On Structure-Preserving Cryptography and Lattices

Dennis Hofheinz[1], Kristina Hostáková[1], Roman Langrehr[1], and Bogdan Ursu[1]

Department of Computer Science, ETH Zurich, Switzerland
{hofheinz, kristina.hostakova, roman.langrehr, bogdan.ursu}@inf.ethz.ch

Abstract. The Groth-Sahai proof system is a highly efficient pairing-based proof system for a specific class of group-based languages. Cryptographic primitives that are compatible with these languages (such that we can express, e.g., that a ciphertext contains a valid signature for a given message) are called "structure-preserving". The combination of structure-preserving primitives with Groth-Sahai proofs allows to prove complex statements that involve encryptions and signatures, and has proved useful in a variety of applications. However, so far, the concept of structure-preserving cryptography has been confined to the pairing setting.

In this work, we propose the first framework for structure-preserving cryptography in the lattice setting. Concretely, we

- define "structure-preserving sets" as an abstraction of (typically noisy) lattice-based languages,
- formalize a notion of generalized structure-preserving encryption and signature schemes (capturing a number of existing lattice-based encryption and signature schemes),
- construct a compatible zero-knowledge argument system that allows to argue about lattice-based structure-preserving primitives,
- offer a lattice-based construction of verifiably encrypted signatures in our framework.

Along the way, we also discover a new and efficient *strongly* secure lattice-based signature scheme. This scheme combines Rückert's lattice-based signature scheme with the lattice delegation strategy of Agrawal et al., which yields more compact and efficient signatures.

We hope that our framework provides a first step towards a modular and versatile treatment of cryptographic primitives in the lattice setting.

Keywords. Structure-preserving cryptography, lattice-based cryptography, public-key cryptography.

1 Introduction

Structure-preserving cryptography. Groth-Sahai (GS) proofs [35] are practical non-interactive zero-knowledge (NIZK) proof systems for a very general class of group-based languages. Essentially, GS proofs allow to argue in zero-knowledge about the satisfiability of systems of equations over groups that may involve exponentiation, of course group operations, and even pairing operations. When used in conjunction with "suitably algebraic" group-based cryptographic primitives (like encryption or signature schemes), GS proofs allow to efficiently prove complex statements like "This ciphertext contains an electronic passport for John Smith that is certified by a government authority."[1] In comparison to a generic approach (with, say, a generic NIZK system for NP [27]), such a "native" approach is significantly more practical.

"Suitably algebraic" cryptographic primitives are called *structure-preserving* [1, 34] (or, in a slightly different formulation, *automorphic* [29]). Numerous examples of structure-preserving signature (e.g., [1–3, 20, 21, 33]) and public-key encryption schemes (e.g., [15, 24, 26, 39]), as well as other primitives (e.g., [12, 53]) are known, based on different computational assumptions, and having different efficiency and security features.

All of these building blocks can be combined, and GS proofs can be used to argue about such combinations efficiently. However, so far, the paradigm of structure-preserving relies on a particular algebraic setting (of pairing-friendly cyclic groups), and it is unclear whether a similar modular combination of cryptographic primitives is also possible over other domains.[2]

[1] Such a combination has been suggested before (e.g., [10, 11, 13]), but GS proofs allow a much more general treatment, and a broader class of languages and potential applications.

[2] Of course, dedicated protocols for concrete tasks (such as identity escrow [37] or verifiable encryption [16]) exist also based on other assumptions. Also, very efficient lattice-based commit-and-prove protocols for general classes of languages exist in the random oracle model [42]. However, nothing comparable to the full "structure-preserving cryptography" paradigm (that ensures a non-interactive and conceptually simple plug-and-play combination of different primitives) exists in other algebraic settings.

This work: structure-preserving cryptography over lattices. In this work, we initiate the study of structure-preserving cryptography over lattices. We put forward suitable definitions of structure-preserving signature and encryption schemes, and present a suitable NIZK system for proving statements about combinations of these primitives. Hence, in short, our core contributions are

- a suitable definition of lattice-based structure-preserving cryptographic primitives (including the modeling of a number of existing signature and encryption schemes according to this definition),
- a suitable zero-knowledge argument system that allows to show statements about lattice-based structure-preserving primitives,
- as an application (and to demonstrate the usefulness of our approach), a modular lattice-based protocol for verifiably encrypted signatures.

As we will explain, our notion of lattice-based structure-preserving primitives is not quite as universal as in the GS setting. This allows us to model a large class of primitives, but also asks for some degree of compatibility among the used primitives. We still believe that our abstract framework is a step towards plug-and-play lattice-based cryptography. Indeed, one benefit of our approach is modularity: It is true that the security analysis for each lattice-based component (i.e., signature or encryption scheme) needs to keep track of noise growth and failure probabilities. However, due to our interface, this analysis needs to be done only once *per component*, not once for every possible *combination of components*.

Contribution 1: a definition of lattice-based structure-preserving primitives. First, we cannot use or easily adapt existing (group-based) definitions of structure-preserving primitives: with computations over lattices, there is no equivalent of "exponentiation" or "pairing". Besides, typically lattice-based ciphertexts or signatures often feature a "noise term", which may grow with operations on these values. Once the noise term becomes too large, decryption or verification becomes unreliable. Hence, operations on these values are limited in a quantitative way, and this limitation should be reflected in a definition of structure-preserving cryptography.

Since lattice-based cryptographic constructions usually work over the ring $\mathbb{Z}_q$ (for a suitable integer q), it is tempting to call the solutions to arbitrary systems of linear equations over $\mathbb{Z}_q$, possibly with boundaries on norms (to accommodate noise terms), structure-preserving. Unfortunately, we do not know how to instantiate a proof system for such general sets in the standard model.[3]

So instead of trying to match the group-based definition, we start from scratch with a relatively simple definition of "structure-preserving sets" modelling exactly the noise terms of lattice-based cryptography. We present a standard-model non-interactive proof system for these sets, and aim to interpret signatures and ciphertexts (or, rather, the randomness of ciphertexts) as structure-preserving sets. To express more powerful statements in terms of structure-preserving sets, we additionally require our structure-preserving signature and encryption schemes to allow for suitable homomorphic operations (that, e.g., allow to evaluate a signature inside an encryption scheme).

