, C \in \mathcal{C}_\kappa$ such that $C(x) = 1$, $\mathsf{M} \in \mathcal{M}_\kappa$, $\mathsf{sk}_x \in \mathsf{ABS.KeyGen}(\mathsf{msk}, i, x)$, and $\sigma \in \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_x, \mathsf{M}, C)$, $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, C, \sigma) = \top$ holds.

Perfect Privacy. We say that the ABS scheme has perfect privacy if for all κ, $N \in \mathrm{poly}(\kappa)$, $(\mathsf{mpk}, \mathsf{msk}) \in \mathsf{ABS.Setup}(1^\kappa, 1^N)$, $x_0, x_1 \in \{0,1\}^k$, $i_0, i_1 \in [N]$, $C \in \mathcal{C}_\kappa$ satisfying $C(x_0) = C(x_1) = 1$, $\mathsf{M} \in \mathcal{M}$, $\mathsf{sk}_{x_0} \in \mathsf{ABS.KeyGen}(\mathsf{msk}, i_0, x_0)$, and $\mathsf{sk}_{x_1} \in \mathsf{ABS.KeyGen}(\mathsf{msk}, i_1, x_1)$, the following distributions are the same:

$$\{\sigma_0 \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{x_0}, \mathsf{M}, C)\} \approx \{\sigma_1 \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{x_1}, \mathsf{M}, C)\}.$$

Co-Selective Unforgeability. We now define the co-selective unforgeability for ABS scheme. This security notion is defined by the following game between a challenger and an adversary A. During the game, the challenger maintains a list $\mathcal{Q}$, which is set to be empty at the beginning of the game.

Key Queries: At the beginning of the game, the adversary A is given 1^κ as input. It then sends 1^N, $\{(i, x^{(i)})\}_{i\in[N]}$, and $\mathcal{S} \subseteq [N]$ such that $x^{(i)} \in \{0,1\}^k$ for all $i \in [N]$ to the challenger.

Setup: The challenger runs $\mathsf{ABS.Setup}(1^\kappa, 1^N) \to (\mathsf{mpk}, \mathsf{msk})$ and $\mathsf{ABS.KeyGen}(\mathsf{msk}, i, x^{(i)}) \to \mathsf{sk}_{x^{(i)}}$ for $i \in [N]$. It then gives mpk and $\{\mathsf{sk}_{x^{(i)}}\}_{i\in[\mathcal{S}]}$ to A.

Signing Queries: During the game, A can make signing queries unbounded polynomially many times. When A queries (M, C, i) such that $\mathsf{M} \in \mathcal{M}$, $C \in \mathcal{C}$, $i \in [N]$, and $C(x^{(i)}) = 1$, the challenger runs $\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{x^{(i)}}, \mathsf{M}, C) \to \sigma$ and returns σ to A. The challenger then adds (M, C) to $\mathcal{Q}$.

Forgery: Eventually, A outputs $(\mathsf{M}^\star, C^\star, \sigma^\star)$ as the forgery. We say that A wins the game if:

1. $C^\star \in \mathcal{C}$,
2. $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}^\star, C^\star, \sigma^\star) \to \top$,
3. $C^\star(x^{(i)}) = 0$ for $i \in \mathcal{S}$,
4. $(\mathsf{M}^\star, C^\star) \notin \mathcal{Q}$.

We define the advantage of the adversary to be the probability that the adversary A wins in the above game, where the probability is taken over the coin tosses made by A and the challenger. We say that a scheme satisfies co-selective unforgeability if the advantage of any PPT adversary A in the above game is negligible in the security parameter.

No-Signing-Query Unforgeability. We now define a weaker definition of unforgeability. We define the no-signing-query unforgeability game by modifying the co-selective unforgeability game above by adding some more restrictions on A. Namely, we prohibit A from making any signing queries and require $\mathcal{S} \neq \emptyset$. We do not change the winning condition of the game and define the advantage of A as the probability that A wins. Note that Item 4 becomes vacuous because we will always have $\mathcal{Q} = \emptyset$. We say that a scheme satisfies no-signing-query unforgeability if the advantage of any PPT adversary A in the game is negligible.

Here, we provide some remarks on the syntax and the security definitions of indexed ABS.

Remark 2 (Comparing indexed ABS with standard ABS)**.** *The syntax of the indexed ABS is a relaxation of the standard ABS [MPR11, OT11, SAH16]: the setup algorithm takes 1^N as an additional input and the key generation algorithm takes an index i as an additional input. It is easy to check that standard ABS can be used as indexed ABS by simply ignoring the additional inputs.*

Remark 3 (Co-selective Unforgeability v.s. Selective Unforgeability)**.** *Here, we briefly compare co-selective unforgeability defined for indexed ABS with selective unforgeability [HLLR12, Tsa17] defined for (standard) ABS. They are in some sense dual notions and are incomparable. Concerning the signing key queries, the latter is a stronger notion since an adversary can adaptively make unbounded number of signing key queries, whereas in the former the adversary must make all the signing key queries before seeing* mpk*. On the other hand, concerning the target policy which an adversary makes a forgery on, the former is a stronger security definition since an adversary can choose its target $C^\star$ adaptively, where as in the latter an adversary must declare its target before seeing* mpk*.*

Remark 4 (Regarding No-Signing-Query Unforgeability)**.** *We note that the no-signing-query security is a very weak security notion. Notably, it may be possible for* A *to forge a signature* $(\mathsf{M}', C, \sigma')$ *such that* $C(x^{(i)}) = 0$ *for* $i \in \mathcal{S}$ *and* $\mathsf{M}' \neq \mathsf{M}$*, if* A *were to obtain a valid message-signature pair* (M, C, σ) *via a signing query (which we prohibit in the definition). In fact, the signing and the verification algorithms of our ABS scheme proposed in Sec. 5.2 ignores completely the message being input and our ABS scheme is vulnerable to such an attack. However, as we show in Sec. 3.2, an indexed ABS scheme that satisfies the no-signing-query unforgeability can be generically converted into a scheme that satisfies the stronger co-selective security.*

3.2 From No-Signing-Query Unforgeability to Co-selective Unforgeability

Here, we show that an indexed ABS scheme $\mathsf{ABS} = (\mathsf{ABS.Setup}, \mathsf{ABS.KeyGen}, \mathsf{ABS.Sign}, \mathsf{ABS.Vrfy})$ that is no-signing-query unforgeable can be generically converted into a new indexed ABS scheme $\mathsf{ABS}' = (\mathsf{ABS}'.\mathsf{Setup}, \mathsf{ABS}'.\mathsf{KeyGen}, \mathsf{ABS}'.\mathsf{Sign}, \mathsf{ABS}'.\mathsf{Vrfy})$ that is co-selective unforgeable. If ABS is perfectly private, so is ABS'. To enable the resulting scheme ABS' to deal with function class $\mathcal{C} = \{\mathcal{C}_\kappa\}_{\kappa\in\mathbb{N}}$, where $\mathcal{C}_\kappa$ is a set of circuits C such that $C : \{0,1\}^{k(\kappa)} \to \{0,1\}$, we require ABS to be able to deal with a (slightly) more complex function class $\mathcal{F} = \{\mathcal{F}_\kappa\}_{\kappa\in\mathbb{N}}$. We define $\mathcal{F}_\kappa$ as

$$\mathcal{F}_\kappa = \left\{ F[\widetilde{\mathsf{M}}, C] : \{0,1\}^{k(\kappa)+2\ell(\kappa)+1} \to \{0,1\} \;\middle|\; \widetilde{\mathsf{M}} \in \{0,1\}^{\ell(\kappa)},\ C \in \mathcal{C}_\kappa \right\}, \tag{1}$$

where $F[\widetilde{\mathsf{M}}, C]$ is defined in Fig. 1. We assume that the circuit $F[\widetilde{\mathsf{M}}, C]$ is deterministically constructed from $\widetilde{\mathsf{M}}$ and C in a predetermined way. Let $\{\mathcal{H}_\kappa\}_\kappa$ be a family of collision resistant hash functions where an index $h \in \mathcal{H}_\kappa$ specifies a function $h : \{0,1\}^* \to \{0,1\}^{\ell'(\kappa)}$, where $\{0,1\}^{\ell'(\kappa)}$ is the input space of an admissible hash function $\mathsf{H}_\kappa : \{0,1\}^{\ell'(\kappa)} \to \{0,1\}^{\ell(\kappa)}$. We construct ABS' as follows.

$\mathsf{ABS}'.\mathsf{Setup}(1^\kappa, 1^N)$: It runs $\mathsf{ABS.Setup}(1^\kappa, 1^{N+1}) \to (\mathsf{mpk}, \mathsf{msk})$ and samples a random index of collision resistant hash function $h \xleftarrow{\$} \mathcal{H}_\kappa$. It then outputs the master public key $\mathsf{mpk}' = (\mathsf{mpk}, h)$ and the master secret key $\mathsf{msk}' := \mathsf{msk}$.

$\mathsf{ABS}'.\mathsf{KeyGen}(\mathsf{msk}, i, x)$: It runs $\mathsf{ABS.KeyGen}(\mathsf{msk}, i, x\|0^{2\ell+1}) \to \mathsf{sk}_{x\|0^{2\ell+1}}$ and returns $\mathsf{sk}'_x := \mathsf{sk}_{x\|0^{2\ell+1}}$.

$\mathsf{ABS}'.\mathsf{Sign}(\mathsf{mpk}', \mathsf{sk}'_x, \mathsf{M}, C)$: It first parses $\mathsf{mpk}' \to (\mathsf{mpk}, h)$ and $\mathsf{sk}'_x \to \mathsf{sk}_{x\|0^{2\ell+1}}$ and computes $\widetilde{\mathsf{M}} = \mathsf{H}(h(\mathsf{M}\|C))$. It then constructs a circuit $F[\widetilde{\mathsf{M}}, C]$ that is defined as in Fig. 1. It finally runs $\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{x\|0^{2\ell+1}}, \mathsf{M}, F[\widetilde{\mathsf{M}}, C]) \to \sigma$ and outputs $\sigma' := \sigma$.

$\mathsf{ABS}'.\mathsf{Vrfy}(\mathsf{mpk}', \mathsf{M}, C, \sigma)$: It first parses $\mathsf{mpk}' \to (\mathsf{mpk}, h)$. It then computes $\widetilde{\mathsf{M}} = \mathsf{H}(h(\mathsf{M}\|C))$ and constructs a circuit $F[\widetilde{\mathsf{M}}, C]$ that is defined as in Fig. 1. It then outputs $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, F[\widetilde{\mathsf{M}}, C], \sigma)$.

Correctness. We observe that if $C(x) = 1$, we have $F[\widetilde{\mathsf{M}}, C](x\|0^{2\ell+1}) = C(x) = 1$ by the definition of $F[\widetilde{\mathsf{M}}, C]$. The correctness of ABS' therefore follows from that of ABS.

Perfect Privacy. The following theorem addresses the privacy of ABS' constructed above.

Theorem 3. *If* ABS *is perfectly private, so is* ABS'.

Proof. If $C(x_0) = C(x_1) = 1$, we have $F[\widetilde{\mathsf{M}}, C](x_0\|0^{2\ell+1}) = C(x_0) = 1$ and $F[\widetilde{\mathsf{M}}, C](x_1\|0^{2\ell+1}) = C(x_1) = 1$ by the definition of $F[\widetilde{\mathsf{M}}, C]$. The theorem therefore follows from the perfect privacy of ABS. □

$F[\widetilde{\mathsf{M}}, C](x\|y\|z\|b)$

Hardwired constants: A bit string $\widetilde{\mathsf{M}} \in \{0,1\}^\ell$ and a circuit $C : \{0,1\}^k \to \{0,1\}$.

1. Parse the input to retrieve $x \in \{0,1\}^k$, $y, z \in \{0,1\}^\ell$, and $b \in \{0,1\}$.
2. If $b = 0$, output $C(x)$.
3. If $b = 1$, output $\mathsf{WldCmp}(y, z, \widetilde{\mathsf{M}}) \in \{0,1\}$.

Figure 1: Description of the circuit $F[\widetilde{\mathsf{M}}, C]$.

Co-selective Unforgeability. The following theorem addresses the co-selective unforgeability of ABS'.

Theorem 4. *If* ABS *is no-signing-query unforgeable and perfectly private,* $\mathcal{H}_\kappa$ *is a family of collision resistant hash functions, and* H_κ *is an admissible hash function, then* ABS' *is co-selective unforgeable.*

Proof. To prove the theorem, it suffices to show that any PPT adversary A against the co-selective unforgeability of ABS' with advantage ϵ can be converted into a PPT adversary B against the no-signing-query unforgeability of ABS with advantage polynomially related to ϵ. Therefore, assuming that ABS satisfies no-signing-query unforgeability, we conclude that ϵ is negligible. We show this by considering the following sequence of games. In the following, let E_i denote the probability that A is successful in $\mathsf{Game}\ i$ and the challenger does not abort. We also let $Q(\kappa)$ be the upper bound of the number of signing queries made by A during the game.

$\mathsf{Game}\ 0$**.** We define $\mathsf{Game}\ 0$ as the actual experiment between the challenger and the adversary A. The success probability of A in the game be $\Pr[\mathsf{E}_0] = \epsilon$.

$\mathsf{Game}\ 1$**.** In this game, the challenger aborts if A outputs $(\mathsf{M}^\star, C^\star, \sigma^\star)$ such that $h(\mathsf{M}^\star\|C^\star) = h(\mathsf{M}\|C)$ for some $(\mathsf{M}, C) \in \mathcal{Q}$ as the forgery. By a straightforward reduction from the collision resistance of $\mathcal{H}$, we have $\Pr[\mathsf{E}_1] \geq \Pr[\mathsf{E}_0] - \mathrm{negl}(\kappa)$.

$\mathsf{Game}\ 2$**.** In this game, we change $\mathsf{Game}\ 1$ so that the challenger performs the following additional step at the end of the game. First, the challenger runs $(y, z) \xleftarrow{\$} \mathsf{AdmSmp}(1^\kappa, Q)$ and checks whether the following condition holds:

$$\left(\ \mathsf{WldCmp}(y, z, \mathsf{H}(h(\mathsf{M}\|C))) = 1 \quad \forall(\mathsf{M}, C) \in \mathcal{Q}\ \right) \ \wedge\ \left(\ \mathsf{WldCmp}(y, z, \mathsf{H}(h(\mathsf{M}^\star\|C^\star))) = 0\ \right).$$

If it does not hold, the challenger aborts the game. Since we have $|\mathcal{Q}| \leq Q$ and $h(\mathsf{M}^\star\|C^\star) \neq h(\mathsf{M}\|C)$ for $(\mathsf{M}, C) \in \mathcal{Q}$, we have $\Pr[\mathsf{E}_2] \geq \Delta_Q(\kappa) \cdot \Pr[\mathsf{E}_1]$ for a noticeable function $\Delta_Q(\kappa)$ by the property of the admissible hash.

$\mathsf{Game}\ 3$**.** In this game, the challenger chooses $(y, z) \xleftarrow{\$} \mathsf{AdmSmp}(1^\kappa, Q)$ at the beginning of the game instead of at the end of the game. Furthermore, the challenger aborts the game as soon as A makes a signing query for (M, C, i) such that $\mathsf{WldCmp}(y, z, \mathsf{H}(h(\mathsf{M}\|C))) = 0$ or makes a forgery $(\mathsf{M}^\star, C^\star, \sigma^\star)$ such that $\mathsf{WldCmp}(y, z, \mathsf{H}(h(\mathsf{M}^\star\|C^\star))) = 1$. Since this is simply a conceptual change, we have $\Pr[\mathsf{E}_3] = \Pr[\mathsf{E}_2]$.

Game 4 In this game, we change the way A answers the signing queries. The challenger runs $\mathsf{ABS.KeyGen}(\mathsf{msk}, N+1, 0^k\|y\|z\|1) \to \mathsf{sk}_{0^k\|y\|z\|1}$ right after having sampled mpk' and msk'. Then, when A makes a signing query for (M, C, i), B first computes $\widetilde{\mathsf{M}} = \mathsf{H}(h(\mathsf{M}\|C))$ and aborts if $\mathsf{WldCmp}(y, z, \widetilde{\mathsf{M}}) = 0$ (as specified in Game 3). Otherwise, B then runs

$$\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{0^k\|y\|z\|1}, \mathsf{M}, F[\widetilde{\mathsf{M}}, C]) \to \sigma$$

and returns σ to A. As we will show in Lemma 1, we have $\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_3]$ by the perfect privacy of ABS.

Finally, in Lemma 2, we show $\Pr[\mathsf{E}_4] \leq \mathrm{negl}(\kappa)$. Putting these together, we have $\Pr[\mathsf{E}_0] = \epsilon = \mathrm{negl}(\kappa)$, which concludes the proof of the theorem. It remains to prove Lemma 1 and 2. □

Lemma 1. *If* ABS *is perfectly private, we have* $\Pr[\mathsf{E}_3] = \Pr[\mathsf{E}_4]$.

Proof. To sample an answer σ for a signing query (M, C, i), the challenger runs $\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{0^k\|y\|z\|1}, \mathsf{M}, F[\widetilde{\mathsf{M}}, C]) \to \sigma$ in Game 4, whereas it is sampled as $\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{x^{(i)}\|0^{2\ell+1}}, \mathsf{M}, F[\widetilde{\mathsf{M}}, C]) \to \sigma$ in Game 3. We have $F[\widetilde{\mathsf{M}}, C](x^{(i)}\|0^{2\ell+1}) = C(x^{(i)}) = 1$ and $F[\widetilde{\mathsf{M}}, C](0^k\|y\|z\|1) = \mathsf{WldCmp}(y, z, \widetilde{\mathsf{M}}) = 1$, where the latter holds when the abort condition is not satisfied. This implies that the distributions of σ in these two games are the same due to the perfect privacy of ABS. By applying this argument for each of the signing queries one-by-one, the lemma follows. □

Lemma 2. *If* ABS *has no-evaluation-query unforgeability, we have* $\Pr[\mathsf{E}_4] \leq \mathrm{negl}(\kappa)$.

Proof. For the sake of the contradiction, let us assume that E_4 happens with non-negligible probability ϵ. We then construct an adversary B that breaks the no-evaluation-query unforgeability of ABS with the same probability. The adversary B proceeds as follows.

At the beginning of the game, B is given 1^κ from its challenger. Then, B inputs 1^κ on A, who returns 1^N, $\{(i, x^{(i)})\}_{i\in[N]}$, and $\mathcal{S} \subseteq [N]$. Here, $x^{(i)} \in \{0,1\}^k$. Then, B samples $h \xleftarrow{\$} \mathcal{H}_\kappa$ and $(y, z) \xleftarrow{\$} \mathsf{AdmSmp}(1^\kappa, Q)$. It then sets $\bar{x}^{(i)} := x^{(i)}\|0^{2\ell+1}$ for $i \in [N]$, $\bar{x}^{(N+1)} := 0^k\|y\|z\|1$, and $\bar{\mathcal{S}} := \mathcal{S} \cup \{N+1\}$. Then, it submits 1^{N+1}, $\{\bar{x}^{(i)}\}_{i\in[N+1]}$, and $\bar{\mathcal{S}}$ to the challenger. Then, the challenger gives mpk and $\{\mathsf{sk}_{x^{(i)}\|0^{2\ell+1}}\}_{i\in[\mathcal{S}]} \cup \{\mathsf{sk}_{0^k\|y\|z\|1}\}$ to B. B then passes $\mathsf{mpk}' := (\mathsf{mpk}, h)$ and $\{\mathsf{sk}_{x^{(i)}\|0^{2\ell+1}}\}_{i\in\mathcal{S}}$ to A. During the game, A makes signing queries. B handles these queries using $\mathsf{sk}_{0^k\|y\|z\|1}$ as specified in Game 4. At the end of the game, A outputs its forgery $(\mathsf{M}^\star, C^\star, \sigma^\star)$. Then, B computes $\widetilde{\mathsf{M}}^\star = \mathsf{H}(h(\mathsf{M}^\star\|C^\star))$. It aborts and outputs $\perp$ if $\mathsf{ABS}'.\mathsf{Vrfy}(\mathsf{mpk}, \mathsf{M}^\star, C^\star, \sigma^\star) \neq \top$ or $\mathsf{WldCmp}(y, z, \widetilde{\mathsf{M}}^\star) = 1$ (as specified in Game 3). Otherwise, B outputs $(\mathsf{M}^\star, F[\widetilde{\mathsf{M}}^\star, C^\star], \sigma^\star)$ as its forgery.

We claim that B wins the game whenever E_4 occurs. To see this, we first observe that we have $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}^\star, F[\widetilde{\mathsf{M}}^\star, C^\star], \sigma^\star) = \top$ by $\mathsf{ABS}'.\mathsf{Vrfy}(\mathsf{mpk}, \mathsf{M}^\star, C^\star, \sigma^\star) = \top$. We also have $\bar{\mathcal{S}} \neq \emptyset$ even if $\mathcal{S} = \emptyset$ because $N+1 \in \bar{\mathcal{S}}$. We then prove that A has never made prohibited corrupt queries. Namely, we show $F[\widetilde{\mathsf{M}}^\star, C^\star](\bar{x}^{(i)}) = 0$ for all $i \in \bar{\mathcal{S}}$. For $i \in \mathcal{S}$, we have $F[\widetilde{\mathsf{M}}^\star, C^\star](\bar{x}^{(i)}) = F[\widetilde{\mathsf{M}}^\star, C^\star](x^{(i)}\|0^{2\ell+1}) = C^\star(x^{(i)}) = 0$, where the last equality follows from the restriction posed on A. As for $i = N+1$, we have $F[\widetilde{\mathsf{M}}^\star, C^\star](\bar{x}^{(N+1)}) = F[\widetilde{\mathsf{M}}^\star, C^\star](0^k\|y\|z\|1) = \mathsf{WldCmp}(y, z, \widetilde{\mathsf{M}}^\star) = 0$, where the last equality follows from the abort condition introduced in Game 2. We also observe that B has never made any signing query during the game. Since B perfectly simulates Game 4, we have that the winning probability of B is exactly ϵ. This concludes the proof of the lemma. □

4 Generic Construction of Group Signatures

In this section, we give a generic construction of a GS scheme from three building blocks: an indexed ABS, an OTS, and an SKE. As we will show in Sec. 7, by appropriately instantiating the building blocks, we obtain the first lattice-based GS scheme in the standard model.