Fortunately, we discover that several existing signature and encryption schemes satisfy our definitions. Examples include Regev encryption [48] and its dual variant [31], the GSW leveled homomorphic encryption scheme [32], and the signature schemes of Boyen [14] and Rückert [49].[4]

At this point, the mentioned required compatibility among used primitives is crucial: we unfortunately cannot combine arbitrary lattice-based structure-preserving encryption and signature schemes. Essentially, we require that the encryption scheme allows to homomorphically evaluate an encrypted signature. This allows to combine, e.g., the GSW FHE scheme with all of the mentioned signature schemes; alternatively, we can combine any additively homomorphic scheme (such as Regev's scheme or its dual variant) with Rückert's scheme or its mentioned new and more compact variant, but *not* with Boyen's scheme.

Contribution 2: a compatible NIZK argument system. To allow arguing about combinations of encryption and signature schemes, we also introduce an analogue of GS proofs. In our case, we use the LWE-based NIZK system of Libert

[3] We note that in the random oracle model, very efficient such proof systems exist [25, 44].

[4] Rückert's scheme uses the "Bonsai trees" lattice delegation method of [19]. As an aside, we also make explicit a vastly more compact version of Rückert's scheme that uses the more compact lattice delegation strategy of [5]. While this modification entails no significant technical complications, it may be worthwhile to point out.

et al. [38] as a basis. This proof system is based upon a Σ-protocol [22] for proving that an LWE encryption contains a certain value. (That Σ-protocol is later converted to a NIZK system by applying the Fiat-Shamir transform [28] in the standard model, with a correlation-intractable hash function.) To suit our needs, however, we need to generalize this proof system to structure-preserving sets (i.e., to statements that are valid "up to noise"). This requires a more careful analysis, and in particular a liberal use of rejection sampling [40].

We should emphasize that we are interested in a standard-model proof system. Indeed, while our application does not require this, we would like to be able to argue about encrypted *proofs* (and thus achieve the "nestable" property of Groth-Sahai proofs). If proof verification involves random oracle queries, this is not possible transparently. We should note, however, that our proof system supports only linear languages, while its verification itself is not linear. Hence, nesting proofs of our proof system is only possible when using leveled homomorphic encryption schemes (that allow to verify even a nonlinear encrypted proof through homomorphic evaluation). We leave open the construction of a lattice-based proof system for a language that includes its own verification.

Contribution 3: lattice-based verifiably encrypted signatures. Finally, we demonstrate the usefulness of our approach using the setting of verifiably encrypted signatures [7, 13, 30, 50]. Concretely, we show how to combine lattice-based structure-preserving signature and an encryption schemes to obtain a scheme that allows to prove that a given ciphertext contains an encryption of a valid signature for given (publicly known) message. While generic constructions (e.g., using lattice-based zero-knowledge for NP [46]) for this task are possible, and very efficient techniques for related problems exist in the random oracle world [25, 44], it appears that our protocol is the first non-generic (i.e., at least somewhat efficient) lattice-based verifiably encrypted signature scheme in the standard model.

More related work. As already mentioned, there is a very successful line of work [8, 25, 42, 43] that aims at practical (non-interactive) zero-knowledge proofs from lattices in the random oracle model. The supported languages are very general and include typical "noisy linear" languages, as crucial for many lattice-based schemes. Conceptually, these schemes are commit-and-prove schemes, much like Groth-Sahai proofs.

On the other hand, the use of random oracles appears inherent. For instance, the scheme from [42] is obtained by using the Fiat-Shamir transform on a suitable Σ-protocol. Unlike in our setting, these Σ-protocols do not appear to satisfy the requirements for the use of correlation-intractable hash functions as replacements for random oracles. Still, when one is not interested in nesting proofs (and if one accepts random oracles), then these protocols appear to be excellent replacements for our proof system.

1.1 Technical overview

We now take a closer look at our framework. Our first step will be to define *structure-preserving sets*, an abstraction of "noise terms" that are omnipresent in lattice-based cryptography.

Structure-preserving sets. We call a set $S \subseteq \mathbb{Z}_q^d$ *structure-preserving* if there is a ("noise") distribution $\mathcal{D}$ such that

- $\mathcal{D}$ "smudges" elements from S in the sense that for any $\mathbf{s}, \mathbf{s}' \in S$ and $\mathbf{d} \leftarrow \mathcal{D}$, the values $\mathbf{s} + \mathbf{d}$ and $\mathbf{s}' + \mathbf{d}$ are statistically close.[5]
- Smudging with $\mathcal{D}$ preserves (non-)membership in S, in the sense that for $\overline{S} = \mathbb{Z}_q^d \setminus S$, we have that $S + \mathrm{supp}(\mathcal{D})$ and $\overline{S} + \mathrm{supp}(\mathcal{D})$ are disjoint.[6] This condition guarantees that the smudging process is non-trivial.

It is easy to see that the set of short-norm vectors is structure-preserving. But structure-preserving sets also cover more complex cases, such as the set of vectors close to a given vector, (the union of) intervals, or the cartesian product of structure-preserving sets. In essence, we only require that a structure-preserving set is "non-trivially smudgeable".

Jumping ahead, structure-preserving sets will be used to model, e.g., the "raw" (i.e., un-rounded) verification output of signature schemes. This verification output only encodes a bit (the verification verdict), but may need to be smudged for further processing to avoid leakage about the signature. In fact, we now proceed to (informally) define structure-preserving signature and encryption schemes.

[5] This is an oversimplification. Our actual definition involves rejection sampling and actually only requires "closeness in a significant portion of cases".

[6] Again, this oversimplifies. We really only require this for almost all vectors of $\overline{S}$ and a large enough subset of $\mathrm{supp}(\mathcal{D})$.

Structure-preserving signatures. A (lattice-based) signature scheme is called structure-preserving for a family $\mathcal{F}$ of functions if each verification key vk and message msg defines an $f \in \mathcal{F}$ such that a given signature σ is valid if and only if $f(\sigma) \in S$ for a (fixed) structure-preserving set S.[7] We will be particularly interested in families $\mathcal{F}$ of *linear* functions, since such $\mathcal{F}$ will allow for (non-generic) zero-knowledge proofs. This is also the reason for the need to smudge f's output: existing lattice-based signature schemes usually postprocess the result of a linear operation with a rounding step obtain the verification verdict bit. Instead of this rounding step, we require that $f(\sigma) \in S$.

We show that Rückert's signature scheme [49] is structure-preserving for a linear $\mathcal{F}$, and that Boyen's signature scheme [14] is structure-preserving for an $\mathcal{F}$ that contains linear functions and functions computed by low-depth Boolean circuits. Additionally, we present a more compact variant of Rückert's scheme (that is also strongly secure and structure-preserving for a linear $\mathcal{F}$). This new scheme is retrieved by replacing the "Bonsai trees" lattice delegation method of [19] with the more compact lattice delegation strategy of [5].

Structure-preserving encryption. We say that a (lattice-based) encryption scheme is structure-preserving if ciphertexts are of the form

$$\mathsf{ct} = \mathbf{B}\mathbf{r} + g(\mathsf{msg})$$

for a matrix $\mathbf{B} \in \mathbb{Z}_q^{d \times r}$, $\mathbf{r} \in S$ for a structure-preserving set S, and an invertible and additively homomorphic "message encoding function" g.[8] Intuitively, we require that $\mathbf{r} \in S$ to be able to argue about "valid encryptions" (for which the encrypted message is uniquely determined).

For our applications, it will also be beneficial if the scheme is $\mathcal{F}$-homomorphic, in the sense that $\mathsf{ct} = \mathbf{B}\mathbf{r} + g(\mathsf{msg})$ allows to efficiently compute $\mathsf{ct}' = \mathbf{B}\mathbf{r}' + g(f(\mathsf{msg}))$ for any $f \in \mathcal{F}$ (possibly at the price of a larger noise).

We observe that Regev's encryption scheme [48], its dual variant [31], and the GSW leveled homomorphic encryption scheme [32] fit our framework (for linear functions, resp. low-depth circuits). While itself not technically involved, this provides a helpful uniform way to reason about these schemes.

A zero-knowledge protocol for encrypted structure-preserving sets. Our last ingredient is a suitable (lattice-based, non-interactive) zero-knowledge proof system that allows to argue about structure-preserving primitives (and in particular structure-preserving sets). More concretely, we start with a Σ-protocol that shows that a given ciphertext (from an arbitrary structure-preserving encryption scheme) encrypts an element $\mathsf{msg} \in S$ from a structure-preserving set S.

This Σ-protocol is derived from a Σ-protocol due to Libert et al. [38] for proving equality of encrypted messages (where the used encryption scheme is a variant [6] of Regev encryption). The basic protocol of [38] (following Schnorr's blueprint [52]) proceeds as follows. Say that we want to show that a given ciphertext ct is an encryption of 0.[9] The prover P then starts by sending a fresh 0-encryption ct_0 to the verifier V. Then V chooses to either open ct_0 or $\mathsf{ct}_0 \cdot \mathsf{ct}$ (by sending the random coins of that ciphertext).

Soundness follows from the fact that if ct is not a 0-encryption, then at least one of the two ciphertexts ct_0 and $\mathsf{ct}_0 \cdot \mathsf{ct}$ encrypts a nonzero value. (Of course, to obtain a negligible soundness error, the above protocol will have to be repeated.) Zero-knowledge follows from the fact that if one knows in advance which ciphertext is opened, one can program ct_0 such that the to-be-opened ciphertext surely encrypts 0.

In our setting, we want to prove that ct encrypts some $\mathbf{s} \in S$ (without revealing $\mathbf{s}$). Since S is a structure-preserving set, we can smudge $\mathbf{s}$ with a suitable smudging vector $\mathbf{d} \leftarrow \mathcal{D}$. When we set up ct_0 as an encryption of such a $\mathbf{d}$, we obtain that

- opening ct_0 reveals only a smudging value $\mathbf{d}$, and
- opening $\mathsf{ct}_0 \cdot \mathsf{ct}$ reveals a smudged value $\mathbf{s} + \mathbf{d}$, which is (almost) statistically independent of $\mathbf{s}$.

Hence, using a similar strategy as in [38], we obtain zero-knowledge. Moreover, since smudging preserves (non-)membership in S, we obtain soundness (after sufficiently many repetitions). The actual proof is more involved than this overview, of course, largely because of the already mentioned rejection sampling necessary for statistical closeness.

[7] Our actual definition also considers signatures which carry "tags" which can be used to preprocess messages prior to verifying (but whose publication does not harm security).

[8] We also define the notion of a "noise level" of a ciphertext which we ignore in this overview.

[9] Since the used homomorphic encryption scheme is homomorphic, we can reduce proving equality of ciphertexts to proving 0-encryptions.

We only briefly mention that our protocol is compatible with recent standard-model techniques [18, 46] to transform Σ-protocols in the lattice setting into non-interactive zero-knowledge (NIZK) proofs. We use a sophisticated variant [38] of this approach [10] that even achieves unbounded simulation-soundness for specific classes of Σ-protocols. In the end, we obtain a NIZK argument system for encrypted structure-preserving sets.

From structure-preserving sets to structure-preserving primitives. As an application (and to demonstrate the usefulness of our proof system), we construct a verifiably encrypted signature (VES [7, 13, 30, 50]) scheme. Intuitively, in a VES scheme, a dedicated signer hands out *encrypted signatures* (i.e., signatures generated using the signer's secret key, and encrypted under the public key of a designated "adjudicator"). Such encrypted signatures also contain a NIZK proof of validity (i.e., of the fact that the given ciphertext really contains a valid signature for a given message). In case of a conflict, however, the adjudicator can extract (by decrypting) a "proper" (i.e., non-simulatable) signature from a given encrypted signature. VES schemes are useful, e.g., in contract signing applications [7, 13].

Using our framework, a lattice-based VES scheme can be obtained generically from a structure-preserving signature scheme, a structure-preserving encryption scheme with compatible message space (and such that it allows to homomorphically verify signatures), and our zero-knowledge proof system for (encrypted) structure-preserving sets. These primitives are combined in a straightforward way. Perhaps the most interesting part of this construction is the fact that it suffices to prove that an encrypted value comes from a structure-preserving set. Indeed, to prove that a given encryption contains a valid signature, we (a) first homomorphically evaluate that signature inside the encryption, and (b) then prove that the result corresponds to an "accept". Recall that by our definition of structure-preserving signatures, this means proving membership in a structure-preserving set.