Ingredients. Here, we give a generic construction of a GS scheme $\mathsf{GS} = (\mathsf{GS.KeyGen}, \mathsf{GS.Sign}, \mathsf{GS.Vrfy}, \mathsf{GS.Open})$ from an indexed ABS scheme $\mathsf{ABS} = (\mathsf{ABS.Setup}, \mathsf{ABS.KeyGen}, \mathsf{ABS.Sign}, \mathsf{ABS.Vrfy})$ with perfect privacy and co-selective unforgeability, an OTS scheme $\mathsf{OTS} = (\mathsf{OTS.KeyGen}, \mathsf{OTS.Sign}, \mathsf{OTS.Vrfy})$ with strong unforgeability, and an SKE scheme $\mathsf{SKE} = (\mathsf{SKE.Gen}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ with key robustness and INDr-CCA security. We require the underlying primitives to satisfy the following constraints:

- $\mathsf{SKE}.\mathcal{M}_\kappa \supseteq [N+1] \times \{0,1\}^{\ell_1(\kappa)}$, where $\mathsf{SKE}.\mathcal{M}_\kappa$ denotes the plaintext space of SKE and $\ell_1(\kappa)$ denotes the upper-bound on the length of ovk that is output by $\mathsf{OTS.Setup}(1^\kappa)$.
- We require the underlying indexed ABS scheme to be able to deal with function class $\mathcal{C} = \{\mathcal{C}_\kappa\}_{\kappa\in\mathbb{N}}$, where $\mathcal{C}_\kappa$ is defined as

$$\mathcal{C}_\kappa = \left\{ C[\mathsf{ovk}, \mathsf{ct}] \;\middle|\; \mathsf{ovk} \in \{0,1\}^{\ell_1(\kappa)}, \mathsf{ct} \in \{0,1\}^{\ell_2(\kappa)} \right\}, \tag{2}$$

  where $C[\mathsf{ovk}, \mathsf{ct}]$ is defined in Fig. 2 and $\ell_2(\kappa)$ is the upper bound on the length of a ciphertext ct output by $\mathsf{SKE.Enc}(\mathsf{K}, \mathsf{M})$ for $\mathsf{K} \in \mathsf{SKE.Gen}(\mathsf{SKE.Setup}(1^\kappa))$ and $\mathsf{M} \in \mathsf{SKE}.\mathcal{M}_\kappa$.
- We require $\mathsf{OTS}.\mathcal{M}_\kappa = \{0,1\}^*$, where $\mathsf{OTS}.\mathcal{M}_\kappa$ denotes the message space of OTS. Note that any OTS scheme with sufficiently large message space can be modified to satisfy this condition by applying a collision resistant hash to a message before signing.

Construction. We construct GS as follows.

$\mathsf{GS.KeyGen}(1^\kappa, 1^N)$: It first samples $\mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa)$ and $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{ABS.Setup}(1^\kappa, 1^{N+1})$. It then samples $\mathsf{K}_i \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp})$ and $\mathsf{sk}_{i\|\mathsf{K}_i} \xleftarrow{\$} \mathsf{ABS.KeyGen}(\mathsf{msk}, i, i\|\mathsf{K}_i)$ for $i \in [N]$. Finally, it outputs

$$\mathsf{gpk} := (\ \mathsf{pp},\ \mathsf{mpk}\ ), \quad \mathsf{gok} := \{\ \mathsf{K}_i\ \}_{i\in[N]}, \quad \left\{\ \mathsf{gsk}_i := (\ i,\ \mathsf{K}_i,\ \mathsf{sk}_{i\|\mathsf{K}_i}\ )\ \right\}_{i\in[N]}.$$

$\mathsf{GS.Sign}(\mathsf{gsk}_i, \mathsf{M})$: It first samples $(\mathsf{ovk}, \mathsf{osk}) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$ and computes $\mathsf{ct} \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}_i, i\|\mathsf{ovk})$. It then runs

$$\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{i\|\mathsf{K}_i}, C[\mathsf{ovk}, \mathsf{ct}], \mathsf{M}) \to \sigma,$$

where the circuit $C[\mathsf{ovk}, \mathsf{ct}]$ is defined in Fig. 2. It then runs

$$\mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M}\|\sigma) \to \tau.$$

Finally, it outputs $\Sigma := (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$.

$\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}, \Sigma)$: It first parses $\Sigma \to (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$. It then outputs $\top$ if

$$\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], \sigma) = \top \ \wedge\ \mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M}\|\sigma, \tau) = \top,$$

where $C[\mathsf{ovk}, \mathsf{ct}]$ is defined in Fig. 2. Otherwise, it outputs $\bot$.

$C[\mathsf{ovk}, \mathsf{ct}](i \| \mathsf{K})$

Hardwired constants: A verification key ovk of OTS and a ciphertext ct of SKE.

1. Parse the input to retrieve $i \in [N+1]$ and K. If the input does not conform to the format, output 0.
2. If $i = N + 1$, output 1.
3. Compute $\mathsf{SKE.Dec}(\mathsf{K}, \mathsf{ct}) = i' \| \mathsf{ovk}'$. If $i' = i$ and $\mathsf{ovk}' = \mathsf{ovk}$, output 1. Otherwise, output 0.

Figure 2: Description of the circuit $C[\mathsf{ovk}, \mathsf{ct}]$.

$\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}, \Sigma)$: It first runs $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}, \Sigma)$ and returns $\perp$ if the verification result is $\perp$. Otherwise, it parses $\Sigma \to (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$. It then computes $d_i \leftarrow \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct})$ for $i \in [N]$ and outputs the smallest index i such that $d_i \neq \perp$. If there is not such i, it returns $\perp$.

Remark 5 (Alternative Construction Using Public Key Encryption)**.** *Here, we discuss an alternative construction of a GS scheme using a public key encryption (PKE) instead of an SKE. In the construction, we instead hardwire a secret key of a weak PRF to the ABS signing key. Then, when signing, a signer chooses a random input to the weak PRF, generates a ciphertext of the PKE using the randomness that is derived by evaluating the weak PRF on the random input, and proves that the ciphertext is generated in this manner using the ABS signing key. Note that all users use the same public key when generating the ciphertext, while each user is provided with a different weak PRF key. The advantage of this construction over the construction in Sec. 4 is that the open algorithm can be very efficient. Namely, we set* gok *as the secret key of the PKE and the open algorithm simply decrypts the ciphertext and extracts the corresponding user index. We do not adopt this alternative construction because we do not know how to instantiate it from the LWE assumption with polynomial approximation factors. In particular, to the best of our knowledge, all constructions of weak PRFs that can be evaluated in* $\boldsymbol{NC}^1$*, which is the required property in order to combine it with our indexed ABS, require the LWE assumption with super-polynomial approximation factors.*

Correctness. We show that correctly generated signature $\Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$ passes the verification. We have $\mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M} \| \sigma, \tau) = \top$ by the correctness of OTS. Furthermore, we have $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], \sigma) = \top$ since $C[\mathsf{ovk}, \mathsf{ct}](i \| \mathsf{K}_i) = 1$, which follows from $\mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}) = i \| \mathsf{ovk}$ by the correctness of SKE.

CCA-Selfless Anonymity. The following theorem addresses the CCA-selfless anonymity of the above GS scheme.

Theorem 5. *If* ABS *is perfectly private and co-selective unforgeable,* OTS *is strongly unforgeable, and* SKE *is INDr-CCA-secure and key robust, then* GS *constructed above is CCA-selfless anonymous.*

Proof. We show the theorem by considering the following sequence of games between a PPT adversary A against the CCA-selfless anonymity and the challenger. In the following, let E_i denote the probability that A wins the game in $\mathsf{Game}\ i$.

Game 0: We define Game 0 as an ordinary CCA-selfless anonymity game between A and the challenger. The advantage of A is $|\Pr[\mathsf{E}_0] - 1/2|$. In the following, the challenge signature is denoted by $\Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star)$. Here, note that $\mathsf{ct}^\star$ is constructed using $\mathsf{K}_{i^\star_{\mathsf{coin}}}$ by the specification of the game.

Game 1: In this game, we change the game so that the challenger aborts the game and forces A to output a random bit if there exists $i \in [N]\backslash\{i^\star_{\mathsf{coin}}\}$ such that $\mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot$. As we will show in Lemma 3, we have $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_1]| = \mathrm{negl}(\kappa)$ by the key robustness of SKE.

Game 2: In this game, we change the way the open queries are answered after the challenge phase. When A makes an open query for $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$ such that $\mathsf{ct} = \mathsf{ct}^\star$ and $\mathsf{ovk} \neq \mathsf{ovk}^\star$, it returns $\bot$. As we will show in Lemma 4, we have $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathrm{negl}(\kappa)$ assuming the co-selective unforgeability of ABS.

Game 3: Recall that in Game 1, the challenger aborts the game if there exists $i \in [N]\backslash\{i^\star_{\mathsf{coin}}\}$ such that $\mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot$. In this game, we remove this abort condition and change the game so that the challenger continues the game even if there exists such i. As we will show in Lemma 5, we have $|\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_3]| = \mathrm{negl}(\kappa)$ by the key robustness of SKE. This step will be used to argue the winning probability of adversary in our final game Game 7.

Game 4: In this game, we further change the way the open queries are answered after the challenge phase. When A makes an open query for $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$ such that $\mathsf{ct} = \mathsf{ct}^\star$ and $\mathsf{ovk} = \mathsf{ovk}^\star$, it returns $\bot$. As we will show in Lemma 6, we have $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathrm{negl}(\kappa)$ assuming the strong unforgeability of OTS. Notice that in this game, the challenger returns $\bot$ for any open query $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$ with $\mathsf{ct} = \mathsf{ct}^\star$ after the challenge phase.

Game 5: In this game, the challenger samples $\mathsf{K}_{N+1} \xleftarrow{\$} \mathcal{K}$ and runs $\mathsf{sk}_{N+1\|K_{N+1}} \xleftarrow{\$} \mathsf{ABS.KeyGen}(\mathsf{msk}, N+1, N+1\|K_{N+1})$ after having run $\mathsf{GS.KeyGen}(1^\kappa, 1^N)$. Then, all the signatures created by the challenger are answered using $\mathsf{sk}_{N+1\|\mathsf{K}_{N+1}}$. More specifically, the challenger answers the signing queries and the challenge queries as follows.

- When A makes a signing query for (b, M), the challenger samples $(\mathsf{ovk}, \mathsf{osk}) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$, $\mathsf{ct} \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}_{i^\star_b}, i^\star_b\|\mathsf{ovk})$,

$$\sigma \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{N+1\|K_{N+1}}, C[\mathsf{ovk}, \mathsf{ct}], \mathsf{M}),$$

  and $\tau \xleftarrow{\$} \mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M}\|\sigma)$ and returns $\Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$ to A.
- When A makes the challenge query for $\mathsf{M}^\star$, the challenger samples $(\mathsf{ovk}^\star, \mathsf{osk}^\star) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$, $\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}_{i^\star_{\mathsf{coin}}}, i^\star_{\mathsf{coin}}\|\mathsf{ovk}^\star)$,

$$\sigma^\star \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{N+1\|K_{N+1}}, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \mathsf{M}^\star),$$

  and $\tau^\star \xleftarrow{\$} \mathsf{OTS.Sign}(\mathsf{osk}^\star, \mathsf{M}^\star\|\sigma^\star)$ and returns $\Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star)$ to A.

As we will show in Lemma 7, we have $\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_5]$ assuming the perfect privacy of ABS.

Game 6: In this game, we change the way the challenge query is answered. When A makes the challenge query for $\mathsf{M}^\star$, the challenger proceeds as in the previous game if $\mathsf{coin} = 1$. If $\mathsf{coin} = 0$, the challenger samples $(\mathsf{ovk}^\star, \mathsf{osk}^\star) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$,

$$\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.CTSamp}(\mathsf{pp}),$$

$\sigma^\star \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{N+1\|K_{N+1}}, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \mathsf{M}^\star)$, and $\tau^\star \xleftarrow{\$} \mathsf{OTS.Sign}(\mathsf{osk}^\star, \mathsf{M}^\star\|\sigma^\star)$ and returns $\Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star)$ to A. As we will show in Lemma 8, we have $|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]| = \mathrm{negl}(\kappa)$ assuming the CCA-security of SKE.

Game 7: In this game, we further change the way the challenge query is answered. When A makes the challenge query for $\mathsf{M}^\star$, the challenger generates the challenge signature as follows (regardless of the value of coin). It samples $(\mathsf{ovk}^\star, \mathsf{osk}^\star) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$,

$$\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.CTSamp}(\mathsf{pp}),$$

$\sigma^\star \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{N+1\|K_{N+1}}, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \mathsf{M}^\star)$, and $\tau^\star \xleftarrow{\$} \mathsf{OTS.Sign}(\mathsf{osk}^\star, \mathsf{M}^\star\|\sigma^\star)$ and returns $\Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star)$ to A. As we will show in Lemma 9, we have $|\Pr[\mathsf{E}_6] - \Pr[\mathsf{E}_7]| = \mathrm{negl}(\kappa)$ assuming the CCA-security of SKE.

In Game 7, the challenge signature $\Sigma^\star$ is sampled from the same distribution regardless of the value of coin. Furthermore, due to the change we made in Game 3 (i.e., getting rid of the abort procedure), the view of the adversary A is independent from the value of coin. Therefore, we have $\Pr[\mathsf{E}_7] = 1/2$. By combining Lemma 3, 4, 5, 6, 7, 8, and 9, we have that $|\Pr[\mathsf{E}_0] - 1/2|$ is negligible.

Lemma 3. *If* SKE *is key robust, we have* $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_1]| = \mathrm{negl}(\kappa)$.

Proof. Let F be the event such that $\mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot$ holds for some $i \in [N]\backslash\{i^\star_{\mathsf{coin}}\}$. It can be easily seen that $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_1]| \leq \Pr[\mathsf{F}]$ and it suffices to show that $\Pr[\mathsf{F}]$ is negligible. We have

$$\begin{aligned}
& \Pr[\mathsf{F}] \\
\leq\ & \Pr\left[\begin{array}{c} \mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa),\ \mathsf{K}_j \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp}) \text{ for } j \in [N] : \\ \exists \mathsf{ct}^\star \in \{0,1\}^*,\ \exists i, i^\star \in [N] \text{ s.t. } i \neq i^\star \wedge\ \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot \wedge\ \mathsf{SKE.Dec}(\mathsf{K}_{i^\star}, \mathsf{ct}^\star) \neq \bot \end{array}\right] \\
\leq\ & \sum_{i,i^\star \in [N] \text{ s.t. } i \neq i^\star} \Pr\left[\begin{array}{c} \mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa),\ \mathsf{K}_i, \mathsf{K}_{i^\star} \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp}) \ : \\ \exists \mathsf{ct}^\star \in \{0,1\}^* \text{ s.t. } \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot \wedge\ \mathsf{SKE.Dec}(\mathsf{K}_{i^\star}, \mathsf{ct}^\star) \neq \bot \end{array}\right] \\
\leq\ & N(N-1)/2 \cdot \mathrm{negl}(\kappa) \\
=\ & \mathrm{negl}(\kappa),
\end{aligned}$$

where the first inequality is by the fact that $\mathsf{SKE.Dec}(\mathsf{K}_{i^\star_{\mathsf{coin}}}, \mathsf{ct}^\star) \neq \bot$, the second inequality is by the union bound, and the third inequality is by the key robustness of SKE. This concludes the proof of the lemma. □

Lemma 4. *If* ABS *is co-selective unforgeable, we have* $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathrm{negl}(\kappa)$.

Proof. We observe that Game 1 and Game 2 are the same unless the adversary makes an open query $\Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$ such that $\mathsf{ct} = \mathsf{ct}^\star$, $\mathsf{ovk} \neq \mathsf{ovk}^\star$, and $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], \sigma) = \top \wedge \mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M}\|\sigma) = \top$ after the challenge phase. We denote this event by F and define ϵ as the probability of F occurring in Game 1. Since $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| \leq \Pr[\mathsf{F}]$, it suffices to show ϵ is negligible. To show this, we prove that there exists an adversary B who breaks the co-selective unforgeability of ABS with probability ϵ. We give the description of B in the following.

At the beginning of the game, B is given 1^κ from its challenger. Then, B inputs 1^κ on A, who returns $(i^\star_0, i^\star_1) \in [N] \times [N]$. Then, B samples $\mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa)$, $\mathsf{coin} \xleftarrow{\$} \{0,1\}$, and

$\mathsf{K}_i \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp})$ for $i \in [N]$ and sends 1^N, $\{(i, i\|\mathsf{K}_i)\}_{i\in[N]}$, and $\mathcal{S} = [N]$ to its challenger. Then, the challenger gives mpk and $\{\mathsf{sk}_{i\|\mathsf{K}_i}\}_{i\in[N]}$ to B. B then sets $\mathsf{gsk}_i = (i, \mathsf{K}_i, \mathsf{sk}_{i\|\mathsf{K}_i})$ for $i \in [N]$ and $\mathsf{gok} = \{\mathsf{K}_i\}_{i\in[N]}$ and gives mpk and $\{\mathsf{gsk}_i\}_{i\in[N]\setminus\{i_0^\star, i_1^\star\}}$ to A. B answers the challenge query, signing queries, and open queries as in Game 1 using $\{\mathsf{gsk}_i\}_{i\in[N]}$ and gok except for the following modification: When A makes an open query for $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$ such that $\mathsf{ct} = \mathsf{ct}^\star$, $\mathsf{ovk} \neq \mathsf{ovk}^\star$, and $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], \sigma) = \top$ after the challenge phase, it outputs $(\mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], \sigma)$ as its forgery and halts. If A terminates without making such an open query, B aborts.

We claim that B wins the game whenever F happens. To prove this, we show that B has not made any prohibited key query. Namely, we show that for the final output $(\mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], \sigma)$ of B, $C[\mathsf{ovk}, \mathsf{ct}](i\|\mathsf{K}_i) = 0$ holds for all $i \in [N]$. For $i \in [N]\setminus\{i^\star_{\mathsf{coin}}\}$, we have $\mathsf{SKE.Dec}(K_i, \mathsf{ct}) = \mathsf{SKE.Dec}(K_i, \mathsf{ct}^\star) = \bot$ by the change introduced in Game 1 and thus $C[\mathsf{ovk}, \mathsf{ct}](i\|\mathsf{K}_i) = 0$. For $i = i^\star_{\mathsf{coin}}$, we also have $C[\mathsf{ovk}, \mathsf{ct}](i^\star_{\mathsf{coin}}\|\mathsf{K}_{i^\star_{\mathsf{coin}}}) = 0$ because $\mathsf{SKE.Dec}(\mathsf{K}_{i^\star_{\mathsf{coin}}}, \mathsf{ct}^\star) = i^\star_{\mathsf{coin}}\|\mathsf{ovk}^\star \neq i^\star_{\mathsf{coin}}\|\mathsf{ovk}$. Since B perfectly simulates Game 1 unless F occurs, the winning probability of B is exactly ϵ. This concludes the proof of the lemma. □

Lemma 5. *If* SKE *is key robust, we have* $|\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_3]| = \mathrm{negl}(\kappa)$.

Proof. The proof is the same as that of Lemma 3. □

Lemma 6. *If* OTS *is a strongly unforgeable one-time signature, we have* $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathrm{negl}(\kappa)$.

Proof. We observe that Game 3 and Game 4 are the same unless the adversary makes an open query $\Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$ such that $\mathsf{ct} = \mathsf{ct}^\star$, $\mathsf{ovk} = \mathsf{ovk}^\star$, and $\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], \sigma) = \top \wedge \mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M}\|\sigma) = \top$ after the challenge query. We denote this event by F and define ϵ as the probability of F occurring in Game 3. Since $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| \leq \Pr[\mathsf{F}]$, it suffices to show ϵ is negligible. To show this, we prove that there exists B who breaks the strong unforgeability of OTS with probability ϵ. We give the description of B in the following.

At the beginning of the game, B is given 1^κ and $\mathsf{ovk}^\star$ from its challenger. Then, B inputs 1^κ to A, who returns $(i_0^\star, i_1^\star) \in [N] \times [N]$. Then, B runs $\mathsf{GS.KeyGen}(1^\kappa, 1^N)$ to obtain gpk, gok and $\{\mathsf{gsk}_i\}_{i\in[N]}$ and samples $\mathsf{coin} \xleftarrow{\$} \{0,1\}$. It then returns $\{\mathsf{gsk}_i\}_{i\in[N]\setminus\{i_0^\star, i_1^\star\}}$ to A. B answers the signing queries and the open queries as in Game 3 using $\{\mathsf{gsk}_i\}_{i\in[N]}$ and gok until the challenge phase. When A makes the challenge query for $\mathsf{M}^\star$, B computes $\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}_{i^\star_{\mathsf{coin}}}, i^\star_{\mathsf{coin}}\|\mathsf{ovk}^\star)$ and runs $\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{i^\star_{\mathsf{coin}}\|\mathsf{K}_{i^\star_{\mathsf{coin}}}}, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \mathsf{M}^\star) \to \sigma^\star$. B then queries a signature on $\mathsf{M}^\star\|\sigma^\star$ to its challenger. B is then given $\tau^\star$ from the challenger and gives $\Sigma^\star := (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star)$ to A. After the challenge phase, B answers the signing queries and open queries as before except for the following modification: When A makes an open query for $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$ such that $\mathsf{ct} = \mathsf{ct}^\star$, $\mathsf{ovk} = \mathsf{ovk}^\star$, and $\mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M}, \tau) = \top$, it outputs $(\mathsf{M}\|\sigma, \tau)$ as its forgery and stops. If A terminates without making such an open query, B aborts.