Our formal proof is similar to a proof for an existing VES scheme by Fuchsbauer [30] that uses pairing-based structure-preserving cryptography.

1.2 Roadmap

After recalling some notation and standard building blocks in Section 2, we present our definition of structure-preserving sets in Section 3. Building on this definition, we proceed with our notions of structure-preserving signatures (Section 4) and structure-preserving encryption schemes (Section 5). We identify and construct example schemes in Sections 4.1 and 5.1 and Appendices A and B. Our Σ-protocol for (encrypted) structure-preserving sets appears in Section 6, followed by its conversion to a NIZK proof system in Section 7. The VES application follows in Section 8 and we discuss its efficiency in Appendix E.

2 Preliminaries

2.1 Notation

A function f is *negligible* if for every polynomial $p(\cdot)$, there exists an $n_0 \in \mathbb{N}$ such that for every $n > n_0$ it holds that $f(n) < \frac{1}{p(n)}$. We write negl to denote an arbitrary negligible function. Let X and Y be two probability distributions over a domain Ω. The *statistical distance* between X and Y is defined as $\Delta(X, Y) := \frac{1}{2}\sum_{\omega \in \Omega} |\Pr[X = \omega] - \Pr[Y = \omega]|$. We say that two ensembles $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ of distributions are *statistically indistinguishable*, denoted as $\{X_n\}_{n\in\mathbb{N}} \approx_s \{Y_n\}_{n\in\mathbb{N}}$, if $\Delta(X_n, Y_n) = \mathsf{negl}(n)$. We say that two ensembles $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ of distributions are *computationally indistinguishable*, denoted as $\{X_n\}_{n\in\mathbb{N}} \approx_c \{Y_n\}_{n\in\mathbb{N}}$, if for every probabilistic polynomial time (PPT) adversary $\mathcal{A}$, we have $|\Pr[\mathcal{A}(X_n) = 1] - \Pr[\mathcal{A}(Y_n) = 1]| = \mathsf{negl}(n)$.

Let S be a finite set. Then by $x \leftarrow_{\mathrm{R}} S$ we mean that x was sampled from the uniform distribution over S. For a probability distribution $\mathcal{D}$ on S we denoted the support by $\mathrm{supp}(\mathcal{D}) \subseteq S$.

Let $\mathbf{x} \in \mathbb{R}^n$ be a column vector. The x_i, for $i \in \{1, \ldots, n\}$ denotes the i-th coordinate of $\mathbf{x}$. The ℓ_2*-norm* of $\mathbf{x}$ is defined as $\|\mathbf{x}\| := \sqrt{\sum_{i=1}^{n} x_i^2}$. The ℓ_2 *norm of a matrix* $\mathbf{M} \in \mathbb{R}^{n\times m}$ is defined as $\|\mathbf{M}\| = \sup_{\mathbf{x}\in\mathbb{R}^m, \mathbf{x}\neq 0} \frac{\|\mathbf{M}\mathbf{x}\|}{\|\mathbf{x}\|}$. We denote $\overline{\mathbf{M}}$ the Gram-Schmidt orthogonalization of the matrix $\mathbf{M}$.

[10] One important advantage of [38] is that it only requires the homomorphic evaluation of a low-depth circuit in the computation of the CI-Hash function from [46].

For two sets $A, B \subseteq \mathbb{Z}_q^n$, we define the sets $A \setminus B, A+B, A-B \subseteq \mathbb{Z}_q^n$ as follows:

$$\begin{aligned} A \setminus B &:= \{x \mid x \in A \ \wedge \ x \notin B\}, \\ A + B &:= \{(a_1+b_1, \dots, a_n+b_n) \mid (a_1, \dots, a_n) \in A, (b_1, \dots, b_n) \in B\}, \\ A - B &:= \{(a_1-b_1, \dots, a_n-b_n) \mid (a_1, \dots, a_n) \in A, (b_1, \dots, b_n) \in B\}. \end{aligned}$$

If $A = \emptyset$ or $B = \emptyset$, then we define $A + B := \emptyset$ and $A - B := \emptyset$.

We use $B_\delta(S) := \{\mathbf{v} \in \mathbb{Z}_q^n \mid (\min_{\mathbf{s} \in S, \mathbf{x} \in \mathbb{Z}^n} \|\mathbf{v} - \mathbf{s} + q\mathbf{x}\|) \leq \delta\}$ to denote the closed δ-ball around a set of vectors $S \subseteq \mathbb{Z}_q^n$.

We write $H \leq G$ to denote that H is a subgroup of a group G.

We say that a function $f\colon X \to Y$ is *invertible* if there exists a function $f^{-1}\colon Y \to X \cup \{\bot\}$ such that (i) f^{-1} is efficiently computable, (ii) for every $x \in X$ it holds $f^{-1}(f(x)) = x$, and (iii) for every $y \in Y \setminus \mathrm{Img}(f)$ it holds $f^{-1}(y) = \bot$.

2.2 Lattices

Let us recall various basic lattice notions and hardness problems that we need in later sections of this work.

Let $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ be a symmetric positive-definite matrix, and $\mathbf{c} \in \mathbb{R}^n$. Then the *Gaussian function* on $\mathbb{R}^n$ is defined as $\rho_{\mathbf{\Sigma}}(\mathbf{x}) := \exp\{-\pi \mathbf{x}^\top \mathbf{\Sigma}^{-1} \mathbf{x}\}$. The function extends to sets in the usual way. That is, for any countable set $A \subset \mathbb{R}^n$, $\rho_{\mathbf{\Sigma}}(A) := \sum_{\mathbf{x} \in A} \rho_{\mathbf{\Sigma}}(\mathbf{x})$. Moreover, for every countable set $A \subset \mathbb{R}^n$ and any $\mathbf{x} \in A$, the *discrete Gaussian function* is defined by $\rho_{A,\mathbf{\Sigma}}(\mathbf{x}) := \frac{\rho_{\mathbf{\Sigma}}(\mathbf{x})}{\rho_{\mathbf{\Sigma}}(A)}$ and we denote the corresponding *discrete Gaussian distribution* as $\mathcal{D}_{A,\mathbf{\Sigma}}$. If $\mathbf{\Sigma} = \sigma^2 \cdot \mathbf{I}_n$, where $\mathbf{I}_n$ is the $n \times n$ identity matrix, we denote the Gaussian function as ρ_σ, the discrete Gaussian function as $\rho_{A,\sigma}$ and the discrete Gaussian distribution as $\mathcal{D}_{A,\sigma}$ for short. We will make use of the following tail bound for the discrete Gaussian distribution for $\mathbb{Z}^n$.

Lemma 2.1 ([41, Lemma 4.4]). *For any $k > 1$ we have $\Pr_{\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^n,\sigma}}[\|\mathbf{x}\| > k\sigma\sqrt{n}] < k^n e^{\frac{n}{2}(1-k^2)}$.*

Let $\mathbf{B} \in \mathbb{R}^{m \times n}$ be a matrix with linearly independent columns $\mathbf{b}_1, \dots, \mathbf{b}_n \in \mathbb{R}^m$ for $m \geq n$. The m-dimensional *lattice* $\mathit{\Lambda}$ with lattice basis $\mathbf{B}$ is defined as $\mathit{\Lambda} = \{\mathbf{y} \in \mathbb{R}^m \mid \exists \mathbf{s} \in \mathbb{Z}^n, \, \mathbf{y} = \mathbf{B}\mathbf{s}\}$. The *dual lattice* of $\mathit{\Lambda}$ is defined as $\mathit{\Lambda}^* := \{\mathbf{z} \in \mathbb{R}^m \mid \forall \mathbf{y} \in \mathit{\Lambda}, \, \mathbf{z}^\top \mathbf{y} \in \mathbb{Z}\}$. For $q \geq 2$ and a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ we define two m-dimensional integer lattices $\mathit{\Lambda}^\perp(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m \mid \mathbf{A}\mathbf{x} = 0 \mod q\}$ and $\mathit{\Lambda}(\mathbf{A}) = \{\mathbf{y} \in \mathbb{Z}^m \mid \exists \mathbf{s} \in \mathbb{Z}^n, \, \mathbf{A}^\top \mathbf{s} = \mathbf{y} \mod q\}$.

Definition 2.2 (Learning With Errors). *Let q, m, n be positive integers and χ be a probability distribution on $\mathbb{Z}$. The $\mathsf{LWE}_{m,n,q,\chi}$ problem is to distinguish the following two distributions: $\{(\mathbf{A}, \mathbf{b}) \mid (\mathbf{A}, \mathbf{b}) \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m\}$ and $\{(\mathbf{A}, \mathbf{b}) \mid \mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n, \mathbf{e} \leftarrow \chi^m, \mathbf{b} := \mathbf{A}^\top \mathbf{s} + \mathbf{e}\}$.*

Definition 2.3 (LWE with short secrets). *Let q, m, n be positive integers and χ be a probability distribution on $\mathbb{Z}$. The $\mathsf{SSLWE}_{m,n,q,\chi}$ problem is to distinguish the following two distributions: $\{(\mathbf{A}, \mathbf{b}) \mid (\mathbf{A}, \mathbf{b}) \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m\}$ and $\{(\mathbf{A}, \mathbf{b}) \mid \mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow \chi^n, \mathbf{e} \leftarrow \chi^m, \mathbf{b} := \mathbf{A}^\top \mathbf{s} + \mathbf{e}\}$.*

Definition 2.4 (Short Integer Solution). *Let q, m, n be positive integers, $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and $\beta \in \mathbb{R}$. The $\mathsf{SIS}_{m,n,q,\beta}$ problem in ℓ_2 norm is to find a non-zero vector $\mathbf{x} \in \mathbb{Z}^m$ such that $\mathbf{A}\mathbf{x} = \mathbf{0} \mod q$ and $\|\mathbf{x}\| \leq \beta$.*

Definition 2.5 (Inhomogeneous Short Integer Solution). *Let q, m, n be positive integers, $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{y} \in \mathbb{Z}_q^n$ and $\beta \in \mathbb{R}$. The $\mathsf{ISIS}_{m,n,q,\beta}$ problem in ℓ_2 norm is to find a non-zero vector $\mathbf{x} \in \mathbb{Z}^m$ such that $\mathbf{A}\mathbf{x} = \mathbf{y} \mod q$ and $\|\mathbf{x}\| \leq \beta$.*

Remark 2.6. When the $\mathsf{SIS}_{m,n,q,\beta}$ problem is hard, the $\mathsf{ISIS}_{m,n,q,\beta'}$ problem is hard as well where β' is only slightly larger than β.

We will use the following variant of the Rejection Sampling Lemma by Lyubashevsky to "smudge" small noise – despite working with a polynomial modulus – by rejection sampling.

Lemma 2.7 **([41, Theorem 4.6]).** *For all $T \in \mathbb{N}$ and $\sigma \geq T\sqrt{n}$ there exists a constant M such that for all $\mathbf{v} \in \mathbb{Z}^n$ with $\|\mathbf{v}\| \leq T$ the distribution*

$$\boxed{\mathbf{d} \leftarrow \mathcal{D}_{\mathbb{Z}^n,\sigma},\ \mathbf{z} := \mathbf{v} + \mathbf{d},\ \texttt{Output}: \begin{cases} \mathbf{z} & \texttt{with prob. } \min\left(\frac{\rho_{\mathbb{Z}^n,\sigma}(\mathbf{z})}{M\rho_{\mathbb{Z}^n,\sigma}(\mathbf{d})}, 1\right) \\ \perp & \texttt{otherwise} \end{cases}}$$

is within statistical distance $1/(M2^n)$ of

$$\boxed{\mathbf{d} \leftarrow \mathcal{D}_{\mathbb{Z}^n,\sigma},\ \texttt{Output}: \begin{cases} \mathbf{d} & \texttt{with prob. } 1/M \\ \perp & \texttt{otherwise} \end{cases}}\ .$$

2.3 Cryptographic primitives

We first recall the definition of a gap Σ-protocol and a trapdoor gap Σ-protocol. Our definitions are adapted from the work of Libert et al. [38] which in turn closely follow the definitions put forward by Canetti et al. [18].