We claim that B wins the game whenever F happens. To prove this, it suffices to show $(\mathsf{M}\|\sigma, \tau) \neq (\mathsf{M}^\star\|\sigma^\star, \tau^\star)$ for B's forgery $(\mathsf{M}\|\sigma, \tau)$. To see this, let us denote the open query by A that corresponds to B's forgery by $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$. For such a query, we have $(\mathsf{M}^\star, \Sigma^\star) \neq (\mathsf{M}, \Sigma)$ by the restriction posed on A in the CCA-selfless anonymous security game. This in particular implies $(\mathsf{M}^\star\|\sigma^\star, \tau^\star) \neq (\mathsf{M}\|\sigma, \tau)$ as desired, since $\mathsf{ct} = \mathsf{ct}^\star$ and $\mathsf{ovk} = \mathsf{ovk}^\star$. Since B simulates Game 3 unless F occurs, the winning probability of B is exactly ϵ. This concludes the proof of the lemma. □

Lemma 7. *If* ABS *is perfectly private, we have* $\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_5]$.

Proof. We claim that the view of the adversary is unchanged. To see this, we first observe that for a signing query (b, M), we have $C[\mathsf{ovk}, \mathsf{ct}](i_b^\star \| \mathsf{K}_{i_b^\star}) = 1$ and $C[\mathsf{ovk}, \mathsf{ct}](N+1 \| \mathsf{K}_{N+1}) = 1$ for any $\mathsf{ovk} \in \mathsf{OTS.KeyGen}(1^\kappa)$ and $\mathsf{ct} \in \mathsf{SKE.Enc}(\mathsf{K}_{i_b^\star}, i_b^\star \| \mathsf{ovk})$. The former follows from the correctness of SKE whereas the latter is by the definition of $C[\mathsf{ovk}, \mathsf{ct}]$. Therefore, by the perfect privacy of ABS, the following distributions are the same:

$$\{\sigma \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{i_b^\star \| K_{i_b^\star}}, C[\mathsf{ovk}, \mathsf{ct}], \mathsf{M})\} \approx \{\sigma \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{N+1 \| K_{N+1}}, C[\mathsf{ovk}, \mathsf{ct}], \mathsf{M})\}.$$

Note that σ is generated as in the left-hand side in Game 4 and as in the right-hand side in Game 5. Similarly, we can show that the distribution of the challenge signature is unchanged from the previous game. By applying the above argument for each of the signing queries and the challenge query one-by-one, we have $\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_5]$. □

Lemma 8. *If* SKE *is INDr-CCA-secure, we have* $|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]| = \mathrm{negl}(\kappa)$.

Proof. For the sake of contradiction, let us assume that $\epsilon := |\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]|$ is non-negligible. We first observe that since the view of Game 5 and Game 6 are the same when $\mathsf{coin} = 1$, we have $|\Pr[\mathsf{E}_5 | \mathsf{coin} = 0] - \Pr[\mathsf{E}_6 | \mathsf{coin} = 0]| = 2\epsilon$. We then construct an adversary B that breaks the INDr-CCA-security of SKE with non-negligible advantage using A. We give the description of B in the following.

At the beginning of the game, B is given 1^κ and pp from its challenger. Then, B inputs 1^κ to A, who returns $(i_0^\star, i_1^\star) \in [N] \times [N]$. Then, B runs $(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{ABS.Setup}(1^\kappa, 1^{N+1})$ and samples $\mathsf{K}_i \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp})$ and $\mathsf{sk}_{i \| \mathsf{K}_i} \xleftarrow{\$} \mathsf{ABS.KeyGen}(\mathsf{msk}, i, i \| \mathsf{K}_i)$ for $i \in [N+1] \backslash \{i_0^\star\}$. It then sets $\mathsf{gpk} := (\mathsf{pp}, \mathsf{mpk})$ and $\mathsf{gsk}_i := (i, \mathsf{K}_i, \mathsf{sk}_{i \| \mathsf{K}_i})$ for $i \in [N] \backslash \{i_0^\star\}$ and returns gpk and $\{\mathsf{gsk}_i\}_{i \in [N] \backslash \{i_0^\star, i_1^\star\}}$ to A. The queries by A are answered as follows. We note that B implicitly sets $\mathsf{K}_{i_0^\star}$ as the secret key chosen by its challenger and coin is set to be 0 in the following.

- For a signing query (b, M) made by A, B proceeds as follows. If $b = 1$, B can answer the query using $\mathsf{K}_{i_1^\star}$ and $\mathsf{sk}_{N+1 \| \mathsf{K}_{N+1}}$. If $b = 0$, it first samples $(\mathsf{ovk}, \mathsf{osk}) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$ and makes an encryption query for its challenger on $i_0^\star \| \mathsf{ovk}$. Given the ciphertext ct from the challenger, B then runs $\sigma \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{N+1 \| \mathsf{K}_{N+1}}, C[\mathsf{ovk}, \mathsf{ct}], \mathsf{M})$ and $\tau \xleftarrow{\$} \mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M} \| \sigma)$ and returns $\Sigma := (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$ to A.

- For an open query $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$ made by A before the challenge phase, B first runs $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}, \Sigma)$ and returns $\perp$ to A if the output is $\perp$. Otherwise, B computes $d_i := \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct})$ for $i \in [N] \backslash \{i_0^\star\}$. It also makes a decryption query for its challenger on ct and sets the decryption result as $d_{i_0^\star}$. Finally, it returns the smallest index $i \in [N]$ such that $d_i \neq \perp$ to A. If there is not such i, it returns $\perp$.

- For the challenge query $\mathsf{M}^\star$ made by A, B proceeds as follows. It first samples $(\mathsf{ovk}^\star, \mathsf{osk}^\star) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$ and makes the challenge query for its challenger on $i_0^\star \| \mathsf{ovk}^\star$. It is then given $\mathsf{ct}^\star$ from the challenger. Then, it samples $\sigma^\star \xleftarrow{\$} \mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_{N+1 \| K_{N+1}}, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \mathsf{M}^\star)$ and $\tau^\star \xleftarrow{\$} \mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M}^\star \| \sigma^\star)$ and returns $\Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star)$ to A.

- For an open query $(\mathsf{M}, \Sigma = (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau))$ made by A after the challenge phase, B returns $\perp$ to A if $\mathsf{ct} = \mathsf{ct}^\star$. Otherwise, it proceeds as in the case where the query is made before the challenge phase.

Finally, A outputs its guess $\widehat{\mathsf{coin}}$. Then, B outputs 1 if $\widehat{\mathsf{coin}} = 0$ and 0 otherwise.

We observe that B perfectly simulates Game 5 with $\mathsf{coin} = 0$ if the challenge ciphertext is sampled as $\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}_{i_0^\star}, i_0^\star \| \mathsf{ovk}^\star)$ and Game 6 with $\mathsf{coin} = 0$ if it is sampled as $\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.CTSamp}(\mathsf{pp})$. Furthermore, it can be seen that B does not make any prohibited decryption queries. Therefore, the lemma readily follows. □

Lemma 9. *If* SKE *is CCA-secure, we have* $|\Pr[\mathsf{E}_6] - \Pr[\mathsf{E}_7]| = \mathrm{negl}(\kappa)$.

Proof. The proof is almost the same as that of Lemma 8. The main difference is that B simulates the game with $\mathsf{coin} = 1$ (instead of $\mathsf{coin} = 0$) and implicitly sets $\mathsf{K}_{i_1^\star}$ (instead of $\mathsf{K}_{i_0^\star}$) as the secret key chosen by its challenger. □

This completes the proof of the theorem. □

Traceability. The following addresses the traceability of the above GS scheme.

Theorem 6. *If* ABS *is co-selective unforgeable and* SKE *has key robustness, then* GS *constructed above has full traceability.*

Proof. Let us fix a PPT adversary A and consider the full traceability game played between A and a challenger. Let $(\mathsf{M}^\star, \Sigma^\star)$ be a forgery output by A. We define F_1 to be the event that A wins the game and $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star) = \bot$ holds, and F_2 be the event that A wins the game and $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star) = i^\star$ holds for $i^\star$ such that $i^\star \notin \mathcal{T}$. Since both F_1 and F_2 are collectively exhaustive events of a successful forgery, it suffices to prove $\Pr[\mathsf{F}_1] = \mathrm{negl}(\kappa)$ and $\Pr[\mathsf{F}_2] = \mathrm{negl}(\kappa)$.

Lemma 10. *If* ABS *is co-selective unforgeable, we have* $\Pr[\mathsf{F}_1] = \mathrm{negl}(\kappa)$.

Proof. For the sake of the contradiction, let us assume that F_1 happens with non-negligible probability ϵ. We then construct an adversary B that breaks the co-selective unforgeability of ABS with the same probability. The adversary B proceeds as follows.

At the beginning of the game, B is given 1^κ from its challenger. B then samples $\mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa)$ and $\mathsf{K}_i \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp})$ for $i \in [N]$ and submits 1^N, $\{(i, i\|\mathsf{K}_i)\}_{i\in[N]}$, and $\mathcal{S} = [N]$ to its challenger. Then, B receives mpk and $\{\mathsf{sk}_{i\|\mathsf{K}_i}\}_{i\in[N]}$ from the challenger. It then gives 1^κ, $\mathsf{gpk} := (\mathsf{pp}, \mathsf{mpk})$, and $\mathsf{gok} := \{\mathsf{K}_i\}_{i\in[N]}$ to A and keeps $\{\mathsf{gsk}_i := (i, \mathsf{K}_i, \mathsf{sk}_{i\|\mathsf{K}_i})\}_{i\in[N]}$ secret. During the game, A makes signing and corrupt queries. These queries are trivial to handle because B has $\{\mathsf{gsk}_i\}_{i\in[N]}$. In particular, B can handle all signing queries from A without making signing query to its challenger. Eventually, A will output a forgery $(\mathsf{M}^\star, \Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star))$. If $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}^\star, \Sigma^\star) = \top$ and $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star) = \bot$ hold, B outputs $(\mathsf{M}^\star, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \sigma^\star)$ as its forgery. Otherwise, B aborts.

We claim that B wins the game whenever F_1 happens. To prove this, we first observe that $\mathsf{ABS.Vrfy}(\mathsf{mpk}, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \sigma^\star) = \top$ holds because $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}^\star, \Sigma^\star) = \top$. We then show that B has not made any prohibited key query. Namely, we show $C[\mathsf{ovk}^\star, \mathsf{ct}^\star](i\|\mathsf{K}_i) = 0$ for all $i \in [N]$. This follows since otherwise we have $\mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot$ for some i, which contradicts $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star) = \bot$. We also note that B has not made any signing query. Since B's simulation is perfect, we can conclude that B wins the game with probability ϵ. This concludes the proof of the lemma. □

Lemma 11. *If* ABS *is co-selective unforgeable and* SKE *has key robustness, we have* $\Pr[\mathsf{F}_2] = \mathrm{negl}(\kappa)$.

Proof. For the sake of the contradiction, let us assume that F_2 happens with non-negligible probability ϵ. We then construct an adversary B that breaks the co-selective unforgeability of ABS with non-negligible probability. We show this by considering the following sequence of games. In the following, let E_i denote the probability that F_2 occurs and the challenger does not abort in $\mathsf{Game}\ i$.

$\mathsf{Game}\ 0$**:** We define $\mathsf{Game}\ 0$ as the ordinary full traceability game between A and the challenger. By assumption, we have $\Pr[\mathsf{E}_0] = \epsilon$.

$\mathsf{Game}\ 1$**:** In this game, the challenger samples $j^\star \xleftarrow{\$} [N]$ at the beginning of the game and aborts if $j^\star \neq i^\star$ at the end of the game. Since the view of A is independent from $j^\star$ and $\mathsf{GS.Open}$ does not output any symbol outside $[N] \cup \{\perp\}$, we have $\Pr[\mathsf{E}_1] = \epsilon/N$.

$\mathsf{Game}\ 2$**:** In this game, the challenger aborts the game as soon as $j^\star \neq i^\star$ turns out to be true. Namely, it aborts if A makes a corruption query for $j^\star$, or $i^\star$ defined at the end of the game does not equal to $j^\star$. Since this is only a conceptual change, we have $\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3]$.

$\mathsf{Game}\ 3$**:** In this game, we change the previous game so that the challenger aborts at the end of the game if $|\{i \in [N] : \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \perp\}| \neq 1$ for $(\mathsf{M}^\star, \Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star))$ output by A as the forgery. We claim that the probability that F_2 and $|\{i \in [N] : \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \perp\}| \neq 1$ occur at the same time is negligibly small. Note that by the definition of $\mathsf{GS.Open}$, F_2 implies $\mathsf{SKE.Dec}(\mathsf{K}_{i^\star}, \mathsf{ct}^\star) \neq \perp$ for $i^\star \in [N]$. We therefore have $|\{i \in [N] : \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \perp\}| \geq 2$. However, the probability of this occurring is bounded by

$$
\begin{aligned}
& \Pr\left[\ |\{i \in [N] : \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \perp\}| \geq 2\ \right] \\
\leq\ & \Pr\left[\begin{array}{c} \mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa),\ \mathsf{K}_j \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp}) \text{ for } j \in [N] :\ \exists \mathsf{ct}^\star \in \{0,1\}^*,\ \exists i, i^\star \in [N] \\ \text{s.t. } i \neq i^\star \wedge\ \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \perp \wedge\ \mathsf{SKE.Dec}(\mathsf{K}_{i^\star}, \mathsf{ct}^\star) \neq \perp \end{array}\right] \\
\leq\ & \sum_{i, i^\star \in [N] \text{ s.t. } i \neq i^\star} \Pr\left[\begin{array}{c} \mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa),\ \mathsf{K}_i, \mathsf{K}_{i^\star} \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp})\ :\ \exists \mathsf{ct}^\star \in \{0,1\}^* \\ \text{s.t. } \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \perp \wedge\ \mathsf{SKE.Dec}(\mathsf{K}_{i^\star}, \mathsf{ct}^\star) \neq \perp \end{array}\right] \\
\leq\ & N(N-1)/2 \cdot \mathrm{negl}(\kappa) \\
=\ & \mathrm{negl}(\kappa),
\end{aligned}
$$

where the second inequality is by the union bound and the third inequality is by the key robustness of SKE. Therefore, we have $|\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_3]| = \mathrm{negl}(\kappa)$.

We then replace the challenger in $\mathsf{Game}\ 3$ with an adversary B against the co-selective unforgeability of ABS with advantage $\Pr[\mathsf{E}_3]$. The adversary B proceeds as follows.

At the beginning of the game, B is given 1^κ from its challenger. Then, B chooses its guess $j^\star \xleftarrow{\$} [N]$ for $i^\star$, samples $\mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa)$ and $\mathsf{K}_i \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp})$ for $i \in [N]$, and sends 1^N, $\{(i, i\|\mathsf{K}_i)\}_{i \in [N]}$, and $\mathcal{S} = [N] \backslash \{j^\star\}$ to the challenger. Then, B receives mpk and $\{\mathsf{sk}_{i\|\mathsf{K}_i}\}_{i \in [N] \backslash \{j^\star\}}$ from the challenger. It then sets $\mathsf{gpk} := (\mathsf{pp}, \mathsf{mpk})$, $\mathsf{gsk}_i := (i, \mathsf{K}_i, \mathsf{sk}_{i\|\mathsf{K}_i})$ for $i \in [N] \backslash \{j^\star\}$, and $\mathsf{gok} := \{\mathsf{K}_i\}_{i \in [N]}$ and gives 1^κ, gpk, and gok to A. During the game, A makes two kinds of queries. B answers the queries as follows.

- When A makes a corrupt query for $i \in [N]$, B proceeds as follows. If $i = j^\star$, B aborts. Otherwise, it gives gsk_i to A.

- When A makes a signing query for (i, M), B answers the query using gsk_i if $i \neq j^\star$. If $i = j^\star$, B first samples $(\mathsf{ovk}, \mathsf{osk}) \xleftarrow{\$} \mathsf{OTS.KeyGen}(1^\kappa)$ and computes $\mathsf{ct} \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}_{j^\star}, j^\star \| \mathsf{ovk})$. It then makes a signing query $(\mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], j^\star)$ to its challenger, who returns σ to B. Then, it runs $\mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M} \| \sigma) \to \tau$ and returns $\Sigma := (\mathsf{ovk}, \mathsf{ct}, \sigma, \tau)$ to A.

Eventually, A will output a forgery $(\mathsf{M}^\star, \Sigma^\star = (\mathsf{ovk}^\star, \mathsf{ct}^\star, \sigma^\star, \tau^\star))$. If either of $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}^\star, \Sigma^\star) = \top$ or $i^\star = j^\star$ does not hold, where $i^\star := \mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star)$, B aborts. It also aborts if $|\{i \in [N] : \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot\}| \neq 1$. Otherwise, B outputs $(\mathsf{M}^\star, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \sigma^\star)$ as its forgery.

We claim that B wins the game whenever E_3 occurs. To see this, we first observe that we have $\mathsf{ABS.Vrfy}(\mathsf{mpk}, C[\mathsf{ovk}^\star, \mathsf{ct}^\star], \sigma^\star) = \top$ by $\mathsf{GS.Vrfy}(\mathsf{gpk}, \mathsf{M}^\star, \Sigma^\star) = \top$. We then prove that B has never made prohibited corrupt queries. Namely, we show $C[\mathsf{ovk}^\star, \mathsf{ct}^\star](i \| \mathsf{K}_i) = 0$ for all $i \in [N] \backslash \{i^\star\}$. This follows since we have $|\{i \in [N] : \mathsf{SKE.Dec}(\mathsf{K}_i, \mathsf{ct}^\star) \neq \bot\}| = 1$ and $\mathsf{SKE.Dec}(\mathsf{K}_{i^\star}, \mathsf{ct}^\star) \neq \bot$, where the latter follows from $\mathsf{GS.Open}(\mathsf{gpk}, \mathsf{gok}, \mathsf{M}^\star, \Sigma^\star) = i^\star$. Finally, we show that B has never made prohibited signing queries. Recall that B has only made signing queries of the form $(\mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}], i^\star)$ and all such queries are made in order to answer the signing query $(i^\star, \mathsf{M})$ made by A. Because A has won the game, we have $\mathsf{M}^\star \neq \mathsf{M}$, which implies $(\mathsf{M}^\star, C[\mathsf{ovk}^\star, \mathsf{ct}^\star]) \neq (\mathsf{M}, C[\mathsf{ovk}, \mathsf{ct}])$ as desired. Since B simulates $\mathsf{Game}\ 3$ perfectly, we have that the winning probability of B is exactly $\Pr[\mathsf{E}_3]$. This concludes the proof of the lemma. □

This completes the proof of the theorem. □

5 Construction of Indexed ABS from Lattices

In this section, we give a new construction of indexed ABS scheme from the SIS assumption. Combined with an appropriate SKE scheme and OTS scheme, we can instantiate the generic construction of GS in Sec. 4 to obtain the first lattice-based GS scheme in the standard model. We refer Sec. 7 to more discussions.

5.1 Preliminaries on Lattices

Here, we recall some facts on lattices that are needed for the exposition of our construction. Throughout this section, n, m, and q are integers such that $n = \mathrm{poly}(\kappa)$ and $m \geq n\lceil \log q \rceil$. In the following, let $\mathsf{SampZ}(\gamma)$ be a sampling algorithm for the truncated discrete Gaussian distribution over $\mathbb{Z}$ with parameter $\gamma > 0$ whose support is restricted to $z \in \mathbb{Z}$ such that $|z| \leq \sqrt{n}\gamma$.[7]

Definition 2 (The SIS Assumption)**.** *Let n, m, q, β be integer parameters. We say that the $\mathsf{SIS}(n, m, q, \beta)$ hardness assumption holds if for any PPT adversaries A we have*

$$\Pr\left[\mathbf{A} \cdot \mathbf{z} = \mathbf{0} \ \wedge \ \|\mathbf{z}\|_\infty \leq \beta(\kappa) \ \wedge \ \mathbf{z} \neq \mathbf{0} \ : \ \mathbf{A} \xleftarrow{\$} \mathbb{Z}_{q(\kappa)}^{n(\kappa) \times m(\kappa)}, \ \mathbf{z} \leftarrow \mathsf{A}(1^\kappa, \mathbf{A}) \right] \leq \mathrm{negl}(\kappa).$$

We also say that the $\mathsf{SIS}(n, m, q, \beta)$ problem is subexponentially hard if the above probability is bounded by $2^{-O(n^\epsilon)} \cdot \mathrm{negl}(\kappa)$ for some constant $0 < \epsilon < 1$.

For any $n = \mathrm{poly}(\kappa)$, any $m = \mathrm{poly}(n)$, any $\beta(n) > 0$, and $q \geq \beta\sqrt{n} \cdot \omega(\log n)$, it is known that the $\mathsf{SIS}(n, m, q, \beta)$ problem is as hard as certain worst case lattice problems with approximation

[7] During construction, we fix n and consider this very weak bound for one-dimensional discrete Gaussian samples for simplicity of analysis.

factor $\beta(n) \cdot \mathrm{poly}(n)$. We abuse the term and refer to $\mathsf{SIS}(n, m, q, \beta)$ with $\beta \leq \mathrm{poly}(\kappa)$ as the SIS problem with polynomial approximation factor.