Definition 2.8 **(Gap Σ-protocol).** *Let $\mathcal{L} = (\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$ be a language associated with two NP relations $\mathcal{R}_{\mathsf{zk}}, \mathcal{R}_{\mathsf{sound}}$ s.t. $\mathcal{L}_{\mathsf{zk}} \subseteq \mathcal{L}_{\mathsf{sound}}$ (i.e., $\mathcal{L}$ is a* gap *language).*

Let $\mathsf{Setup}(1^\lambda, \mathcal{L})$ be an algorithm that takes an unary encoded security parameter $\lambda \in \mathbb{N}$ and a language description $\mathcal{L}$ as input and outputs a common reference string crs. An interactive proof system $\Pi = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ in the common reference string model is a Gap Σ-protocol *for $\mathcal{L}$ if it has the following 3-move form, where $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{L})$, x is a statement and w is a witness:*

$$\boxed{\begin{array}{lcl} \underline{\textit{Prover } \mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2)} & & \underline{\textit{Verifier } \mathsf{V}} \\ \texttt{Input}: (\mathsf{crs}, x, w) & & \texttt{Input}: (\mathsf{crs}, x) \\ (\mathsf{a}, \mathsf{st}) \leftarrow \mathsf{P}_1(\mathsf{crs}, x, w) & \xrightarrow{\mathsf{a}} & \\ & \xleftarrow{\mathsf{Chal}} & \mathsf{Chal} \leftarrow_{\mathsf{R}} \mathcal{C} \\ \mathsf{z} \leftarrow \mathsf{P}_2(\mathsf{st}, \mathsf{a}, \mathsf{Chal}) & \xrightarrow{\mathsf{z}} & \\ & & b \leftarrow \mathsf{V}(\mathsf{crs}, x, \mathsf{a}, \mathsf{Chal}, \mathsf{z}) \\ & & \texttt{Output}: b \end{array}}$$

and the following properties holds:

Completeness: *If $(x, w) \in \mathcal{R}_{\mathsf{zk}}$ and both P and V follow the protocol, then V accepts with probability $1 - \mathsf{negl}(\lambda)$. Formally, for every $(x, w) \in \mathcal{R}_{\mathsf{zk}}$, we have*

$$\Pr\left[\mathsf{V}(\mathsf{crs}, x, \mathsf{a}, \mathsf{Chal}, \mathsf{z}) = 1 \;\middle|\; \begin{array}{c} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{L}), \\ (\mathsf{a}, \mathsf{st}) \leftarrow \mathsf{P}_1(\mathsf{crs}, x, w), \\ \mathsf{Chal} \leftarrow_{\mathsf{R}} \mathcal{C}, \mathsf{z} \leftarrow \mathsf{P}_2(\mathsf{st}, \mathsf{a}, \mathsf{Chal}) \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Special zero-knowledge: *There exists a PPT simulator ZKSim such that for any $\mathsf{crs} \in \mathsf{Setup}(1^\lambda, \mathcal{L})$, any $(x, w) \in \mathcal{R}_{\mathsf{zk}}$ and any challenge $\mathsf{Chal} \in \mathcal{C}$, the following distributions are computationally indistinguishable:*

$$\{(\mathsf{a}, \mathsf{Chal}, \mathsf{z}) \mid (\mathsf{a}, \mathsf{z}) \leftarrow \mathsf{ZKSim}(\mathsf{crs}, x, \mathsf{Chal})\} \approx_c \\ \{(\mathsf{a}, \mathsf{Chal}, \mathsf{z}) \mid (\mathsf{a}, \mathsf{st}) \leftarrow \mathsf{P}_1(\mathsf{crs}, x, w), \mathsf{z} \leftarrow \mathsf{P}_2(\mathsf{st}, \mathsf{a}, \mathsf{Chal})\}\,.$$

Special soundness: *For any CRS $\mathsf{crs} \in \mathsf{Setup}(1^\lambda, \mathcal{L})$, any $x \notin \mathcal{L}_{\mathsf{sound}}$, and any first prover's message a, there exists at most one challenge $\mathsf{Chal} = f(\mathsf{crs}, x, \mathsf{a}) \in \mathcal{C}$ for which there exists a valid prover's reply z, i.e., $\mathsf{V}(\mathsf{crs}, x, \mathsf{a}, \mathsf{Chal}, \mathsf{z}) = 1$. The function f is called the* bad challenge function *of Π.*

Definition 2.9 (Trapdoor gap Σ-protocol). *Let $\mathcal{L} = (\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$ be a language associated with two NP relations $\mathcal{R}_{\mathsf{zk}}, \mathcal{R}_{\mathsf{sound}}$, s.t. $\mathcal{L}_{\mathsf{zk}} \subseteq \mathcal{L}_{\mathsf{sound}}$. A gap Σ-protocol $\Pi = (\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ for $\mathcal{L}$ with a bad challenge function f is a* trapdoor gap Σ-protocol *if there exist PPT algorithms* $(\mathsf{TrapSetup}, \mathsf{BadChallenge})$ *with the following syntax:*

$\mathsf{TrapSetup}(1^\lambda, \mathcal{L}, \tau_{\mathcal{L}})$**:** *Given public parameters* par*, language $\mathcal{L}$ and a membership trapdoor $\tau_{\mathcal{L}}$ for the language $\mathcal{L}_{\mathsf{sound}}$ as input, it outputs a CRS* crs *and a trapdoor $\tau_\Sigma \in \{0,1\}^{\ell_\tau}$ for some $\ell_\tau(\lambda)$;*

$\mathsf{BadChallenge}(\tau_\Sigma, \mathsf{crs}, x, \mathsf{a})$**:** *Given a trapdoor τ_Σ, a CRS* crs*, a statement x and a first prover message* a *as input, it outputs a challenge* Chal*;*

and satisfying the following properties:

CRS indistinguishability: *For any trapdoor $\tau_{\mathcal{L}}$ for the language $\mathcal{L}_{\mathsf{sound}}$, the following distributions are computationally indistinguishable*

$$\{\mathsf{crs} \mid \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{L})\} \approx_c \{\mathsf{crs} \mid \mathsf{crs} \leftarrow \mathsf{TrapSetup}(1^\lambda, \mathcal{L}, \tau_{\mathcal{L}})\}.$$

Correctness: *There exists a language-specific trapdoor $\tau_{\mathcal{L}}$ s.t. for any instance $x \notin \mathcal{L}_{\mathsf{sound}}$, all pairs $(\mathsf{crs}, \tau_\Sigma) \in \mathsf{TrapSetup}(1^\lambda, \mathcal{L}, \tau_{\mathcal{L}})$ and any first prover message* a*, we have* $\mathsf{BadChallenge}(\tau_\Sigma, \mathsf{crs}, x, \mathsf{a}) = f(\mathsf{crs}, x, \mathsf{a})$.

Let us now recall the definition of a Non-Interactive Zero Knowledge (NIZK) proofs. We closely follow the definition given by Libert et al. [38].

Definition 2.10 (NIZK). *Let $\mathcal{L} = (\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$ be a language associated with two NP relations $\mathcal{R}_{\mathsf{zk}}$, $\mathcal{R}_{\mathsf{sound}}$, such that $\mathcal{L}_{\mathsf{zk}} \subseteq \mathcal{L}_{\mathsf{sound}}$ and statements are of bit-length N. A non-interactive zero-knowledge (NIZK) argument system Π for a language $\mathcal{L}$ consists of three PPT algorithms* $(\mathsf{Setup}, \mathsf{P}, \mathsf{V})$ *with the following syntax:*

$\mathsf{Setup}(1^\lambda, \mathcal{L}, \tau_{\mathcal{L}})$: *Given an unary encoded security parameter λ, a language $\mathcal{L}$ and a membership testing trapdoor $\tau_{\mathcal{L}}$ for $\mathcal{L}$ as input, it outputs a CRS* crs*.*

$\mathsf{P}(\mathsf{crs}, x, w)$**:** *Given a CRS* crs*, a statement $x \in \{0,1\}^N$, and a witness w as input, the proving algorithm outputs a proof π.*

$\mathsf{V}(\mathsf{crs}, x, \pi)$**:** *Given a CRS* crs*, a statement $x \in \{0,1\}^N$, and a proof π as input, the verification algorithm outputs a decision bit.*

Moreover, Π should satisfy the following properties.
Completeness: *For any $(x, w) \in \mathcal{R}_{\mathsf{zk}}$, any $\mathsf{lbl} \in \{0,1\}^*$ and any membership testing trapdoor $\tau_{\mathcal{L}}$ for $\mathcal{L}$, we have*

$$\Pr[\mathsf{V}(\mathsf{crs}, x, \pi) = 1 \mid \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{L}, \tau_{\mathcal{L}}), \pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w)] \geq 1 - \mathsf{negl}(\lambda).$$

Soundness: *For any $x \in \{0,1\}^N \setminus \mathcal{L}_{\mathsf{sound}}$, any membership testing trapdoor $\tau_{\mathcal{L}}$ for $\mathcal{L}$ and any PPT prover P^*, we have*

$$\Pr[\mathsf{V}(\mathsf{crs}, x, \pi) = 1 \mid \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{L}, \tau_{\mathcal{L}}), \pi \leftarrow \mathsf{P}^*(\mathsf{crs}, x)] \leq \mathsf{negl}(\lambda).$$

Zero-Knowledge: *There is a PPT simulator $(\mathsf{Sim}_0, \mathsf{Sim}_1)$ such that for any PPT adversary $\mathcal{A}$, we have that for all trapdoors $\tau_{\mathcal{L}}$:*

$$\begin{aligned} &|\Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{P}}(\mathsf{crs},\cdot,\cdot)}(\mathsf{crs}) \mid \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, \mathcal{L}, \tau_{\mathcal{L}})] \\ &\qquad - \Pr[1 \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Sim}}(\mathsf{crs},\tau_{\mathsf{zk}},\cdot,\cdot)}(\mathsf{crs}) \mid (\mathsf{crs}, \tau_{\mathsf{zk}}) \leftarrow \mathsf{Sim}_0(1^\lambda, \mathcal{L})]| \leq \mathsf{negl}(\lambda), \end{aligned}$$

where $\mathcal{O}_{\mathsf{P}}(\mathsf{crs}, x, w)$ outputs $\perp$ if $(x, w) \notin \mathcal{R}_{\mathsf{zk}}$ and $\pi \leftarrow \mathsf{P}(\mathsf{crs}, x, w)$ otherwise, and $\mathcal{O}_{\mathsf{Sim}}(\mathsf{crs}, \tau_{\mathsf{zk}}, x, w)$ outputs $\perp$ if $(x, w) \notin \mathcal{R}_{\mathsf{zk}}$ and $\mathsf{Sim}_1(\mathsf{crs}, \tau_{\mathsf{zk}}, x)$ otherwise.

Finally we recall the standard definition for digital signature and a public key encryption scheme.

Definition 2.11 (Digital Signature). *A* digital signature scheme *Σ for a message space $\mathcal{M}$ and signature space $\mathbb{S}$ consist of three PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver})$ *with the following syntax*

$\mathsf{KeyGen}(1^\lambda)$**:** *Given an unary encoded security parameter* λ *as input, it outputs a verfication key* vk *and a signing key* sk.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{msg})$**:** *Given a signing key* sk *and a message* $\mathsf{msg} \in \mathcal{M}$ *as input, it outputs a signature* $\mathsf{sig} \in \mathbb{S}$.

$\mathsf{Ver}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig})$**:** *Given a verification key* vk*, a message* $\mathsf{msg} \in \mathcal{M}$ *and a signature* $\mathsf{sig} \in \mathbb{S}$ *as input, it outputs* 1 *(indicating a valid signature) or* 0 *(indicating an invalid signature).*

A digital signature scheme $\Sigma = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver})$ *is* correct*, if for every message* $\mathsf{msg} \in \mathcal{M}$*, we have*

$$|\Pr[\mathsf{Ver}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig}) = 1 \mid (\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda), \mathsf{sig} \leftarrow \mathsf{Sign}(\mathsf{sk}, \mathsf{msg})]| \geq 1 - \mathsf{negl}(\lambda).$$

Definition 2.12 (Public-Key Encryption). *A* public key encryption scheme Π *for a message space* $\mathcal{M}$ *consist of three PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *with the following syntax*

$\mathsf{KeyGen}(1^\lambda)$**:** *Given an unary encoded security parameter* λ *as input, it outputs a public key* pk *and a secret key* sk.