Trapdoors. Let us consider a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$. For all $\mathbf{V} \in \mathbb{Z}_q^{n \times m'}$, we let $\mathbf{A}_\gamma^{-1}(\mathbf{V})$ be an output distribution of $\mathsf{SampZ}(\gamma)^{m \times m'}$ conditioned on $\mathbf{A} \cdot \mathbf{A}_\gamma^{-1}(\mathbf{V}) = \mathbf{V}$. A γ-trapdoor for $\mathbf{A}$ is a trapdoor that enables one to sample from the distribution $\mathbf{A}_\gamma^{-1}(\mathbf{V})$ in time $\mathrm{poly}(n, m, m', \log q)$, for any $\mathbf{V}$. We slightly overload notation and denote a γ-trapdoor for $\mathbf{A}$ by $\mathbf{A}_\gamma^{-1}$. We also define the special gadget matrix $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ as the matrix obtained by padding $\mathbf{I}_n \otimes (1, 2, 4, 8, \ldots, 2^{\lceil \log q \rceil})$ with zero-columns. The following properties had been established in a long sequence of works [GPV08, CHKP10, ABB10a, ABB10b, MP12, BLP+13].

Lemma 12 (Properties of Trapdoors)**.** *Lattice trapdoors exhibit the following properties.*

1. *Given $\mathbf{A}_\gamma^{-1}$, one can obtain $\mathbf{A}_{\gamma'}^{-1}$ for any $\gamma' \geq \gamma$.*
2. *Given $\mathbf{A}_\gamma^{-1}$, one can obtain $[\mathbf{A}\|\mathbf{B}]_\gamma^{-1}$ and $[\mathbf{B}\|\mathbf{A}]_\gamma^{-1}$ for any $\mathbf{B}$.*
3. *For all $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{R} \in \mathbb{Z}^{m \times m}$, one can obtain $[\mathbf{AR} + \mathbf{G}\|\mathbf{A}]_\gamma^{-1}$ for $\gamma = m \cdot \|\mathbf{R}\|_\infty \cdot \omega(\sqrt{\log m})$.*
4. *There exists an efficient procedure $\mathsf{TrapGen}(1^n, 1^m, q)$ that outputs $(\mathbf{A}, \mathbf{A}_{\gamma_0}^{-1})$ where $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ for some $m = O(n \log q)$ and is 2^{-n}-close to uniform, where $\gamma_0 = \omega(\sqrt{n \log q \log m})$. Furthermore, the following distributions are statistically close for any $m' = \mathrm{poly}(\kappa)$ and $\gamma \geq \gamma_0$:*

$$(\mathbf{A}, \mathbf{A}_\gamma^{-1}, \mathbf{U}, \mathbf{V}) \overset{\mathrm{stat}}{\approx} (\mathbf{A}, \mathbf{A}_\gamma^{-1}, \mathbf{U}', \mathbf{V}'),$$

where $\mathbf{U} \xleftarrow{\$} \mathsf{SampZ}(\gamma)^{m \times m'}$, $\mathbf{V} = \mathbf{AU}$, $\mathbf{V}' \xleftarrow{\$} \mathbb{Z}_q^{n \times m'}$, and $\mathbf{U}' \xleftarrow{\$} \mathbf{A}_\gamma^{-1}(\mathbf{V}')$.

Lemma 13 (Fully Homomorphic Computation [GV15])**.** *There exists a pair of deterministic algorithms $(\mathsf{PubEval}, \mathsf{TrapEval})$ with the following properties.*

- $\mathsf{PubEval}(\vec{\mathbf{B}}, F) \to \mathbf{B}_F$. *Here, $\vec{\mathbf{B}} = [\mathbf{B}_1\| \cdots \|\mathbf{B}_k] \in (\mathbb{Z}_q^{n \times m})^k$ and $F : \{0,1\}^k \to \{0,1\}$ is a circuit.*
- $\mathsf{TrapEval}(\vec{\mathbf{R}}, F, x) \to \mathbf{R}_{F,x}$. *Here, $\vec{\mathbf{R}} = [\mathbf{R}_1\| \ldots \|\mathbf{R}_k] \in (\mathbb{Z}_q^{n \times m})^k$, $\|\mathbf{R}_i\|_\infty \leq \delta$ for $i \in [k]$, $x \in \{0,1\}^k$, and $F : \{0,1\}^k \to \{0,1\}$ is a circuit with depth d. We have*

$$\mathsf{PubEval}(\mathbf{A}\vec{\mathbf{R}} + x \otimes \mathbf{G}) = \mathbf{AR}_{F,x} + F(x)\mathbf{G}$$

where we denote $[x_1\mathbf{G}\| \cdots \|x_k\mathbf{G}]$ by $x \otimes \mathbf{G}$. Furthermore, we have

$$\|\mathbf{R}_{F,x}\|_\infty \leq \delta \cdot m \cdot 2^{O(d)}.$$

- *The running time of $(\mathsf{PubEval}, \mathsf{TrapEval})$ is bounded by $\mathrm{poly}(k, n, m, 2^d, \log q)$.*

The above algorithms are taken from [GV15], which is a variant of a similar algorithms proposed by Boneh et al. [BGG+14]. The algorithms in [BGG+14] work for any polynomial-sized circuit F, but $\|\mathbf{R}_{F,x}\|_\infty$ becomes super-polynomial even if the depth of the circuit is shallow (i.e., logarithmic depth). On the other hand, the above algorithm runs in polynomial time only when F is of logarithmic depth, but $\|\mathbf{R}_{F,x}\|_\infty$ can be polynomially bounded. The latter property is useful since our main focus is on the constructions of GS schemes from the SIS assumption with polynomial approximation factors.

Min-Entropy and Leftover Hash Lemma. We define the min-entropy of a random variable X as

$$\mathbf{H}_\infty(X) = -\log(\max_x \Pr[X = x])$$

and the *average min-entropy* of X conditioned on another random variable Y as

$$\mathbf{H}_\infty(X|Y) = -\log \mathbb{E}_{y \xleftarrow{\$} Y}[2^{-H_\infty(X|Y=y)}],$$

where $\mathbb{E}$ denotes the expectation. The following lemma will be used in the security proof.

Lemma 14 (Generalized Leftover Hash Lemma [DRS04])**.** *Let X and Y be arbitrarily random variables where the support of Y lies in $\mathcal{Y}$. Then $\mathbf{H}_\infty(X|Y) \geq \mathbf{H}_\infty(X) - \log(|\mathcal{Y}|)$.*

5.2 Construction

Here, we show our construction of indexed ABS. The scheme satisfies no-signing-query unforgeability. By applying the conversion in Sec. 3.2 to the scheme, we can obtain a scheme with co-selective unforgeability. Note that the signing and the verification algorithm below ignore the input message M. This is not a problem because the no-signing-query security does not require non-malleability with respect to the message (See also Remark 4).

We denote the circuit class that is dealt with by the scheme by $\{\mathcal{F}_\kappa\}_\kappa$, where $\mathcal{F}_\kappa$ is a set of circuits F such that $F : \{0,1\}^{k(\kappa)} \to \{0,1\}$ and with depth at most $d_\mathcal{F} = O(\log \kappa)$.

$\mathsf{ABS.Setup}(1^\kappa, 1^N)$: On input 1^κ and 1^N, it sets the parameters n, m, q, γ_0, γ, and β as specified later in this section, where q is a prime number. Then, it picks random matrices $\mathbf{B}_j^{(i)} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ for $i \in [N], j \in [k]$. We denote $\vec{\mathbf{B}}^{(i)} = [\mathbf{B}_1^{(i)} \| \cdots \| \mathbf{B}_k^{(i)}]$. It also picks $(\mathbf{A}, \mathbf{A}_{\gamma_0}^{-1}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q)$ such that $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a random vector $\mathbf{r} \xleftarrow{\$} \{0,1\}^m$. It then computes $\mathbf{u} := \mathbf{A}\mathbf{r} \in \mathbb{Z}_q^n$. It finally outputs

$$\mathsf{mpk} = \left(\mathbf{A},\ \{\vec{\mathbf{B}}^{(i)}\}_{i\in[N]},\ \mathbf{u},\ \right) \qquad \text{and} \qquad \mathsf{msk} = \left(\mathbf{A}_{\gamma_0}^{-1},\ \{\vec{\mathbf{B}}^{(i)}\}_{i\in[N]} \right).$$

$\mathsf{ABS.KeyGen}(\mathsf{msk}, i, x)$: On input $\mathsf{msk} = (\mathbf{A}_{\gamma_0}^{-1}, \{\vec{\mathbf{B}}^{(i)}\}_{i\in[N]})$, $i \in [N]$, and $x \in \{0,1\}^k$, it samples

$$\vec{\mathbf{R}}^{(i)} \xleftarrow{\$} \mathbf{A}_{\gamma_0}^{-1}\big(\vec{\mathbf{B}}^{(i)} - x \otimes \mathbf{G}\big) \qquad \text{where} \qquad \vec{\mathbf{R}}^{(i)} \in \mathbb{Z}^{m\times mk}$$

using $\mathbf{A}_{\gamma_0}^{-1}$. Note that $\vec{\mathbf{B}}^{(i)} = \mathbf{A}\vec{\mathbf{R}}^{(i)} + x \otimes \mathbf{G}$ and $\|\vec{\mathbf{R}}^{(i)}\|_\infty \leq \gamma_0\sqrt{n}$ holds by the definition of the distribution $\mathbf{A}_{\gamma_0}^{-1}(\vec{\mathbf{B}}^{(i)} - x \otimes \mathbf{G})$. It then outputs $\mathsf{sk}_x := (i, \vec{\mathbf{R}}^{(i)})$.

$\mathsf{ABS.Sign}(\mathsf{mpk}, \mathsf{sk}_x, \mathsf{M}, F)$: It outputs $\perp$ if $\mathsf{M} \notin \mathcal{M}_\kappa$, $F \notin \mathcal{F}$, or $F(x) = 0$. Otherwise, it first parses $\mathsf{sk}_x \to (i, \vec{\mathbf{R}}^{(i)})$. It then computes $\mathbf{B}_F^{(i)} := \mathsf{PubEval}(\vec{\mathbf{B}}^{(i)}, F)$ and $\mathbf{R}_{F,x}^{(i)} := \mathsf{TrapEval}(\vec{\mathbf{R}}^{(i)}, F, x)$ such that $\|\mathbf{R}_{F,x}^{(i)}\|_\infty \leq \gamma$. By Lemma 13 and since $F(x) = 1$, we have $\mathbf{B}_F^{(i)} = \mathbf{A}\mathbf{R}_{F,x}^{(i)} + \mathbf{G}$. It then computes $[\mathbf{A}\|\mathbf{B}_F^{(i)}]_\beta^{-1}$ from $\mathbf{R}_{F,x}^{(i)}$ (see Item 3 in Lemma 12) and further computes

$$\left[\mathbf{A}\|\mathbf{B}_F^{(1)}\| \cdots \|\mathbf{B}_F^{(N)}\right]_\beta^{-1}$$

from $[\mathbf{A}\|\mathbf{B}_F^{(i)}]_\beta^{-1}$ (see Item 2 in Lemma 12). Finally, it samples

$$\mathbf{e} \xleftarrow{\$} [\mathbf{A}\|\mathbf{B}_F^{(1)}\| \cdots \|\mathbf{B}_F^{(N)}]_\beta^{-1}(\mathbf{u})$$

and outputs the signature $\sigma := \mathbf{e} \in \mathbb{Z}^{m(N+1)}$.

$\mathsf{ABS.Vrfy}(\mathsf{mpk}, \mathsf{M}, \sigma, F)$: It outputs $\bot$ if $F \notin \mathcal{F}$ or $\sigma = \mathbf{e} \notin \mathbb{Z}^{m(N+1)}$. Otherwise, it first computes $\mathbf{B}_F^{(i)} = \mathsf{PubEval}(F, \vec{\mathbf{B}}^{(i)})$ for $i \in [N]$. It then checks whether $\|\mathbf{e}\|_\infty \leq \sqrt{n}\beta$ and

$$\left[\mathbf{A} \| \mathbf{B}_F^{(1)} \| \cdots \| \mathbf{B}_F^{(N)}\right] \mathbf{e} = \mathbf{u}. \tag{3}$$

If they hold, it outputs $\top$ and otherwise $\bot$.

Correctness. The correctness of the scheme can be seen by observing that Eq. (3) and $\|\mathbf{e}\|_\infty \leq \sqrt{n}\beta$ follow from the definition of the distribution $[\mathbf{A} \| \mathbf{B}_F^{(1)} \| \cdots \| \mathbf{B}_F^{(N)}]_\beta^{-1}(\mathbf{u})$ from which $\mathbf{e}$ is sampled.

Parameter Selection. As long as the maximum depth of the circuit class $\mathcal{F}_\kappa$ is bounded by $O(\log \kappa)$, we can set all of n, m, γ_0, γ, β, and q to be polynomial in κ. Notably, this allows us to reduce the security of the scheme to $\mathsf{SIS}(n, m, q, \beta_{\mathsf{SIS}})$ with $\beta_{\mathsf{SIS}} = \mathrm{poly}(\kappa)$. We refer to Appendix B for the precise requirements for these parameters and a concrete selection.

Remark 6 (A variant that can deal with polynomial depth circuits.)**.** *We remark that if we assume the SIS assumption with subexponential approximation factor, then the above construction yields an indexed ABS that can deal with any (bounded) polynomial $d_{\mathcal{F}}$. The only modifications is to use the evaluation algorithms of [BGG+14] rather than [GV15] and change the parameters appropriately (See also the discussion right after Lemma 13). In particular, we have to take $q = m^{O(d_{\mathcal{F}})}$.*

5.3 Security Proofs

Theorem 7. *Our ABS scheme is perfectly private.*

Proof. It can be seen that the signature $\sigma = \mathbf{e}$ for (F, M) is chosen from the distribution $[\mathbf{A} \| \mathbf{B}_F^{(1)} \| \cdots \| \mathbf{B}_F^{(N)}]_\beta^{-1}(\mathbf{u})$, which only depends on mpk and F. The theorem readily follows. □

Theorem 8. *Our ABS scheme satisfies no-signing-query unforgeability assuming $\mathsf{SIS}(n, m, q, \beta_{\mathrm{SIS}})$ is hard.*

Proof. To prove the theorem, it suffices to show that for any PPT adversary A against the no-signing-query unforgeability of our ABS scheme with advantage ϵ, we can construct a PPT algorithm B that solves the SIS problem with probability at least $\epsilon - \mathrm{negl}(\kappa)$. Therefore, by assuming the hardness of the SIS problem, we conclude that ϵ is negligible. We show this by considering the following sequence of games. Below, let E_i denote the probability that A is successful in $\mathsf{Game}\ i$.

$\mathsf{Game}\ 0$**:** We define $\mathsf{Game}\ 0$ as an experiment between the challenger and the adversary A. Here, we have $\Pr[\mathsf{E}_0] = \epsilon$.

$\mathsf{Game}\ 1$**:** In this game, we change the way $\{\vec{\mathbf{B}}^{(i)}\}_{i \in [N]}$ are chosen. At the beginning of the game, the challenger receives 1^N, $\{(i, x^{(i)})\}_{i \in [N]}$, and $\mathcal{S} \subseteq [N]$ from A. Then, the challenger samples $\vec{\mathbf{R}}^{(i)} \xleftarrow{\$} \mathsf{SampZ}(\gamma_0)^{m \times mk}$ for $i \in [N]$ and sets the matrices as

$$\vec{\mathbf{B}}^{(i)} = \begin{cases} \mathbf{A}\vec{\mathbf{R}}^{(i)} + x^{(i)} \otimes \mathbf{G} & \text{if } i \in \mathcal{S} \\ \mathbf{A}\vec{\mathbf{R}}^{(i)} + x^{(i_{\min})} \otimes \mathbf{G} & \text{if } i \notin \mathcal{S} \end{cases}, \quad \text{where} \quad i_{\min} := \min\{\ i \mid i \in \mathcal{S}\ \}. \tag{4}$$

We remark that we have $\mathcal{S} \neq \emptyset$ by the definition of the game and thus $i_{\min}$ above is well-defined. Then the challenger gives $\{\mathsf{sk}_{x^{(i)}} := (i, \vec{\mathbf{R}}^{(i)})\}_{i \in \mathcal{S}}$ to A. By Item 4 of Lemma 12, the

distribution of $(\mathbf{A}, \{\vec{\mathbf{B}}^{(i)}\}_{i\in[N]}, \{\vec{\mathbf{R}}^{(i)}\}_{i\in\mathcal{S}})$ in both games are statistically close. Therefore, we have $\Pr[\mathsf{E}_1] \geq \Pr[\mathsf{E}_0] - \mathrm{negl}(\kappa)$. Note that in this game, the challenger does not use the trapdoor $\mathbf{A}_{\gamma_0}^{-1}$ any more.

Game 2: In this game, we change the way $\mathbf{A}$ is sampled. Namely, the challenger samples $\mathbf{A}$ as $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ instead of sampling it with the trapdoor as $(\mathbf{A}, \mathbf{A}_{\gamma_0}^{-1}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q)$. By Item 4 of Lemma 12, the distribution of $\mathbf{A}$ in this game is statistically close to that of the previous game. Therefore, we have $\Pr[\mathsf{E}_2] \geq \Pr[\mathsf{E}_1] - \mathrm{negl}(\kappa)$.

Finally, we replace the challenger in Game 2 with an algorithm B that solves the SIS problem with probability $\Pr[\mathsf{E}_2] - \mathrm{negl}(\kappa)$, which is at least $\epsilon - \mathrm{negl}(\kappa)$ combining the above together. We give the description of B in the following.

Initial Phase: The adversary B is given $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ as the problem instance of the SIS problem. Then, the adversary B runs A on input 1^κ and is given 1^N, $\{(i, x^{(i)})\}_{i\in[N]}$, and $\mathcal{S} \subseteq [N]$ from A.

Setup: The challenger picks $\vec{\mathbf{R}}^{(i)} \xleftarrow{\$} \mathsf{SampZ}(\gamma_0)^{m\times mk}$ for $i \in [N]$ and sets $\{\vec{\mathbf{B}}^{(i)}\}_{i\in[N]}$ as Eq. (4). It also samples a random vector $\mathbf{r} \xleftarrow{\$} \{0,1\}^m$ and sets $\mathbf{u} = \mathbf{A}\mathbf{r}$. B then gives $\mathsf{mpk} = (\mathbf{A}, \{\vec{\mathbf{B}}^{(i)}\}_{i\in[N]}, \mathbf{u})$ and $\{\mathsf{sk}_{x^{(i)}} := (i, \vec{\mathbf{R}}^{(i)})\}_{i\in\mathcal{S}}$ to A.

Forgery: Eventually, A outputs $(\mathsf{M}^\star, F^\star, \sigma^\star = \mathbf{e})$ as the forgery. B first checks whether A won the game and aborts otherwise. Then, B computes

$$\mathbf{S}^{(i)} := \begin{cases} \mathsf{TrapEval}(\vec{\mathbf{R}}^{(i)}, F^\star, x^{(i)}) & \text{if } i \in \mathcal{S} \\ \mathsf{TrapEval}(\vec{\mathbf{R}}^{(i)}, F^\star, x^{(i_{\min})}) & \text{if } i \notin \mathcal{S} \end{cases}$$

for $i \in [N]$. Finally, B sets $\mathbf{S} := [\mathbf{I}_m \| \mathbf{S}^{(1)} \| \cdots \| \mathbf{S}^{(N)}]$ and outputs $\mathbf{z} := \mathbf{S}\mathbf{e} - \mathbf{r}$ as the solution to the SIS problem.

Analysis: We prove that B succeeds in solving the SIS problem with a probability that is negligibly close to $\Pr[\mathsf{E}_2]$. It can easily be seen that B perfectly simulates Game 2. We first prove $\mathbf{A}\mathbf{z} = \mathbf{0}$ and $\|\mathbf{z}\|_\infty \leq Nmn\beta\gamma$. From Eq.(4) and $F^\star(x^{(i)}) = 0$ for $i \in \mathcal{S}$, which in particular implies $F^\star(x^{(i_{\min})}) = 0$, we have $\mathbf{B}_{F^\star}^{(i)} = \mathbf{A}\mathbf{S}^{(i)}$ for $i \in [N]$ by Lemma 13. By the same lemma and our choice of γ, we also have $\|\mathbf{S}^{(i)}\|_\infty \leq \gamma$. Assuming A won the game, we have $[\mathbf{A}\|\mathbf{B}_{F^\star}^{(1)}\| \cdots \|\mathbf{B}_{F^\star}^{(N)}]\mathbf{e} = \mathbf{u}$ and $\|\mathbf{e}\|_\infty \leq \sqrt{n}\beta$. This implies that we have

$$[\mathbf{A}\|\mathbf{B}_{F^\star}^{(1)}\| \cdots \|\mathbf{B}_{F^\star}^{(N)}] \cdot \mathbf{e} = \mathbf{A} \cdot \underbrace{[\mathbf{I}_m \| \mathbf{S}^{(1)} \| \cdots \| \mathbf{S}^{(N)}]}_{=\mathbf{S}} \cdot \mathbf{e} = \mathbf{u},$$

which implies $\mathbf{A}\mathbf{z} = \mathbf{A}(\mathbf{S}\mathbf{e} - \mathbf{r}) = \mathbf{u} - \mathbf{u} = \mathbf{0}$. Furthermore, we have

$$\|\mathbf{z}\|_\infty = \|\mathbf{r} - \mathbf{S}\mathbf{e}\|_\infty \leq \|\mathbf{r}\|_\infty + (N+1)m \cdot \|\mathbf{S}\|_\infty \cdot \|\mathbf{e}\|_\infty \leq 1 + (N+1)m\beta\gamma\sqrt{n} \leq Nmn\beta\gamma$$

as desired.

It remains to show $\mathbf{z} = \mathbf{r} - \mathbf{S}\mathbf{e} \neq \mathbf{0}$ with overwhelming probability. We can prove this even if A is computationally unbounded and is given all the randomness during the game except for $\mathbf{r}$. The proof is by an entropy argument. We have

$$\mathbf{H}_\infty(\mathbf{r}|\mathbf{A}\mathbf{r}) \geq m - n\log q \geq \omega(\log\kappa)$$

by Lemma 14. This implies $\Pr[\mathbf{r} = \mathbf{S}\mathbf{e}] \leq 2^{-\omega(\log\kappa)} \leq \mathrm{negl}(\kappa)$ since $\mathbf{r}$ is chosen uniformly at random and independently from any other variables during the game. We therefore have that B successfully solves the SIS problem with probability $\Pr[\mathsf{E}_2] - \mathrm{negl}(\kappa)$. This concludes the proof of the theorem. □

6 Instantiating SKE

Here, we discuss how to instantiate the SKE required for the generic construction of GS in Sec. 4. Since this can be done by a combination of known results and standard techniques, we only give a high level overview here and refer to Appendix C for more details. We require the SKE to be INDr-CCA secure and to have key robustness. We also want the decryption circuit of the SKE to be as shallow as possible. In particular, if the depth of the decryption circuit is $O(\log \kappa)$, we can instantiate our indexed ABS scheme in Sec. 5.2 with polynomial-sized modulus when combining it with the SKE to obtain a GS scheme, which is desirable both from security and efficiency reasons. On the other hand, if the depth of the decryption circuit is $\mathrm{poly}(\kappa)$, the modulus size for the indexed ABS becomes subexponential (See also Remark 6).

To obtain such an SKE scheme, we follow the MAC-then-Encrypt paradigm and show a generic construction of such an SKE from another SKE and a MAC. For the latter SKE, we require INDr-CPA security and key robustness. For the MAC, we require strong unforgeability. We also want the depth of the decryption circuit of the (ingredient) SKE and the verification circuit of the MAC to be as shallow as possible since the depth of the decryption circuit of the resulting SKE will be their sum. Although an insecure example of the MAC-then-Encrypt approach is known [BN00], we avoid the pitfall by authenticating a part of the ciphertext in addition to the plaintext using the MAC. We also note that the Encrypt-then-MAC approach may not work in our setting because the MAC part may reveal the information about the user and destroy the INDr-CCA security (in particular, anonymity) of the resulting SKE scheme.

In the following, we discuss various ways of instantiating the SKE from different assumptions.

Instantiation from LWE. We discuss how to instantiate the inner SKE and MAC from the LWE assumption. For the SKE, we use a secret key variant of the Regev encryption scheme [Reg05], where we pad the message with zeroes before encrypting it and the decryption algorithm returns $\perp$ to a ciphertext that does not conform to this format. The padding makes the ciphertext somewhat redundant, and due to this redundancy, we can prove key robustness of the scheme by a standard counting argument. The INDr-CPA security of the scheme is proven from the LWE assumption by a straightforward reduction. The decryption circuit of the scheme can be implemented by an $O(\log \kappa)$-depth circuit, since the decryption algorithm only involves basic algebraic operations such as the computation of an inner-product, modulo reduction, and comparison, all of which are known to be in $\mathbf{NC}^1$. We then discuss how to instantiate the MAC. We need the MAC scheme to have strong unforgeability and a decryption circuit with $O(\log \kappa)$-depth. To obtain such a scheme, we downgrade the (public key) signature scheme proposed by Micciancio and Peikert [MP12] to a MAC scheme. Since the scheme satisfies strong unforgeability as a signature scheme, it is trivial to see that the scheme is strongly unforgeable as a MAC as well. The verification circuit of the scheme can be implemented by an $O(\log \kappa)$-depth circuit, since the verification algorithm only involves basic algebraic operations, similarly to the decryption algorithm of the above SKE.

Finally, we remark that another way of obtaining the SKE required for the generic construction in Sec. 4 from lattices may be to downgrade the CCA-secure public key encryption scheme by Micciancio and Peikert [MP12] to an SKE scheme. However, this approach requires the LWE assumption with larger approximation factor than our approach described above.

Instantiation from LPN. We discuss how to instantiate the SKE from the LPN assumption with constant noise rate. As for the SKE, we use the analogue of the LWE-based SKE we discussed in the previous paragraph. The difference here is that in the LPN setting, we encode the message by the repetition code. The key robustness of the scheme can be shown by a similar combinatorial

argument to the LWE setting. The decryption circuit for the SKE scheme can be implemented in depth $O(\log \kappa)$ because it only involves simple algebraic operations. As for the MAC, we use the construction in [DKPW12] from the LPN assumption, which is an improvement of [KPV+11]. Though it is not explicitly mentioned in [DKPW12], the MAC is strongly unforgeable as observed in [AHM+14]. The verification algorithm of the MAC can be implemented by a circuit with depth $O(\log \kappa)$, since the verification algorithm only involves simple algebraic operations and the computation of a pair-wise independent hash which can also be implemented by $O(\log \kappa)$-depth circuits.

Instantiation from SIS. Finally, we discuss how to instantiate the SKE from the SIS assumption. We instantiate the SKE with the key robust SKE proposed by Ishida [Ish18], which relies on one-way functions. Note that since the SIS assumption implies the existence of one-way functions, the SKE can be instantiated by the same assumption. We also instantiate the MAC by downgrading the strongly unforgeable (public key) signature scheme by Rückert [Rüc10]. Note that in this instantiation, we can no longer bound the depth of the decryption circuit of the resulting SKE scheme by $O(\log \kappa)$. Therefore, when we combine the SKE with an indexed ABS in the construction of GS, we require the indexed ABS to be able to deal with polynomial-depth circuits.

7 New Group Signature Constructions

By combining all the results in the previous sections, we obtain the first lattice-based group signatures in the standard model. We show four instantiations, which provide tradeoffs between the security assumption and efficiency. The first instantiation leads to a scheme that is proven secure under the SIS and LWE assumption with polynomial approximation factors, but has long parameters, meaning that the group public key and signatures are linear in the number of users N. The second instantiation is more efficient and these parameters do not depend on N. However, in order to prove security, we have to assume the subexponential hardness of the SIS problem (with polynomial approximation factors). The third instantiation is proven secure under the LPN assumption with constant noise rate and the SIS assumption with polynomial approximation factors, and has long parameters. The fourth instantiation is proven secure from the SIS assumption with subexponential approximation factors and has large parameters.

First Instantiation. The generic construction of GS schemes in Sec. 4 requires an OTS scheme, an SKE scheme, and an indexed ABS scheme. We instantiate the OTS by the scheme proposed by Mohassel [Moh11], which is strongly unforgeable under the SIS assumption with polynomial approximation factors. We instantiate the SKE by the scheme sketched in Sec. 6 (and described in full details in Appendix C). The scheme satisfies INDr-CCA security under the LWE assumption with polynomial approximation factors, key-robustness, and can have arbitrarily large message space, which are the required properties for the generic construction. Furthermore, the maximum depth of the decryption circuit of the SKE, which is denoted by d_{Dec} hereafter, is $O(\log \kappa)$. We now consider how to instantiate the indexed ABS scheme. In addition to the perfect privacy and co-selective unforgeability, we require the indexed ABS to be capable of dealing with the circuit class $\mathcal{C}_\kappa$ defined in Eq. (2). It is easy to see that we can bound the maximum depth $d_{\mathcal{C}}$ of circuits in $\mathcal{C}_\kappa$ by $d_{\mathcal{C}} = O(\log N + \log \ell_1 + d_{\mathsf{Dec}}) = O(\log \kappa)$. To obtain such an indexed ABS scheme, we apply the conversion in Sec. 3.2 to our indexed ABS scheme in Sec. 5.2, whose no-signing-query unforgeability is shown under the SIS assumption with polynomial approximation factors. Note that the conversion requires a collision resistant hash, which is known to be implied by the

same SIS assumption [MR04]. In order to make sure that the ABS scheme obtained through this conversion can deal with the circuit class $\mathcal{C}_\kappa$, we require the original indexed ABS to be capable of dealing with a circuit class $\mathcal{F}_\kappa$ defined in Eq. (1). It is easy to see that the function WldCmp can be implemented by an $O(\log \ell)$-depth circuit and thus we can bound the maximum depth $d_{\mathcal{F}}$ of the circuit class $\mathcal{F}_\kappa$ by $d_{\mathcal{F}} = d_{\mathcal{C}} + O(\log \ell) = O(\log \kappa)$. Since $d_{\mathcal{F}} = O(\log \kappa)$, we can instantiate the latter indexed ABS by the construction in Sec. 5.2. Summing up the above discussion, we have the following theorem:

Theorem 9 (Theorem 1 restated)**.** *Under the hardness of the SIS and LWE problem with polynomial approximation factors, we have a group signature scheme with CCA-selfless anonymity and full traceability in the standard model whose public parameters and signature sizes are linear in the number of users N.*

Second Instantiation. Here, we show another way of instantiating our generic construction in Sec. 4. We use the same SKE as the first instantiation above, but we instantiate the indexed ABS scheme with the scheme proposed by Tsabary [Tsa17]. To do so, we first state the following theorem.

Theorem 10 (Adapted from Sec. 6 of [Tsa17])**.** *There is an indexed ABS scheme for the circuit class $\mathcal{C}_\kappa$ defined in Eq. (2) with perfect privacy and co-selective unforgeability whose master public key and signature sizes are bounded by $\mathrm{poly}(\kappa)$, i.e., independent of the number of users N, assuming the subexponential hardness of the SIS problem with polynomial approximation factors.*

The above theorem is obtained by the result of [Tsa17], but some adaptations are required. We refer to Appendix D for discussions.

We then combine the ABS scheme given by Theorem 10 with the SKE scheme used in the first instantiation. We obtain the following theorem.

Theorem 11 (Theorem 2 restated)**.** *Under the hardness of the LWE problem and the subexponential hardness of the SIS problem with polynomial approximation factors, there exists a group signature scheme with full-traceability and CCA-selfless anonymity whose public parameters and signature sizes are $\mathrm{poly}(\kappa)$, i.e., independent of the number of users N.*

Third Instantiation. By combining the SKE from the LPN described in the previous section with the indexed ABS used in the first instantiation, we have a new group signature from the LPN assumption with constant noise rate and the SIS assumption with polynomial approximation factors. Note that this combination is possible since the decryption circuit of the LPN-based SKE can be implemented by a circuit with depth $O(\log \kappa)$.

Theorem 12 (Theorem 1 restated)**.** *Under the hardness of the SIS problem with polynomial approximation factors and the LPN problem with constant noise rate, we have a group signature scheme with CCA-selfless anonymity and full traceability in the standard model whose public parameters and signature sizes are linear in the number of users N.*

Fourth Instantiation. By combining the indexed ABS that can deal with polynomial depth circuits described in Remark 6 along with the SKE scheme based on the SIS assumption described in Sec. 6, we obtain a group signature solely from the SIS assumption, but with a subexponential approximation factor.

Theorem 13 (Theorem 1 restated)**.** *Under the hardness of the SIS problem with subexponential approximation factors, we have a group signature scheme with CCA-selfless anonymity and full traceability in the standard model whose public parameters and signature sizes are linear in the number of users N.*

Acknowledgement. The authors would like to thank Yusuke Sakai and Ai Ishida for helpful discussions and anonymous reviewers of Eurocrypt 2019 for their valuable comments. The first author was partially supported by JST CREST Grant NumberJPMJCR1302 and JSPS KAKENHI Grant Number 17J05603. The second author was supportedby JST CREST Grant No. JPMJCR1688 and JSPS KAKENHI Grant Number 16K16068.

References

[ABB10a] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, pages 553–572, 2010.

[ABB10b] S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *CRYPTO*, pages 98–115, 2010.

[ACJT00] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik. A practical and provably secure coalition-resistant group signature scheme. In *CRYPTO*, pages 255–270, 2000.

[AHM+14] J. Alwen, M. Hirt, U. Maurer, A. Patra, P. Raykov. Key-Indistinguishable Message Authentication Codes. *SCN*, pages 476-493, 2014.

[BB04a] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *CRYPTO*, pages 443–459, 2004.

[BB04b] D. Boneh and X. Boyen. Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In *EUROCRYPT*, pages 223–238, 2004.

[BBS04] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO*, pages 41–55, 2004.

[Boy10] X. Boyen. Lattice Mixing and Vanishing Trapdoors: A Framework for Fully Secure Short Signatures and More. In *PKC*, pages 499–517, 2010.

[BCC+16] J. Bootle, A. Cerulli, P. Chaidos, E. Ghadafi, and J. Groth. Foundations of fully dynamic group signatures. In *ACNS 16*, pages 117–136, 2016.

[BCH86] P. Beame, S. A. Cook, and H. J. Hoover. Log depth circuits for division and related problems. *SIAM J. Comput.*, 15(4):994–1003, 1986.

[BF14] M. Bellare and G. Fuchsbauer. Policy-based signatures. In *PKC*, pages 520–537, 2014.

[BGG+14] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, pages 533–556, 2014.

[BLP+13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *STOC*, pages 575–584, 2013.

[BMW03] M. Bellare, D. Micciancio, and B. Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In *EUROCRYPT*, pages 614–629, 2003.

[BN00] M. Bellare and C. Namprempre. Authenticated encryption: Relations among notions and analysis of the generic composition paradigm. In *ASIACRYPT*, pages 531–545, 2000.

[BS04] D. Boneh and H. Shacham. Group signatures with verifier-local revocation. In *ACM CCS*, pages 168–177, 2004.

[BSZ05] M. Bellare, H. Shi, and C. Zhang. Foundations of group signatures: The case of dynamic groups. In *CT-RSA*, pages 136–153, 2005.

[BV15] Z. Brakerski and V. Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In *TCC 2015, Part II*, pages 1–30, 2015.

[BW06] X. Boyen and B. Waters. Compact group signatures without random oracles. In *EUROCRYPT 2006*, pages 427–444, 2006.

[BW07] X. Boyen and B. Waters. Full-domain subgroup hiding and constant-size group signatures. In *PKC*, pages 1–15, 2007.

[BY93] M. Bellare and M. Yung. Certifying cryptographic tools: The case of trapdoor permutations. In *CRYPTO*, pages 442–460, 1993.

[CCH+19] R. Canetti, Y. Chen, J. Holmgren, A. Lombardi, G. Rothblum, R. Rothblum, and D. Wichs. Fiat-Shamir: From Practice to Theory. In *STOC 2019*, to appear.

[CCR16] R. Canetti, Y. Chen, and L. Reyzin. On the correlation intractability of ob-fuscated pseudorandom functions. In *TCC*, pages 389415, 2016.

[CCRR18] R. Canetti, Y. Chen, and L. Reyzin, and R. D. Rothblum. Fiat-Shamir and cor-relation intractability from strong KDM-secure encryption. *EUROCRYPT2018*, pages 91122, 2018.

[CG05] J. Camenisch and J. Groth. Group signatures: Better efficiency and new theoretical aspects. In *SCN 04*, pages 120–133, 2005.

[CH85] S. A. Cook and H. J. Hoover. A depth-universal circuit. *SIAM J. Comput.*, 14(4):833–839, 1985.

[CHKP10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552, 2010.

[CL02] J. Camenisch and A. Lysyanskaya. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *CRYPTO*, pages 61–76, 2002.

[CL04] J. Camenisch and A. Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In *CRYPTO*, pages 56–72, 2004.

[Cv91] D. Chaum and E. van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.

[DFN06] I. Damgård, N. Fazio, and A. Nicolosi. Non-interactive zero-knowledge from homomorphic encryption. In *TCC*, pages 41–59, 2006.

[DMP88] A. De Santis, S. Micali, and G. Persiano. Non-interactive zero-knowledge proof systems. In *CRYPTO*, pages 52–72, 1988.

[DKPW12] Y. Dodis, E. Kiltz, K. Pietrzak, and D. Wichs. Message authentication, revisited. In *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 355–374, 2012.

[DRS04] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT*, pages 523–540, 2004.

[FHPS13] E. S. V. Freire, D. Hofheinz, K. G. Paterson, and C. Striecks. Programmable hash functions in the multilinear setting. In *CRYPTO 2013, Part I*, pages 513–530, 2013.

[FLS90] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *FOCS*, pages 308–317, 1990.

[FS87] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO'86*, pages 186–194, 1987.

[GKV10] S. D. Gordon, J. Katz, and V. Vaikuntanathan. A group signature scheme from lattice assumptions. In *ASIACRYPT*, pages 395–412, 2010.

[Gol04] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, 2004.

[Gol08] O. Goldreich. *Computational complexity - a conceptual perspective*. Cambridge University Press, 2008.

[GOS06] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. In *EUROCRYPT*, pages 339–358, 2006.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206, 2008.

[Gro07] J. Groth. Fully anonymous group signatures without random oracles. In *ASIACRYPT*, pages 164–180, 2007.

[GS08] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT*, pages 415–432, 2008.

[GV15] S. Gorbunov and D. Vinayagamurthy. Riding on asymmetry: Efficient ABE for branching programs. In *ASIACRYPT 2015, Part I*, pages 550–574, 2015.

[GVW15] S. Gorbunov, V. Vaikuntanathan, and D. Wichs. Leveled fully homomorphic signatures from standard lattices. In *STOC*, pages 469–477, 2015.

[HL18] J. Holmgren, R. Rothblum. Delegating Computations with (Almost) Minimal Time and Space Overhead. In *FOCS*, pages 124–135, 2018.

[HLLR12] J. Herranz, F. Laguillaumie, B. Libert, and C. Ràfols. Short attribute-based signatures for threshold predicates. In *CT-RSA 2012*, pages 51–67, 2012.

[Ish18] A. Ishida. Studies on Group Signature, 2018. PhD thesis, Tokyo Institute of Technology.

[KRR17] Y. T. Kalai, G. N. Rothblum, and R. D. Rothblum. From obfuscation to thesecurity of fiat-shamir for proofs. In *CRYPTO 2017, Part II*, pages. 224251, 2017.

[KW18] S. Kim and D. J. Wu. Multi-theorem preprocessing NIZKs from lattices. In *CRYPTO 2018, Part II*, pages 733–765, 2018.

[KY06] A. Kiayias and M. Yung. Secure scalable group signature with dynamic joins and separable authorities. *IJSN*, 1(1/2):24–45, 2006.

[KPV+11] E. Kiltz, K. Pietrzak, D. Cash, A. Jain, D. Venturi: Efficient Authentication from Hard Learning Problems. In *EUROCRYPT 2011*, pages 7–26, 2011.

[LLLS13] F. Laguillaumie, A. Langlois, B. Libert, and D. Stehlé. Lattice-based group signatures with logarithmic signature size. In *ASIACRYPT 2013, Part II*, pages 41–61, 2013.

[LLM+16a] B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In *ASIACRYPT 2016, Part II*, pages 373–403, 2016.

[LLM+16b] B. Libert, S. Ling, F. Mouhartem, K. Nguyen, and H. Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In *ASIACRYPT 2016, Part II*, pages 101–131, 2016.

[LLNW14] A. Langlois, S. Ling, K. Nguyen, and H. Wang. Lattice-based group signature scheme with verifier-local revocation. In *PKC*, pages 345–361, 2014.

[LLNW16] B. Libert, S. Ling, K. Nguyen, and H. Wang. Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In *EUROCRYPT 2016, Part II*, pages 1–31, 2016.

[LNW15] S. Ling, K. Nguyen, and H. Wang. Group signatures from lattices: Simpler, tighter, shorter, ring-based. In *PKC*, pages 427–449, 2015.

[LNWX17] S. Ling, K. Nguyen, H. Wang, and Y. Xu. Lattice-based group signatures: Achieving full dynamicity with ease. In *ACNS 17*, pages 293–312, 2017.

[LNWX18] S. Ling, K. Nguyen, H. Wang, and Y. Xu. Constant-size group signatures from lattices. In *PKC 2018, Part II*, pages 58–88, 2018.

[Lys02] A. Lysyanskaya. Unique signatures and verifiable random functions from the DH-DDH separation. In *CRYPTO*, pages 597–612, 2002.

[Moh11] P. Mohassel. One-time signatures and chameleon hash functions. In *SAC 2010*, pages 302–319, 2011.

[MP12] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718, 2012.

[MPR11] H. K. Maji, M. Prabhakaran, and M. Rosulek. Attribute-based signatures. In *CT-RSA*, pages 376–392, 2011.

[MR04] D. Micciancio and O. Regev. Worst-case to average-case reductions based on Gaussian measures. In *FOCS*, pages 372–381, 2004.

[Nao91] M. Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[NZZ15] P. Q. Nguyen, J. Zhang, and Z. Zhang. Simpler efficient group signatures from lattices. In *PKC*, pages 401–426, 2015.

[OT11] T. Okamoto and K. Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In *PKC*, pages 35–52, 2011.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC*, pages 333–342, 2009.

[PLS18] R. Del Pino, Va. Lyubashevsky, and G. Seiler. Lattice-based group signatures and zero-knowledge proofs of automorphism stability. *ACM-CCS*, 2018 (To appear).

[PsV06] R. Pass, a. shelat, and V. Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *CRYPTO*, pages 271–289, 2006.

[PV08] C. Peikert and V. Vaikuntanathan. Noninteractive statistical zero-knowledge proofs for lattice problems. In *CRYPTO*, pages 536–553, 2008.

[PS19] C. Peikert and S. Shiehian. Noninteractive Zero Knowledge for NP from (Plain) Learning With Errors. Available from https://eprint.iacr.org/2019/158.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93, 2005.

[Rom90] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394, 1990.

[Rüc10] M. Rückert. Strongly Unforgeable Signatures and Hierarchical Identity-Based Signatures from Lattices without Random Oracles. In *PQCrypto*, pages 182–200, 2010.

[SAH16] Y. Sakai, N. Attrapadung, and G. Hanaoka. Attribute-based signatures for circuits from bilinear map. In *PKC 2016, Part I*, pages 283–300, 2016.

[SEH+13] Y. Sakai, K. Emura, G. Hanaoka, Y. Kawai, T. Matsuda, and K. Omote. Group signatures with message-dependent opening. In *PAIRING*, pages 270–294, 2013.

[SS96] M. Sipser and D. A. Spielman. Expander codes. *IEEE Trans. Information Theory*, 42(6):1710–1722, 1996.

[SSE+12] Y. Sakai, J. C. N. Schuldt, K. Emura, G. Hanaoka, and K. Ohta. On the security of dynamic group signatures: Preventing signature hijacking. In *PKC*, pages 715–732, 2012.

[ST01] A. Shamir and Y. Tauman. Improved online/offline signature schemes. In *CRYPTO*, pages 355–367, 2001.

[Tsa17] R. Tsabary. An equivalence between attribute-based signatures and homomorphic signatures, and new constructions for both. In *TCC 2017, Part II*, pages 489–518, 2017.

[Zém01] G. Zémor. On expander codes. *IEEE Trans. Information Theory*, 47(2):835–837, 2001.

A Omitted Definitions from Section 2

A.1 Secret Key Encryption

Syntax. Let $\{\mathcal{M}_\kappa\}_{\kappa\in\mathbb{N}}$ be a family of message spaces. In the following, we occasionally drop the subscript and simply write $\mathcal{M}$ when the meaning is clear. An SKE is defined by the following algorithms:

$\mathsf{SKE.Setup}(1^\kappa) \to \mathsf{pp}$: The setup algorithm takes as input the security parameter κ and generates a public parameter pp.

$\mathsf{SKE.Gen}(\mathsf{pp}) \to \mathsf{K}$: The key generation algorithm takes as input the public parameter pp and outputs a secret key K.

$\mathsf{SKE.Enc}(\mathsf{K},\mathsf{M}) \to \mathsf{ct}$: The encryption algorithm takes as input a secret key K and a message $\mathsf{M} \in \mathcal{M}_\kappa$ and outputs a ciphertext ct.

$\mathsf{SKE.Dec}(\mathsf{K},\mathsf{ct}) \to \mathsf{M}$ or $\bot$: The decryption algorithm takes as input a secret key K and a ciphertext ct and outputs a message M or $\bot$, which indicates that the ciphertext is not in a valid form. We assume that the decryption algorithm is deterministic.

Correctness. We require correctness: that is, for all κ, $\mathsf{pp} \in \mathsf{SKE.Setup}(1^\kappa)$, $\mathsf{K} \in \mathsf{SKE.Gen}(\mathsf{pp})$, and $\mathsf{M} \in \mathcal{M}_\kappa$, $\mathsf{SKE.Dec}(\mathsf{K},\mathsf{SKE.Enc}(\mathsf{K},\mathsf{M})) = \mathsf{M}$ holds.

Key Robustness. We say that an SKE scheme has key robustness if the following holds:

$$\Pr\left[\begin{array}{c}\mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa),\ \mathsf{K} \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp}),\ \mathsf{K}' \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp}),\ : \\ \exists \mathsf{ct} \in \{0,1\}^* \text{ s.t. } \mathsf{SKE.Dec}(\mathsf{K},\mathsf{ct}) \neq \bot \ \wedge\ \mathsf{SKE.Dec}(\mathsf{K}',\mathsf{ct}) \neq \bot\end{array}\right] = \mathrm{negl}(\kappa).$$

INDr-CCA-security. We now define INDr-CCA-security for an SKE scheme. This security notion is defined by the following game between a challenger and an adversary A. Let $\mathsf{SKE.CTSamp}$ be some efficient algorithm that takes pp as input and outputs a pseudorandom ciphertext.

Setup: At the beginning of the game, the challenger runs $\mathsf{SKE.Setup}(1^\kappa) \to \mathsf{pp}$ and samples $\mathsf{K} \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp})$. It then gives 1^κ and pp to A.

Queries: During the game, A can make the following two kinds of queries unbounded polynomially many times.

- **Encryption Queries:** Upon a query $\mathsf{M} \in \mathcal{M}_\kappa$ from A, the challenger runs $\mathsf{SKE.Enc}(\mathsf{K},\mathsf{M}) \to \mathsf{ct}$ and returns ct to A.
- **Decryption Queries:** Upon a query ct from A, the challenger runs $\mathsf{SKE.Dec}(\mathsf{K},\mathsf{ct})$ and returns the result to A.

Challenge Phase: At some point, A chooses its target message $\mathsf{M}^\star$. The challenger then samples a secret coin $\mathsf{coin} \xleftarrow{\$} \{0,1\}$. If $\mathsf{coin} = 0$, it samples the challenge ciphertext as $\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}, \mathsf{M}^\star)$. If $\mathsf{coin} = 1$, it samples the challenge ciphertext as $\mathsf{ct}^\star \xleftarrow{\$} \mathsf{SKE.CTSamp}(\mathsf{pp})$. Finally, it returns $\mathsf{ct}^\star$ to A.

Queries: After the challenge phase, A may continue to make encryption and decryption queries unbounded polynomially many times. Here, we add a restriction that A cannot make a decryption query for $\mathsf{ct}^\star$.

Guess: Eventually, A outputs $\widehat{\mathsf{coin}}$ as a guess for coin.

We say that the adversary A wins the game if $\widehat{\mathsf{coin}} = \mathsf{coin}$. We define the advantage of an adversary to be $|\Pr[\mathsf{A} \text{ wins}] - 1/2|$, where the probability is taken over the randomness of the challenger and the adversary. An SKE scheme is said to be CCA-secure if there exists an efficient pseudorandom ciphertext sampling algorithm $\mathsf{SKE.CTSamp}$ and the advantage of any PPT adversary A is negligible in the above game.

INDr-CPA-security. We now define INDr-CPA security, which is a weaker security notion than INDr-CCA security. We define the INDr-CPA security game by modifying the INDr-CCA security by prohibiting the adversary A from making any decryption queries. We do not change the winning condition of the game and the definition of the advantage. We say that a scheme satisfies INDr-CPA if the advantage of any PPT adversary A in the game is negligible.

Remark 7. *We note that INDr-CPA and INDr-CCA security defined above encompass the notion of anonymity, since it requires that a correctly generated ciphertext using a secret key* K *is indistinguishable from a random sample from* $\mathsf{SKE.CTSamp}(\mathsf{pp})$*, which is a distribution that only depends on* pp*.*

A.2 One-Time Signature

Syntax. Let $\{\mathcal{M}_\kappa\}_{\kappa\in\mathbb{N}}$ be a family of message spaces. A one-time signature (OTS) is defined by the following algorithms:

$\mathsf{OTS.KeyGen}(1^\kappa) \to (\mathsf{ovk}, \mathsf{osk})$: The key generation algorithm takes as input the security parameter 1^κ and a message $\mathsf{M} \in \mathcal{M}_\kappa$ and outputs a verification key ovk and a singing key osk.

$\mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M}) \to \tau$: The signing algorithm takes as input a secret key osk and a message $\mathsf{M} \in \mathcal{M}_\kappa$ and outputs a signature τ.

$\mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M}, \tau) \to \top$ or $\bot$: The verification algorithm takes as input a verification key ovk, a message M, and a signature τ and outputs $\top$ or $\bot$.

Correctness. We require correctness: that is, for all κ, $(\mathsf{ovk}, \mathsf{osk}) \in \mathsf{OTS.KeyGen}(1^\kappa)$, $\mathsf{M} \in \mathcal{M}_\kappa$, and $\tau \in \mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M})$, $\mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M}, \tau) = \top$ holds.

Strong Unforgeability. We now define strong unforgeability for an OTS scheme. This security notion is defined by the following game between a challenger and an adversary A.

Setup: At the beginning of the game, the challenger runs $\mathsf{OTS.KeyGen}(1^\kappa) \to (\mathsf{ovk}, \mathsf{osk})$ and gives 1^κ and ovk to A.

The Signing Query: During the game, A makes a single signing query. When A submits $\mathsf{M} \in \mathcal{M}_\kappa$ to the challenger, it runs $\mathsf{OTS.Sign}(\mathsf{osk}, \mathsf{M}) \to \tau$ and returns (M, τ) to A.

Forgery: A then outputs a forgery $(\mathsf{M}^\star, \tau^\star)$.

We say that A wins the game if $\mathsf{OTS.Vrfy}(\mathsf{ovk}, \mathsf{M}^\star, \tau^\star) = \top$ and $(\mathsf{M}, \tau) \neq (\mathsf{M}^\star, \tau^\star)$. We define the advantage of an adversary to be the probability that the adversary A wins, where the probability is taken over the randomness of the challenger and the adversary. An OTS scheme is said to be strongly unforgeable if the advantage of any any PPT adversary A in the above game is negligible.

A.3 Message Authentication Codes

Syntax. Let $\{\mathcal{M}_\kappa\}_{\kappa\in\mathbb{N}}$ be a family of message spaces. A message authentication code (MAC) is defined by the following algorithms:

$\mathsf{MAC.Gen}(1^\kappa) \to \mathsf{K}$: The key generation algorithm takes as input a security parameter 1^κ and outputs a secret key K.

$\mathsf{MAC.Sign}(\mathsf{K}, \mathsf{M}) \to \mu$: The signing algorithm takes as input a secret key K and a message $\mathsf{M} \in \mathcal{M}_\kappa$ and outputs a tag μ.

$\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{M}, \mu) \to \top$ or $\bot$: The verification algorithm takes as input a secret key K, a message M, and a tag μ and outputs $\top$ or $\bot$.

Correctness. We require correctness: that is, for all κ, $\mathsf{K} \in \mathsf{MAC.Gen}(1^\kappa)$ and $\mathsf{M} \in \mathcal{M}_\kappa$, $\mu \in \mathsf{MAC.Sign}(\mathsf{K}, \mathsf{M})$, $\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{M}, \mu) = \top$ holds.

Strong Unforgeability. We now define strong unforgeability for a MAC scheme. This security notion is defined by the following game between a challenger and an adversary A. During the game, the challenger maintains a list $\mathcal{Q}$, which is set to be empty at the beginning of the game.

Setup: At the beginning of the game, the challenger samples $\mathsf{K} \xleftarrow{\$} \mathsf{MAC.Gen}(1^\kappa)$ and gives 1^κ to A.

Queries: During the game, A can make the following two kinds of queries unbounded polynomially many times.

- **Signing Queries:** When A submits $\mathsf{M} \in \mathcal{M}_\kappa$ to the challenger, it runs $\mathsf{MAC.Sign}(\mathsf{K}, \mathsf{M}) \to \mu$ and returns μ to A. Then, the challenger adds the tuple (M, μ) to $\mathcal{Q}$.
- **Verification Queries:** Upon a query (M, μ) from A, the challenger runs $\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{M}, \mu)$ and returns the result to A.

Forgery: A then outputs a forgery $(\mathsf{M}^\star, \mu^\star)$.

We say that A wins the game if $\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{M}^\star, \mu^\star) = \top$ and $(\mathsf{M}^\star, \mu^\star) \notin \mathcal{Q}$. We define the advantage of an adversary to be the probability that the adversary A wins, where the probability is taken over the randomness of the challenger and the adversary. A MAC scheme is said to be strongly unforgeable if the advantage of any PPT adversary A in the above game is negligible.

A.4 Collision Resistant Hash Functions

Let $\mathcal{H} = \{\mathcal{H}_\kappa\}_{\kappa\in\mathbb{N}}$ be a family of functions, where $\mathcal{H}_\kappa$ is a set of functions with domain $\{0,1\}^*$ and range $\{0,1\}^\ell$, where $\ell(\kappa)$ is some polynomial. We say that $\mathcal{H}$ is a family of collision resistant hash functions if it is possible to efficiently sample an index of the function h from $\mathcal{H}_\kappa$ and

$$\Pr\left[\ h \xleftarrow{\$} \mathcal{H}_\kappa,\ (x, x') \leftarrow \mathsf{A}(1^\kappa, h),\ \ :\ x \neq x'\ \wedge\ h(x) = h(x')\ \right] = \mathrm{negl}(\kappa)$$

holds for any PPT adversary A.

B Parameter Selection

Here, we choose parameters and the circuit class for the indexed ABS scheme in Sec. 5.2. For the system to satisfy correctness and make the security proof work, we need the following restrictions.

- $\mathsf{TrapGen}$ operates properly (i.e., $m = \Omega(n \log q)$ and $\gamma_0 = \omega(\sqrt{n \log q \log m})$ by Item 4 of Lemma 12),
- γ is sufficiently large so that $\mathbf{R}^{(i)}_{F,x}$ in the signing algorithm and $\mathbf{S}^{(i)}$ in the security proof satisfy $\|\mathbf{R}^{(i)}_{F,x}\|_\infty \leq \gamma$ and $\|\mathbf{S}^{(i)}\|_\infty \leq \gamma$, (i.e., $\gamma > \gamma_0 \sqrt{n} \cdot 2^{O(d_{\mathcal{F}})}$ by Lemma 13),
- β is sufficiently large so that $\mathbf{R}^{(i)}_{F,x}$ can be extended to $[\mathbf{A} \| \mathbf{B}^{(1)}_F \| \cdots \| \mathbf{B}^{(N)}_F]^{-1}_\beta$ (i.e., $\beta \geq \gamma m \cdot \omega(\sqrt{\log m})$ by Item 2 and 3 in Lemma 12),
- we can apply Lemma 14 in the proof (i.e., $m > n \log q + \omega(\log \kappa)$),
- the security of the scheme can be reduced to the hardness of $\mathsf{SIS}(n, m, q, \beta_{\mathrm{SIS}})$ (i.e., $\beta_{\mathrm{SIS}} > Nnm\beta\gamma$), and
- $\mathsf{SIS}(n, m, q, \beta_{\mathrm{SIS}})$ is hard (i.e., $q > \beta_{\mathrm{SIS}} \cdot \sqrt{n} \cdot \omega(\log n)$).

Candidate Choice of the Parameters. We choose the parameters for the scheme as follows:

$$n = \tilde{\Theta}(\kappa), \quad m = n^{1.1}, \quad \gamma_0 = n^{0.6}, \quad \gamma = n^{1.1} \cdot 2^{O(d_{\mathcal{F}})},$$
$$\beta = n^{2.3} \cdot 2^{O(d_{\mathcal{F}})}, \quad \beta_{\mathrm{SIS}} = N \cdot n^{5.5} \cdot \left(2^{O(d_{\mathcal{F}})}\right)^2, \quad q = N \cdot n^{6.2} \cdot \left(2^{O(d_{\mathcal{F}})}\right)^2,$$

where $O(d_{\mathcal{F}})$ above represents the same function that is determined by Lemma 13. Since we have $d_{\mathcal{F}}(\kappa) = O(\log \kappa) = O(\log n)$ and $N = \mathrm{poly}(\kappa) = \mathrm{poly}(n)$, we have $\beta_{\mathrm{SIS}} = \mathrm{poly}(n)$ and thus the security of the scheme is eventually based on worst case lattice problems with polynomial approximation factors. Note that we cannot deal with $\omega(\log \kappa)$-depth circuits because $(\mathsf{PubEval}, \mathsf{TrapEval})$ does not work in polynomial time any more in this setting.

C Omitted Details from Section 6

In this section, we give the construction of SKE that is required for the generic construction of GS in Sec. 4. After reviewing some background in Appendix C.1, we show a generic construction of the required SKE from an SKE scheme with some mild (non-standard) properties and a strongly unforgeable MAC scheme in Appendix C.2. We then show how to instantiate the two building blocks from lattices: SKE from the LWE assumption in Appendix C.3 and the MAC from the SIS assumption in Appendix C.4.

C.1 Computation in $\mathbf{NC}^1$

Here, we review some operations can be performed in $\mathbf{NC}^1$, since we will use the facts extensively in this section. We first recall the result of Beame, Cook, and Hoover [BCH86]. They show that the computation of $x \mod p$ given two positive integers $x \in \{0,1\}^n$ and $p \in \{0,1\}^n$ can be performed in $\mathbf{NC}^1$, namely, can be implemented by an $O(\log n)$-depth circuit, where x and p are interpreted as integers by natural binary representations. This together with the well-known fact

that arithmetic operations such as addition, subtraction, and multiplication over the integer are in $\mathbf{NC}^1$ imply that these operations in $\mathbb{Z}_p$ are in $\mathbf{NC}^1$ as well. Similarly, since the inner-product of two vectors over the integer can be performed in $\mathbf{NC}^1$, so is the inner-product over $\mathbb{Z}_p$. Finally, we note that given two integers $x \in \{0,1\}^n$ and $y \in \{0,1\}^n$, the comparison of them can be performed in $\mathbf{NC}^1$ as follows:

$$\left(x \overset{?}{>} y\right) = \bigvee_{i\in[n]} \left(\left(\bigwedge_{j\in[i-1]} \left(x_j \overset{?}{=} y_j\right)\right) \bigwedge \left(x_i \overset{?}{>} y_i\right)\right),$$

where x and y are interpreted as integers by natural binary representations and x_i and y_i are the i-th bit of x and y, respectively.

C.2 Generic Construction

Here, we construct an SKE scheme $\mathsf{SKE}' = (\mathsf{SKE}'.\mathsf{Setup}, \mathsf{SKE}'.\mathsf{Gen}, \mathsf{SKE}'.\mathsf{Enc}, \mathsf{SKE}'.\mathsf{Dec})$ that has INDr-CCA security, key-robustness, and a decryption circuit with $O(\log \kappa)$-depth from an SKE scheme $\mathsf{SKE} = (\mathsf{SKE}.\mathsf{Setup}, \mathsf{SKE}.\mathsf{Gen}, \mathsf{SKE}.\mathsf{Enc}, \mathsf{SKE}.\mathsf{Dec})$ that has INDr-CPA security, key-robustness, and a decryption circuit with $O(\log \kappa)$-depth and a MAC scheme $\mathsf{MAC} = (\mathsf{MAC}.\mathsf{Gen}, \mathsf{MAC}.\mathsf{Sign}, \mathsf{MAC}.\mathsf{Vrfy})$ that has strong unforgeability and a verification circuit with $O(\log \kappa)$-depth. We require SKE and MAC to satisfy the following (very mild) constraints:

- The message space of SKE can be set arbitrarily large. Namely, for any $\ell(\kappa) = \mathrm{poly}(\kappa)$, we can set $\mathsf{SKE}.\mathcal{M} \supseteq \{0,1\}^{\ell}$, where $\mathsf{SKE}.\mathcal{M}$ is the message space of SKE.
- The message space of MAC can be set arbitrarily large. Namely, for any $\ell(\kappa) = \mathrm{poly}(\kappa)$, we can set $\mathsf{MAC}.\mathcal{M} \supseteq \{0,1\}^{\ell}$ where $\mathsf{MAC}.\mathcal{M}$ is the message space of MAC.
- The tag size of MAC is independent of the size of the message space. Namely, there exists a fixed polynomial $\bar{\ell}(\kappa)$ such that for any $\ell(\kappa) = \mathrm{poly}(\kappa)$, $\mathsf{M} \in \{0,1\}^{\ell} \subseteq \mathsf{MAC}.\mathcal{M}$, $\mathsf{K}_{\mathsf{MAC}} \in \mathsf{MAC}.\mathsf{Gen}(1^{\kappa})$, and $\mu \in \mathsf{MAC}.\mathsf{Sign}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{M})$, we have $\mu \in \{0,1\}^{\bar{\ell}}$. We say that MAC has *succinctness* if it satisfies this condition.

Note that any SKE scheme can be modified to satisfy the first constraint by encrypting smaller chunks in parallel. Similarly, any MAC scheme can be modified to satisfy the second and the third constraints by applying a collision resistant hash function to a message before signing on it. However, we explicitly require these conditions because it slightly simplifies the generic construction mentioned here. The instantiations of SKE and MAC that appear in Appendix C.3 and C.4 satisfy these conditions.

Let the message space of the resulting SKE scheme SKE' be $\{0,1\}^{\ell}$ and $\bar{\ell}$ be the upper bound on the length of the tag for MAC. We then set the message space $\mathsf{SKE}.\mathcal{M}$ of SKE sufficiently large so that we have $\mathsf{SKE}.\mathcal{M} \supseteq \{0,1\}^{L}$, where $L := \ell + \bar{\ell}$. We then also set the message space of MAC sufficiently large so that for any $\mathsf{R} \in \{0,1\}^{L}$, $\mathsf{pp} \in \mathsf{SKE}.\mathsf{Setup}(1^{\kappa})$, $\mathsf{K}_{\mathsf{SKE}} \in \mathsf{SKE}.\mathsf{Gen}(\mathsf{pp})$, $\mathsf{ct}_0 \in \mathsf{SKE}.\mathsf{Enc}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{R})$, and $\mathsf{M} \in \{0,1\}^{\ell}$, we have $\mathsf{ct}_0 \| \mathsf{M} \in \mathsf{MAC}.\mathcal{M}$. The construction of SKE' is as follows.

$\mathsf{SKE}'.\mathsf{Setup}(1^{\kappa})$: It first runs $\mathsf{SKE}.\mathsf{Setup}(1^{\kappa}) \to \mathsf{pp}$ and outputs $\mathsf{pp}' := \mathsf{pp}$.

$\mathsf{SKE}'.\mathsf{Gen}(\mathsf{pp}')$: Recall that we have $\mathsf{pp}' = \mathsf{pp}$. It samples $\mathsf{K}_{\mathsf{SKE}} \overset{\$}{\leftarrow} \mathsf{SKE}.\mathsf{Gen}(\mathsf{pp})$, $\mathsf{K}_{\mathsf{MAC}} \overset{\$}{\leftarrow} \mathsf{MAC}.\mathsf{Gen}(1^{\kappa})$ and outputs $\mathsf{K}' = (\mathsf{K}_{\mathsf{SKE}}, \mathsf{K}_{\mathsf{MAC}})$.

$\mathsf{SKE}'.\mathsf{Enc}(\mathsf{K}', \mathsf{M})$: It first parses K' as $\mathsf{K}' \to (\mathsf{K}_{\mathsf{SKE}}, \mathsf{K}_{\mathsf{MAC}})$. It then samples $\mathsf{R} \xleftarrow{\$} \{0,1\}^L$ and computes $\mathsf{SKE}.\mathsf{Enc}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{R}) \to \mathsf{ct}_0$ and $\mathsf{MAC}.\mathsf{Sign}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{ct}_0 \| \mathsf{M}) \to \mu$. Finally, it sets $\mathsf{ct}_1 := \mathsf{R} \oplus (\mathsf{M} \| \mu)$ and outputs $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$.

$\mathsf{SKE}'.\mathsf{Dec}(\mathsf{K}', \mathsf{ct}')$: It first parses K' as $\mathsf{K}' \to (\mathsf{K}_{\mathsf{SKE}}, \mathsf{K}_{\mathsf{MAC}})$ and ct' as $\mathsf{ct}' \to (\mathsf{ct}_0, \mathsf{ct}_1)$. It then proceeds as follows.

1. It first computes $\mathsf{R} := \mathsf{SKE}.\mathsf{Dec}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{ct}_0)$. If $\mathsf{R} = \bot$ or $\mathsf{R} \notin \{0,1\}^L$, it outputs $\bot$.
2. Otherwise, it computes $\mathsf{M} \| \mu := \mathsf{ct}_1 \oplus \mathsf{R}$.
3. It then computes $\mathsf{MAC}.\mathsf{Vrfy}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{ct}_0 \| \mathsf{M}, \mu)$. If the result is $\bot$, it outputs $\bot$. Otherwise, it outputs M.

Correctness. The correctness of the scheme is easy to verify.

Key Robustness. It is easy to see that SKE' has key robustness if so does SKE.

Depth of the Decryption Circuit. It is easy to see that the decryption algorithm of SKE' can be implemented by a $O(\log \kappa)$-depth circuit if so are the decryption algorithm of SKE and the verification algorithm of MAC.

INDr-CCA Security. We have the following theorem.

Theorem 14. SKE' *defined above is INDr-CCA-secure if* MAC *is strongly unforgeable and* SKE *is INDr-CPA-secure.*

Proof. We show the theorem by considering the following sequence of games between a PPT adversary A against the INDr-CCA security game and the challenger. In the following, let E_i denote the probability that A wins in $\mathsf{Game}\ i$. We define the distribution $\mathsf{SKE}'.\mathsf{CTSamp}(\mathsf{pp})$ to be the direct product of $\mathsf{SKE}.\mathsf{CTSamp}(\mathsf{pp})$ and the uniform distribution over $\{0,1\}^L$, where $\mathsf{SKE}.\mathsf{CTSamp}$ is the pseudorandom ciphertext sampling algorithm associated to SKE.

$\mathsf{Game}\ 0$**:** We define $\mathsf{Game}\ 0$ as an ordinary INDr-CCA-security game between A and the challenger. The advantage of A is $|\Pr[\mathsf{E}_0] - 1/2|$.

$\mathsf{Game}\ 1$**:** We change the game so that the challenger maintains lists $\mathcal{L}_0$ and $\mathcal{L}_1$ that are set to be $\emptyset$ at the beginning of the game. When A makes an encryption query for M, the challenger first generates the ciphertext $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ as $\mathsf{R} \xleftarrow{\$} \{0,1\}^L$, $\mathsf{SKE}.\mathsf{Enc}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{R}) \to \mathsf{ct}_0$, $\mathsf{MAC}.\mathsf{Sign}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{ct}_0 \| \mathsf{M}) \to \mu$, and $\mathsf{ct}_1 := \mathsf{R} \oplus (\mathsf{M} \| \mu)$. Right after the challenger returns the ciphertext ct' to A, it updates the lists as $\mathcal{L}_0 \leftarrow \mathcal{L}_0 \cup \{\mathsf{ct}_0\}$ and $\mathcal{L}_1 \leftarrow \mathcal{L}_1 \cup \{(\mathsf{ct}_0, \mathsf{R})\}$. Note that the challenger does *not* update the lists when generating a ciphertext at the challenge phase. This is done only after the encryption queries. Since this change is only conceptual, we have $\Pr[\mathsf{E}_0] = \Pr[\mathsf{E}_1]$.

$\mathsf{Game}\ 2$**:** In this game, we change the way the decryption queries are answered. When A makes a decryption query for $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$, it returns $\bot$ if $\mathsf{ct}_0 \notin \mathcal{L}_0$. As we will show in Lemma 15, we have $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathrm{negl}(\kappa)$ assuming the strong unforgeability of MAC.

$\mathsf{Game}\ 3$**:** In this game, we further change the way the decryption queries are answered. When A makes a decryption query for $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$, the challenger searches for the tuple of the form $(\mathsf{ct}_0, \mathsf{R})$ for some R in $\mathcal{L}_1$. If it cannot find such a tuple, it returns $\bot$. Otherwise, it computes $\mathsf{M} \| \mu := \mathsf{R} \oplus \mathsf{ct}_1$ and $\mathsf{MAC}.\mathsf{Vrfy}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{ct}_0 \| \mathsf{M}, \mu)$. If $\mathsf{MAC}.\mathsf{Vrfy}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{ct}_0 \| \mathsf{M}, \mu) = \top$, it returns M. Otherwise, it returns $\bot$. Note that in this game, the challenger no longer needs

the secret key to answer the decryption queries. (However, the secret key is still needed to answer the encryption queries and the challenge query.)

We claim that this does not change the view of A from the previous game. For a decryption query $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ such that $\mathsf{ct}_0 \notin \mathcal{L}_0$, the challenger returns $\perp$ in both games. For a decryption query $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ such that $\mathsf{ct}_0 \in \mathcal{L}_0$, the only difference is the way the challenger computes R. While the challenger computes $\mathsf{SKE.Dec}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{ct}_0) \to \mathsf{R}$ in the previous game (as in the actual $\mathsf{SKE}'.\mathsf{Dec}$ algorithm), it simply retrieves R in $(\mathsf{ct}_0, \mathsf{R}) \in \mathcal{L}_1$ in this game. Since we have $\mathsf{SKE.Dec}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{ct}_0) = \mathsf{R}$ for all $(\mathsf{ct}_0, \mathsf{R}) \in \mathcal{L}_1$ by the correctness of SKE, the two ways of computing result in the same R, and hence, the subsequent procedure of the challenger after computing R are exactly the same. We therefore have $\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3]$.

Game 4**:** In this game, we change the way the challenge query is answered. When A makes a challenge query for $\mathsf{M}^\star$, it samples $\mathsf{ct}_0^\star \xleftarrow{\$} \mathsf{SKE.CTSamp}(\mathsf{pp})$, $\mathsf{ct}_1^\star \xleftarrow{\$} \{0,1\}^L$ and returns $(\mathsf{ct}_0^\star, \mathsf{ct}_1^\star)$ to A regardless of whether $\mathsf{coin} = 0$ or 1. As we will show in Lemma 16, we have $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathrm{negl}(\kappa)$ assuming the INDr-CPA security of SKE.

In Game 4, the challenge ciphertext is sampled from the same distribution regardless of the value of coin. Therefore, we have $\Pr[\mathsf{E}_4] = 1/2$. By combining Lemma 15 and 16, we have that $|\Pr[\mathsf{E}_0] - 1/2|$ is negligible.

Lemma 15. *If* MAC *is strongly unforgeable, we have* $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathrm{negl}(\kappa)$

Proof. We observe that Game 1 and Game 2 are the same unless the adversary makes a decryption query $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ such that $(\mathsf{SKE}'.\mathsf{Dec}(\mathsf{K}', \mathsf{ct}') \neq \perp) \wedge (\mathsf{ct}_0 \notin \mathcal{L}_0)$. We denote this event by F and define ϵ as the probability of F occurring in Game 1. Since $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| \leq \Pr[\mathsf{F}]$, it suffices to show ϵ is negligible. To show this, we prove that there exists B who breaks the strong unforgeability of MAC with probability ϵ. We give the description of B in the following.

At the beginning of the game, B is given 1^κ from its challenger. Then, B runs $\mathsf{SKE.Setup}(1^\kappa) \to \mathsf{pp}$ and $\mathsf{SKE.Gen}(\mathsf{pp}) \to \mathsf{K}_{\mathsf{SKE}}$. It then gives 1^κ and pp to A. During the game, A makes three kinds of queries. B answers these queries as follows.

- When A makes an encryption query for M, B first samples $\mathsf{R} \xleftarrow{\$} \{0,1\}^L$ and computes $\mathsf{SKE.Enc}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{R}) \to \mathsf{ct}_0$. It then makes a signing query to its challenger for $\mathsf{ct}_0 \| \mathsf{M}$. Given μ from the challenger, it then sets $\mathsf{ct}_1 := \mathsf{R} \oplus (\mathsf{M} \| \mu)$ and returns $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ to A. It also updates $\mathcal{L}_0 \leftarrow \mathcal{L}_0 \cup \{\mathsf{ct}_0\}$ and $\mathcal{L}_1 \leftarrow \mathcal{L}_1 \cup \{(\mathsf{ct}_0, \mathsf{R})\}$.

- When A makes the challenge query for $\mathsf{M}^\star$, B proceeds as follows. If $\mathsf{coin} = 0$, B first samples $\mathsf{R}^\star \xleftarrow{\$} \{0,1\}^L$ and computes $\mathsf{ct}_0^\star$ as $\mathsf{SKE.Enc}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{R}^\star) \to \mathsf{ct}_0^\star$. It then makes a signing query to its challenger for $\mathsf{ct}_0^\star \| \mathsf{M}^\star$. Then, given $\mu^\star$ from the challenger, B computes $\mathsf{ct}_1^\star := \mathsf{R}^\star \oplus (\mathsf{M}^\star \| \mu^\star)$. Finally, it returns $\mathsf{ct}^{\star\prime} = (\mathsf{ct}_0^\star, \mathsf{ct}_1^\star)$ to A. If $\mathsf{coin} = 1$, it samples $\mathsf{ct}_0^\star \xleftarrow{\$} \mathsf{SKE.CTSamp}(\mathsf{pp})$ and $\mathsf{ct}_1^\star \xleftarrow{\$} \{0,1\}^L$ and returns $\mathsf{ct}^{\star\prime} = (\mathsf{ct}_0^\star, \mathsf{ct}_1^\star)$ to A.

- When A makes a decryption query for $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$, B first computes $\mathsf{R} := \mathsf{SKE.Dec}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{ct}_0)$. If $\mathsf{R} = \perp$ or $\mathsf{R} \notin \{0,1\}^L$, it returns $\perp$ to A. Otherwise, it computes $\mathsf{M} \| \mu := \mathsf{ct}_1 \oplus \mathsf{R}$ and makes a verification query to its challenger for $(\mathsf{ct}_0 \| \mathsf{M}, \mu)$. If the answer is $\perp$, it returns $\perp$ to A. Otherwise, it checks whether $\mathsf{ct}_0 \notin \mathcal{L}_0$. If it holds, B aborts and outputs $(\mathsf{ct}_0 \| \mathsf{M}, \mu)$ as its forgery. Otherwise, it returns M to A.

If A terminates and outputs a bit without making B abort, B outputs $\perp$.

We claim that B wins the game whenever F happens. It can easily be seen that B aborts and outputs a forgery if and only if F happens. Therefore, it suffices to prove that if B outputs a forgery $(\mathsf{ct}_0\|\mathsf{M},\mu)$, then the pair $(\mathsf{ct}_0\|\mathsf{M},\mu)$ is a successful forgery (i.e., not obtained from a signing query). There are three cases to consider.

- We first consider the case of $\mathsf{coin}=1$. In this case, B has made signing queries only for messages of the form $\widetilde{\mathsf{ct}}_0\|\widetilde{\mathsf{M}}$ such that $\widetilde{\mathsf{ct}}_0\in\mathcal{L}_0$. This implies that B has not made a signing query for $\mathsf{ct}_0\|\mathsf{M}$ since $\mathsf{ct}_0\notin\mathcal{L}_0$.
- We then consider the case where $\mathsf{coin}=0$ and B output $(\mathsf{ct}_0\|\mathsf{M},\mu)$ before A makes the challenge query. In this case, similarly to the above case, B has made signing queries only for messages of the form $\widetilde{\mathsf{ct}}_0\|\widetilde{\mathsf{M}}$ such that $\widetilde{\mathsf{ct}}_0\in\mathcal{L}_0$. This implies that B has not made a signing query for $\mathsf{ct}_0\|\mathsf{M}$ since $\mathsf{ct}_0\notin\mathcal{L}_0$.
- We finally consider the case where $\mathsf{coin}=0$ and B output $(\mathsf{ct}_0\|\mathsf{M},\mu)$ after A made the challenge query. In this case, B has made signing signing queries only for messages of the form $\widetilde{\mathsf{ct}}_0\|\widetilde{\mathsf{M}}$ such that $\widetilde{\mathsf{ct}}_0\in\mathcal{L}_0$ or $(\widetilde{\mathsf{ct}}_0,\widetilde{\mathsf{M}})=(\mathsf{ct}_0^\star,\mathsf{M}^\star)$. There are two sub-cases to consider.
  - We first consider the case of $\mathsf{ct}_0\neq\mathsf{ct}_0^\star$. In this case, B has not made a signing query for $\mathsf{ct}_0\|\mathsf{M}$ since $\mathsf{ct}_0\notin\mathcal{L}_0$ and $\mathsf{ct}_0\neq\mathsf{ct}_0^\star$.
  - We then consider the case of $\mathsf{ct}_0=\mathsf{ct}_0^\star$. It suffices to show $(\mathsf{ct}_0\|\mathsf{M},\mu)\neq(\mathsf{ct}_0^\star\|\mathsf{M}^\star,\mu^\star)$. Let $(\mathsf{ct}_0,\mathsf{ct}_1)$ be the decryption query made by A that corresponds to the forgery $(\mathsf{ct}_0\|\mathsf{M},\mu)$. We have $\mathsf{ct}_1=\mathsf{R}^\star\oplus(\mathsf{M}\|\mu)$, otherwise B has not output $(\mathsf{ct}_0\|\mathsf{M},\mu)$ as the forgery. Since we have $(\mathsf{ct}_0,\mathsf{ct}_1)\neq(\mathsf{ct}_0^\star,\mathsf{ct}_1^\star)$ by the restriction posed on A, we have $\mathsf{ct}_1\neq\mathsf{ct}_1^\star$, where we recall $\mathsf{ct}_1^\star=\mathsf{R}^\star\oplus(\mathsf{M}^\star\|\mu^\star)$. This implies $\mathsf{M}^\star\|\mu^\star\neq\mathsf{M}\|\mu$, which in particular implies $(\mathsf{ct}_0\|\mathsf{M},\mu)\neq(\mathsf{ct}_0^\star\|\mathsf{M}^\star,\mu^\star)$ as desired.

Since B simulates $\mathsf{Game}\ 1$ unless F occurs, we have that the winning probability of B is exactly ϵ. This concludes the proof of the lemma. □

Lemma 16. *If* SKE *is IND-CPA-secure, we have* $|\Pr[\mathsf{E}_3]-\Pr[\mathsf{E}_4]|=\mathrm{negl}(\kappa)$.

Proof. For the sake of contradiction, let us assume that $\epsilon:=|\Pr[\mathsf{E}_3]-\Pr[\mathsf{E}_4]|$ is non-negligible. We first observe that since the view of $\mathsf{Game}\ 3$ and $\mathsf{Game}\ 4$ when $\mathsf{coin}=1$ are the same, we have $|\Pr[\mathsf{E}_3|\mathsf{coin}=0]-\Pr[\mathsf{E}_4|\mathsf{coin}=0]|=2\epsilon$. It suffices to show an adversary B that breaks the IND-rCPA security of SKE. We give the description of B in the following.

At the beginning of the game, B is given pp from its challenger. B then samples $\mathsf{K}_{\mathsf{MAC}}\xleftarrow{\$}\mathsf{MAC.Gen}(1^\kappa)$ and $\mathsf{R}^\star\xleftarrow{\$}\{0,1\}^L$ and makes the challenge query to its challenger for $\mathsf{R}^\star$. Then, $\mathsf{ct}_0^\star$ is given to B. Here, we have $\mathsf{ct}_0^\star\xleftarrow{\$}\mathsf{SKE.CTSamp}(\mathsf{pp})$ or $\mathsf{ct}_0^\star\xleftarrow{\$}\mathsf{SKE.Enc}(\mathsf{K}_{\mathsf{SKE}},\mathsf{R}^\star)$, where $\mathsf{K}_{\mathsf{SKE}}$ is the secret key chosen by the challenger. B then gives pp to A. During the game, A makes three kinds of queries. B answers these queries as follows.

- For an encryption query M made by A, B samples $\mathsf{R}\xleftarrow{\$}\{0,1\}^L$ and makes an encryption query for its challenger on R. Given ct_0, it runs $\mathsf{MAC.Sign}(\mathsf{K}_{\mathsf{MAC}},\mathsf{ct}_0\|\mathsf{M})\to\mu$ and computes $\mathsf{ct}_1:=\mathsf{R}\oplus(\mathsf{M}\|\mu)$. Finally, it returns $\mathsf{ct}'=(\mathsf{ct}_0,\mathsf{ct}_1)$ to A and updates the list as $\mathcal{L}_0\leftarrow\mathcal{L}_0\cup\{\mathsf{ct}_0\}$ and $\mathcal{L}_1\leftarrow\mathcal{L}_1\cup\{(\mathsf{ct}_0,\mathsf{R})\}$.

- For a decryption query $\mathsf{ct}' = (\mathsf{ct}_0, \mathsf{ct}_1)$ made by A before the challenge phase, B returns $\bot$ if $\mathsf{ct}_0 \notin \mathcal{L}_0$. Otherwise, B retrieves the unique tuple of the form $(\mathsf{ct}_0, \mathsf{R})$ from $\mathcal{L}_1$ and computes $\mathsf{M}\|\mu := \mathsf{R} \oplus \mathsf{ct}_1$. It then returns M if $\mathsf{MAC.Vrfy}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{ct}_0\|\mathsf{M}, \mu) = \top$ and $\bot$ otherwise.
- For the challenge query $\mathsf{M}^\star$ made by A, B runs $\mathsf{MAC.Sign}(\mathsf{K}_{\mathsf{MAC}}, \mathsf{ct}_0^\star\|\mathsf{M}^\star) \to \mu^\star$ and computes $\mathsf{ct}_1^\star := \mathsf{R}^\star \oplus (\mathsf{M}^\star\|\mu^\star)$. Then, it returns $\mathsf{ct}^{\star\prime} = (\mathsf{ct}_0^\star, \mathsf{ct}_1^\star)$ to A.

Finally, A outputs its guess coin'. Then, B outputs 1 if $\mathsf{coin}' = 0$ and 0 otherwise.

We observe that B perfectly simulates Game 3 with $\mathsf{coin} = 0$ if $\mathsf{ct}_0^\star$ is generated as $\mathsf{ct}_0^\star \xleftarrow{\$} \mathsf{SKE.Enc}(\mathsf{K}_{\mathsf{SKE}}, \mathsf{R}^\star)$. On the other hand, if it is sampled as $\mathsf{ct}_0^\star \xleftarrow{\$} \mathsf{SKE.CTSamp}(\mathsf{pp})$, $\mathsf{R}^\star$ is distributed uniformly at random over $\{0,1\}^L$ independently from $\mathsf{ct}_0^\star$. Therefore, $\mathsf{ct}_1^\star = \mathsf{R}^\star \oplus (\mathsf{M}^\star\|\mu^\star)$ is uniformly random as well. Therefore, B simulates Game 4 with $\mathsf{coin} = 0$ in this case. Combining these observations, the lemma readily follows. □

This concludes the proof of Theorem 14. □

C.3 Instantiating the Inner SKE from the LWE

Here, we give an instantiation of the SKE that is required for the generic construction in Sec. C.2 from the LWE assumption. We require the SKE scheme to be INDr-CPA secure and to have a decryption circuit with $O(\log \kappa)$-depth. Here, we show that a secret key variant of the Regev's scheme [Reg05] satisfies these properties.

The message space of the scheme is $\{0,1\}^\ell$. In the following, we set $n = \mathrm{poly}(\kappa)$, q to be a polynomially bounded prime with $q \geq 24n + 2$, and $m = \kappa + n\lceil \log q \rceil$.

$\mathsf{SKE.Setup}(1^\kappa)$: It first sets the dimension of the matrix n and m and the modulus q. It then outputs $\mathsf{pp} := (n, m, q)$.

$\mathsf{SKE.Gen}(\mathsf{pp})$: It samples $\mathbf{S}_0 \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and $\mathbf{S}_1 \xleftarrow{\$} \mathbb{Z}_q^{n\times \ell}$. It then outputs the secret key $\mathsf{K} = (\mathbf{S}_0, \mathbf{S}_1)$.

$\mathsf{SKE.Enc}(\mathsf{K}, \mathsf{M})$: It first parses K as $\mathsf{K} \to (\mathbf{S}_0, \mathbf{S}_1)$. It then samples $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n \backslash \{\mathbf{0}\}$, $\mathbf{x}_0 \xleftarrow{\$} \mathsf{SampZ}(3\sqrt{n})^m$, and $\mathbf{x}_1 \xleftarrow{\$} \mathsf{SampZ}(3\sqrt{n})^\ell$. It then computes $\mathbf{c}_0^\top := \mathbf{a}^\top \mathbf{S}_0 + \mathbf{x}_0^\top$ and $\mathbf{c}_1^\top := \mathbf{a}^\top \mathbf{S}_1 + \mathbf{x}_1^\top + \lceil q/2 \rceil \cdot \mathsf{M}$, where $\mathsf{M} \in \{0,1\}^\ell$ is treated as a row vector in $\mathbb{Z}_q^{1\times\ell}$ here. Finally, it outputs $\mathsf{ct} := (\mathbf{a}, \mathbf{c}_0, \mathbf{c}_1)$.

$\mathsf{SKE.Dec}(\mathsf{K}, \mathsf{ct})$: It first parses the ciphertext as $\mathsf{ct} \to (\mathbf{a}, \mathbf{c}_0, \mathbf{c}_1)$. It then proceeds as follows.

1. It checks whether $(\mathbf{a} \in \mathbb{Z}_q^n \backslash \{\mathbf{0}\} \wedge \mathbf{c}_0 \in \mathbb{Z}_q^m \wedge \mathbf{c}_1 \in \mathbb{Z}_q^\ell)$ and outputs $\bot$ if otherwise.
2. It then computes $\mathbf{v}_0^\top := \mathbf{c}_0^\top - \mathbf{a}^\top \mathbf{S}_0$ and $\mathbf{v}_1^\top := \mathbf{c}_1^\top - \mathbf{a}^\top \mathbf{S}_1$.
3. If $\mathbf{v}_0 \notin [-3n, 3n]^m$, it outputs $\bot$.
4. Otherwise, it recovers $\mathsf{M}_i \in \{0,1\}^\ell$ for $i \in [\ell]$ as follows. If the i-th coefficient of $\mathbf{v}_1$ is in $[-3n, 3n]$, $\mathsf{M}_i = 0$. Otherwise, $\mathsf{M}_i = 1$.

Correctness. Since we have $\|\mathbf{x}_0\|_\infty, \|\mathbf{x}_1\|_\infty \leq 3n$ with probability 1 and $q \geq 24n + 2$, the correctness of the scheme immediately follows.

Depth of the Decryption Circuit. We show that the decryption circuit $\mathsf{SKE.Dec}(\cdot,\cdot)$ can be implemented by a circuit with $O(\log \kappa)$-depth. To show this, we observe that Step 1 checks whether each entry of the vectors is in $\mathbb{Z}_q$ or $\mathbb{Z}_q^*$, Step 2 computes inner products between vectors, and Step 3 and Step 4 check whether each entry of the vectors is in $[-3n, 3n]$ or not. As we have seen in Appendix C.1, these operations can be implemented by a circuit with $O(\log \kappa)$-depth.

Key Robustness.

Theorem 15. *The above scheme has key robustness.*

$$
\begin{aligned}
& \Pr\left[\begin{array}{c} \mathsf{pp} \xleftarrow{\$} \mathsf{SKE.Setup}(1^\kappa),\ \mathsf{K},\mathsf{K}' \xleftarrow{\$} \mathsf{SKE.Gen}(\mathsf{pp}) : \\ \exists \mathsf{ct} \in \{0,1\}^*,\ \text{ s.t. } \mathsf{SKE.Dec}(\mathsf{K},\mathsf{ct}) \neq \bot \wedge\ \mathsf{SKE.Dec}(\mathsf{K}',\mathsf{ct}) \neq \bot \end{array}\right] \\
\leq\ & \Pr\left[\begin{array}{c} \mathbf{S}_0, \mathbf{S}'_0 \xleftarrow{\$} \mathbb{Z}_q^{n\times m} : \\ \exists \mathbf{a} \in \mathbb{Z}_q^n \backslash \{\mathbf{0}\},\ \exists \mathbf{c}_0 \in \mathbb{Z}_p^m\ \text{ s.t. } \mathbf{c}_0^\top - \mathbf{a}^\top \mathbf{S}_0 \in [-3n,3n]^{1\times m} \wedge \mathbf{c}_0^\top - \mathbf{a}^\top \mathbf{S}'_0 \in [-3n,3n]^{1\times m} \end{array}\right] \\
\leq\ & \Pr\left[\ \mathbf{S}_0, \mathbf{S}'_0 \xleftarrow{\$} \mathbb{Z}_q^{n\times m}\ :\ \exists \mathbf{x} \in [-6n,6n]^m,\ \exists \mathbf{a} \in \mathbb{Z}_q^n \backslash \{\mathbf{0}\} \text{ s.t. } \mathbf{a}^\top(\mathbf{S}_0 - \mathbf{S}'_0) = \mathbf{x}^\top \right] \\
\leq\ & \sum_{\mathbf{x}\in[-6n,6n]^m, \mathbf{a}\in\mathbb{Z}_q^n\backslash\{\mathbf{0}\}} \Pr\left[\mathbf{S}''_0 \xleftarrow{\$} \mathbb{Z}_q^{n\times m},\ \mathbf{a}^\top \mathbf{S}''_0 = \mathbf{x}^\top\right] \\
\leq\ & (12n+1)^m \cdot (q^n - 1) \cdot q^{-m} \\
\leq\ & 2^{-\kappa},
\end{aligned}
$$

where the first inequality follows from Step 3 of the decryption algorithm, the third inequality follows from the union bound, and the last inequality follows from our choice of parameters.

INDr-CPA Security. Here, we define the LWE assumption and then prove security of the scheme under the assumption.

Definition 3 (The LWE Assumption)**.** *Let $n := n(\kappa)$,$m := m(\kappa)$, and $q := q(\kappa)$ be integer parameters and χ be some distribution over $\mathbb{Z}$. We say that the $\mathsf{LWE}(n,m,q,\chi)$ hardness assumption holds if for any PPT adversaries A we have*

$$
\left|\Pr\left[\ \mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n\ :\ \mathsf{A}^{\mathcal{O}_\mathbf{s}}(n,m,q) \to 1\ \right] - \Pr\left[\mathsf{A}^{\mathcal{O}_\$}(n,m,q) \to 1\right]\right| = \mathrm{negl}(\kappa),
$$

where $\mathcal{O}_\mathbf{s}$ is an oracle that returns $(\mathbf{a},b)$ that is (freshly) sampled as $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n\backslash\{\mathbf{0}\}$, $b = \mathbf{a}^\top\mathbf{s} + x$ for $x \xleftarrow{\$} \chi$ when it is called and $\mathcal{O}_\$$ is an oracle that returns $(\mathbf{a},b)$ that is freshly sampled as $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n\backslash\{\mathbf{0}\}$, $b \xleftarrow{\$} \mathbb{Z}_q$ when it is called.

Regev [Reg05] (see also [GKV10]) showed that solving $\mathsf{LWE}_{n,m,q,\chi}$ for $\chi = \mathsf{SampZ}(3\sqrt{n})$ is (quantumly) as hard as approximating the SIVP and GapSVP problems to within $\tilde{O}(\sqrt{n}q)$ factors in the ℓ_2 norm, in the worst case. In the subsequent works, (partial) dequantumization of the Regev's reduction were achieved [Pei09, BLP+13].

The following lemma can be shown by a straightforward hybrid argument.

Lemma 17. *Let us assume that $\mathsf{LWE}(n,m,q,\chi)$ assumption holds. Then, for any PPT adversaries A and polynomially bounded integer $k(\kappa)$, we have*

$$
\left|\Pr\left[\ \mathbf{S} \xleftarrow{\$} \mathbb{Z}_q^{n\times k}\ :\ \mathsf{A}^{\mathcal{O}_\mathbf{S}}(n,m,q) \to 1\ \right] - \Pr\left[\mathsf{A}^{\mathcal{O}'_\$}(n,m,q) \to 1\right]\right| = \mathrm{negl}(\kappa),
$$

where $\mathcal{O}_\mathbf{S}$ is an oracle that returns $(\mathbf{a},\mathbf{b})$ that is (freshly) sampled as $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n\backslash\{\mathbf{0}\}$, $\mathbf{b}^\top = \mathbf{a}^\top\mathbf{S} + \mathbf{x}^\top$ for $\mathbf{x} \xleftarrow{\$} \chi^k$ when it is called and $\mathcal{O}'_\$$ is an oracle that returns $(\mathbf{a},\mathbf{b})$ that is freshly sampled as $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n\backslash\{\mathbf{0}\}$, $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_q^k$ when it is called.

The following theorem addresses the INDr-CPA security of the scheme.

Theorem 16. *The above construction is INDr-CPA-secure if the* $\mathsf{LWE}_{n,m,q,\chi}$ *assumption holds with* $\chi = \mathsf{SampZ}(3\sqrt{n})$.

Proof. We show the theorem by considering the following sequence of games between a PPT adversary A against the INDr-CPA security game and the challenger. In the following, let E_i denote the probability that A outputs 1 in $\mathsf{Game}\ i$. We first define $\mathsf{SKE.CTSamp}(\mathsf{pp})$ as the algorithm that samples $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n \backslash \{\mathbf{0}\}$, $\mathbf{c}_0 \xleftarrow{\$} \mathbb{Z}_q^m$, and $\mathbf{c}_1 \xleftarrow{\$} \mathbb{Z}_q^\ell$ and outputs $(\mathbf{a}, \mathbf{c}_0, \mathbf{c}_1)$.

$\mathsf{Game}\ 0$**:** We define $\mathsf{Game}\ 0$ as an ordinary INDr-CPA-security game between A and the challenger with $\mathsf{coin} = 0$.

$\mathsf{Game}\ 1$**:** We change the game so that the challenger returns a ciphertext that is sampled from $\mathsf{SKE.CTSamp}(\mathsf{pp})$ when A makes a challenge query or an encryption query. As we will show in Lemma 18, we have $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_1]| = \mathrm{negl}(\kappa)$ under the $\mathsf{LWE}_{n,m,q,\chi}$ assumption.

$\mathsf{Game}\ 2$**:** We change the game so that the challenger honestly returns the encryption of M when A makes an encryption query for a message M. However, the challenge query is still answered by the sample from $\mathsf{SKE.CTSamp}(\mathsf{pp})$. As we will show in Lemma 19, we have $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathrm{negl}(\kappa)$ under the $\mathsf{LWE}_{n,m,q,\chi}$ assumption.

$\mathsf{Game}\ 3$**:** We define $\mathsf{Game}\ 3$ as an ordinary INDr-CPA-security game between A and the challenger with $\mathsf{coin} = 1$. As we can see, $\mathsf{Game}\ 2$ and $\mathsf{Game}\ 3$ are the same. Therefore, we have $\Pr[\mathsf{E}_3]| = \Pr[\mathsf{E}_2]$.

By combining Lemma 18 and 19, we have $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_3]| = \mathrm{negl}(\kappa)$, which concludes the proof.

Lemma 18. *If* $\mathsf{LWE}_{n,m,q,\chi}$ *holds, we have* $|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_1]| = \mathrm{negl}(\kappa)$.

Proof. For the sake of the contradiction, let us assume that there exists an adversary A who distinguishes the games. We then prove that there exists an adversary B who is given (n, m, q) as input and an oracle access to $\mathcal{O}$ and distinguish whether $\mathcal{O} = \mathcal{O}'_{\$}$ or $\mathcal{O} = \mathcal{O}'_{\mathbf{S}}$ with non-negligible adavantage, where these oracles are defined as in Lemma 17. By the same lemma, it indicates that there exists an adversary who breaks the $\mathsf{LWE}_{n,m,q,\chi}$, which is a contradiction. The adversary B proceeds as follows.

At the beginning of the game, B is given (n, m, q) as the problem instance. Then, B sets $\mathsf{pp} := (n, m, q)$ and inputs $(1^\kappa, \mathsf{pp})$ to A. During the game, A makes two kinds of queries. B answers these queries as follows.

- When A makes an encryption query for M, B first makes an oracle call to $\mathcal{O}$ to obtain $(\mathbf{a}, \mathbf{b}_0, \mathbf{b}_1)$. It then sets $\mathbf{c}_0 = \mathbf{b}_0$ and $\mathbf{c}_1 := \mathbf{b}_1 + \lceil q/2 \rceil \cdot \mathsf{M}$ and returns $\mathsf{ct} = (\mathbf{a}, \mathbf{c}_0, \mathbf{c}_1)$ to A.
- When A makes a challenge query for $\mathsf{M}^\star$, B first makes an oracle call to $\mathcal{O}$ to obtain $(\mathbf{a}^\star, \mathbf{b}_0^\star, \mathbf{b}_1^\star)$. It then sets $\mathbf{c}_0^\star = \mathbf{b}_0^\star$ and $\mathbf{c}_1^\star := \mathbf{b}_1^\star + \lceil q/2 \rceil \cdot \mathsf{M}^\star$ and returns $\mathsf{ct} = (\mathbf{a}^\star, \mathbf{c}_0^\star, \mathbf{c}_1^\star)$ to A.

Finally, A outputs its guess. B outputs the same bit as its guess.

We can easily see that B simulates $\mathsf{Game}\ 0$ if $\mathcal{O} = \mathcal{O}_{\mathbf{S}}$ and $\mathsf{Game}\ 1$ if $\mathcal{O} = \mathcal{O}'_{\$}$. Therefore, B has non-negligible advantage if so does A. This completes the proof of the lemma. □

Lemma 19. *If* $\mathsf{LWE}_{n,m,q,\chi}$ *holds, we have* $|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathrm{negl}(\kappa)$.

Proof. The proof of the lemma is the same as that of Lemma 18, except that B does not make a query to $\mathcal{O}$ to answer the challenge query. Instead, it runs $\mathsf{SKE.CTSamp}(\mathsf{pp})$ to obtain the pseudorandom ciphertext and returns it to A. □

This completes the proof of the theorem. □

C.4 Instantiating the MAC from the SIS

Here, we give an instantiation of the MAC from the SIS assumption that is required for the generic construction in Appendix C.2. We require the MAC scheme to be strongly unforgeable and to have succinct tags and a verification circuit with $O(\log \kappa)$-depth. Such a scheme can be obtained by using the (public key) signature scheme proposed by Micciancio and Peikert [MP12], which is a variant of the signature scheme by Boyen [Boy10], as a MAC scheme. In more details, they construct a signature scheme that satisfies an intermediate security notion that they call strong unforgeability under static chosen-message attack from the SIS assumption and suggest to combine the scheme with a suitable chameleon hash function. It is known that the resulting scheme satisfies strong unforgeability under adaptive chosen message attack [ST01]. The chameleon hash can be instantiated by the construction given by Cash et al. [CHKP10]. After combining them, we then downgrade the resulting (public key) signature scheme to obtain a MAC scheme. The resulting MAC scheme inherits strong unforgeability from the public key signature scheme.

The message space of the MAC will be $\{0,1\}^{\ell(\kappa)}$ for some polynomial $\ell(\kappa)$. In the following, we set integer parameters $n(\kappa) = \mathrm{poly}(\kappa)$, $k = \lceil \log q \rceil = O(\log n)$, $L = nk$, $q = O(n^4)$, $\bar{m} = O(nk)$, $m = \bar{m} + 2nk$, $\beta = \omega(\sqrt{n \log q \log m})$, and $\gamma = O(nk) \cdot \omega(\log n)$.

$\mathsf{MAC.Gen}(1^\kappa)$: It first sets the parameters $n, k, L, q, \bar{m}, m, \beta, \gamma$ as above. It then samples $(\mathbf{A}, \mathbf{A}_{\gamma_0}^{-1}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^{\bar{m}+nk}, q)$, $\mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_L \xleftarrow{\$} \mathbb{Z}_q^{n \times nk}$, $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{\ell+m}$, and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^n$. It finally outputs $\mathsf{K} = (\mathbf{A}, \mathbf{A}_{\gamma_0}^{-1}, \mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_L, \mathbf{B}, \mathbf{u})$.

$\mathsf{MAC.Sign}(\mathsf{K}, \mathsf{M})$: To sign on a message $\mathsf{M} \in \{0,1\}^\ell$, it first samples $\mathbf{r} \xleftarrow{\$} \mathsf{SampZ}(\beta)^m$. It then computes $\widetilde{\mathsf{M}} := \mathbf{B}\binom{\mathsf{M}}{\mathbf{r}} \in \mathbb{Z}_q^n$. We then interpret $\widetilde{\mathsf{M}}$ as a bitstring in $\{0,1\}^L$. Then, it computes $\mathbf{A}_{\widetilde{\mathsf{M}}}$ defined as

$$\mathbf{A}_{\widetilde{\mathsf{M}}} := \left[\mathbf{A}_0 + \sum_{i \in [L]} \widetilde{\mathsf{M}}_i \cdot \mathbf{A}_i \right], \tag{5}$$

where $\widetilde{\mathsf{M}}_i$ is the i-th bit of $\widetilde{\mathsf{M}}$. Then, it extends $\mathbf{A}_{\gamma_0}^{-1}$ to $[\mathbf{A} \| \mathbf{A}_{\widetilde{\mathsf{M}}}]_\gamma^{-1}$ and samples a vector $\mathbf{e}$ as $\mathbf{e} \xleftarrow{\$} [\mathbf{A} \| \mathbf{A}_{\widetilde{\mathsf{M}}}]_\gamma^{-1}(\mathbf{u})$. Finally, it returns $\mu = (\mathbf{r}, \mathbf{e})$.

$\mathsf{MAC.Vrfy}(\mathsf{K}, \mathsf{M}, \mu)$: It first parses the tag as $\mu \to (\mathbf{r}, \mathbf{e}) \in \mathbb{Z}^m \times \mathbb{Z}^m$. It then proceeds as follows.

1. It first checks whether $\|\mathbf{r}\|_2^2 \le \beta^2 m$ and $\|\mathbf{e}\|_2^2 \le \gamma^2 m$ and outputs $\perp$ otherwise.
2. It then computes $\widetilde{\mathsf{M}} := \mathbf{B}\binom{\mathsf{M}}{\mathbf{r}}$ and then $\mathbf{A}_{\widetilde{\mathsf{M}}}$ as Eq. (5). Here, each operations between the entries of vectors and matrices are computed in parallel.
3. It then checks whether $[\mathbf{A} \| \mathbf{A}_{\widetilde{\mathsf{M}}}]\mathbf{e} \stackrel{?}{=} \mathbf{u}$. Otherwise, it outputs $\perp$.

Correctness and Strong Unforgeability. The correctness is immediate. The strong unforgeability of the scheme as a MAC can be proven under the SIS assumption (with polynomial modulus) by the results of [MP12] and [CHKP10].

Succinctness of the Tag. We can see that the length of a tag is bounded by $2m \log q \leq O(n \log^2 q) = O(n \log^2 n)$, which is independent of the message length ℓ. Thus, the tag is succinct in our sense.

Depth of the Verification Circuit. We show that the verification circuit $\mathsf{MAC.Vrfy}(\cdot, \cdot)$ can be implemented by a circuit with $O(\log \kappa)$-depth. To show this, we observe that Step 1 requires computation of Euclidean norms of vectors and comparisons between integers, Step 2 requires computation of inner-products over $\mathbb{Z}_q$ to compute $\widetilde{\mathsf{M}}$ and $\mathbf{A}_{\widetilde{\mathsf{M}}}$, and Step 3 requires multiplications of vectors and matrices over $\mathbb{Z}_q$. As we have seen in Appendix C.1, these operations can be implemented by a circuit with $O(\log \kappa)$-depth.

D Detailed Discussion on Theorem 10

Theorem 10 is obtained by the result by [Tsa17], but some adaptations are required. In fact, the scheme shown in the paper is a constrained signature (CS), not an indexed ABS. Here, we discuss that the scheme implies an adaptively unforgeable ABS scheme via complexity leveraging, which in turn implies an indexed ABS scheme with the required properties. Here, we use the fact that an ABS scheme with adaptive unforgeability and perfect privacy can be used as an indexed ABS with co-selective unforgeability and perfect privacy, by simply ignoring the additional inputs.

In CS, a signing key is associated with a circuit C and it is possible to sign on a string x iff $C(x) = 1$. On the other hand, in ABS, a signing key is associated with a string x and it is possible to sign on a circuit-message pair (C, M) iff $C(x) = 1$. To use CS as an ABS, we associate the signing key with a universal circuit $U(\cdot, \cdot, x)$, which takes as input a circuit-message pair (C, M) and outputs $C(x)$, and generate a signature for (C, M) by regarding it as a string. As for the function class, we require the ABS scheme to support the class of $O(\log \kappa)$-depth circuit. Since we can construct a universal circuit U whose depth is only constant times deeper than that of C by the result of Cook and Hoover [CH85], we require the original CS to support the function class of $O(\log \kappa)$-depth circuits. While the CS scheme given by Tsabary can support any polynomially bounded depth circuits, this requires subexponential modulus for the scheme. However, observing that we only require a CS scheme for $O(\log \kappa)$-depth circuits for our application, we can obtain a scheme with polynomial modulus by switching to the more modulus-size efficient lattice evaluation algorithm proposed by Gorbunov and Vinayagamurthy [GV15] (which only works for $O(\log \kappa)$-depth circuits) from those proposed by Boneh et al. [BGG+14] (which works for any polynomially bounded depth circuits). As for security, since the CS scheme is proven selectively unforgeable, so is the ABS scheme. Namely, we can prove the security when the adversary chooses its target circuit-message pair $(C^\star, \mathsf{M}^\star)$ for the forgery at the beginning of the game. This security notion is not enough for our purpose, but by utilizing complexity leveraging [BB04b], we can prove adaptive security of the scheme. In the reduction from selective security to adaptive security, we have to randomly guess $(C^\star, \mathsf{M}^\star)$. Recall that we need to support the circuit class $\mathcal{C}_\kappa$ that is defined in Eq. (2). Since the length of the message $\mathsf{M}^\star$ is bounded by $\mathrm{poly}(\kappa)$ and a circuit $C^\star \in \mathcal{C}_\kappa$ can be described by ovk and ct, which can be described by binary strings with length $\mathrm{poly}(\kappa, \log N)$, the reduction loss is $2^{-\mathrm{poly}(\kappa, \log N)}$. To compensate for the degradation in the advantage, we need to enlarge the dimension n of the scheme to be $\mathrm{poly}(\kappa, \log N)^{1/\epsilon} = \mathrm{poly}(\kappa)$ where ϵ is some constant in $(0, 1)$ dictating the subexponential hardness of the SIS problem. Summing up our discussion, we obtain Theorem 10.

Contents