$\mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$**:** *Given a public key* pk *and a message* $\mathsf{msg} \in \mathcal{M}$ *as input, it outputs a ciphertext* ct.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$**:** *Given a secret key* sk *and a ciphertext* ct *as input, it outputs a message* $\mathsf{msg} \in \mathcal{M}$ *or* $\perp$ *(indicating a failure).*

A PKE scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is* correct*, if for every* $\mathsf{msg} \in \mathcal{M}$*, we have*

$$|\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = \mathsf{msg} \mid (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda), \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mathsf{msg})]| \geq 1 - \mathsf{negl}(\lambda).$$

3 Structure-Preserving Sets

The first building block in our framework is the notion of a structure-preserving set, which is a crucial tool in capturing the defining characteristics of a specific family of lattice-based signatures, encryption schemes and NIZKs which are compatible with each other. The properties that lead to such structure-preserving cryptographic primitives are described in later sections.

Let q be a large prime. A structure-preserving set S is a special subset of $\mathbb{Z}_q^d$ that can be rerandomized to obtain a rerandomized set $S' = S + D$ (where D is a set which contains the rerandomizing terms). Given a vector $\mathbf{s} \in S$, we can rerandomize $\mathbf{s}$ to obtain $\mathbf{s}' \in S + D$. The structure-preserving property of S ensures that given $\mathbf{s}'$, one is able to check whether the original vector $\mathbf{s} \in \mathbb{Z}_q^d$ belonged to S or whether it lied outside of S. In particular, vector $\mathbf{s}'$ allows to check membership of the original $\mathbf{s}$, but it hides its original value.

Definition 3.1 (Uniformly Structure-Preserving Set). *We say that a set* $S \subseteq \mathbb{Z}_q^d$ *is* uniformly structure-preserving *if (i) there exists a subset* $D \subseteq \mathbb{Z}_q^d$ *such that for all messages* $\mathbf{s}, \mathbf{s}' \in S$

$$\boxed{\mathbf{d} \leftarrow_{\mathrm{R}} D, \quad \mathtt{Output:}\ \mathbf{s} + \mathbf{d}} \approx_s \boxed{\mathbf{d} \leftarrow_{\mathrm{R}} D, \quad \mathtt{Output:}\ \mathbf{s}' + \mathbf{d}}$$

(ii) for $\overline{S} := \mathbb{Z}_q^d \setminus S$ *it holds that* $(S + D) \cap (\overline{S} + D) = \emptyset$*, and the membership problem for* D *and* $S + D$ *are easy. We call the maximal statistical distance between the first two boxed distributions the* structure-preserving error.

To provide some intuition about the introduced notion, let us demonstrate the definition of a concrete example that we use later in the paper. Namely, we show that cosets of subgroups are uniformly structure-preserving.

Example 3.2 (Cosets of subgroups). Every coset S of an additive subgroup $G \leq \mathbb{Z}_q^d$ is uniformly structure-preserving.

Proof. By definition of a coset, all the sets $S_{\mathbf{s}} = \{\mathbf{s} + \mathbf{d} \mid \mathbf{d} \in G\}$ (for $\mathbf{s} \in S$) are the same set S again. Thus by picking $D := G$, we get that for all $\mathbf{s}, \mathbf{s}' \in S$, $\mathbf{s} + \mathbf{d}$ and $\mathbf{s}' + \mathbf{d}$ for $\mathbf{d} \leftarrow_{\mathrm{R}} D$ are identically distributed. Hence the first part of the definition is satisfied and the structure-preserving error is 0.

For $\mathbf{x} \in \mathbb{Z}_q^d \setminus S$, we know that $\mathbf{x} \in S'$ for $S' \neq S$ being another coset of G. Thus for every $\mathbf{d} \in G$, we have $\mathbf{x} + \mathbf{d} \in S'$. Since different cosets are disjoint, the second part of the definition is satisfied as well. □

Remark 3.3. The above example, in particular, implies that

1. all additive subgroups of $\mathbb{Z}_q^d$ are uniformly structure-preserving; and
2. all singleton sets are uniformly structure-preserving, because they are cosets of the trivial subgroup $\{\mathbf{0}\}$.

In order to define lattice-based structure-preserving signatures and encryptions, we will need a more generic definition of a structure-preserving set. Namely, we do not want to restrict ourselves to $\mathbf{d}$ being sampled uniformly at random, but from any distribution on $\mathbb{Z}_q^d$. Looking ahead, since we work with lattice-based primitives, we are particularly interested in Gaussian distributions. Along with the change of distribution for $\mathbf{d}$, we generalize the definition by loosening some of its condition. At a high level, in both the first and the second part of the definition, we allow for small errors with some probability.

Definition 3.4 (Structure-Preserving Set). *We say that a set $S \subseteq \mathbb{Z}_q^d$ is* structure-preserving *with noise growth δ if there exists an efficiently sampleable probability distribution $\mathcal{D}$ on $\mathbb{Z}_q^d$, a constant $\alpha \in (0,1]$ and a function* $\mathsf{success} : S \times S \times \operatorname{supp}(\mathcal{D}) \to (0,1]$ *such that (i) for all messages $\mathbf{s}, \mathbf{s}' \in S$*

$$\boxed{\begin{array}{l} \mathbf{d} \leftarrow \mathcal{D} \\ \texttt{Output:} \begin{cases} \mathbf{s}+\mathbf{d} & \texttt{with prob.} \\ & \quad \mathsf{success}(\mathbf{s},\mathbf{s}',\mathbf{d}) \\ \perp & \texttt{otherwise} \end{cases} \end{array}} \approx_s \boxed{\begin{array}{l} \mathbf{d} \leftarrow \mathcal{D} \\ \texttt{Output:} \begin{cases} \mathbf{s}'+\mathbf{d} & \texttt{with prob. } \alpha \\ \perp & \texttt{otherwise} \end{cases} \end{array}}$$

and (ii) there exists a set $D' \subseteq \mathbb{Z}_q^d$, that we will call the smudging set, *such that $\Pr_{\mathbf{d}\leftarrow\mathcal{D}}[\mathbf{d} \in D'] \geq 1 - \mathsf{negl}(\lambda)$ for a negligible function* negl, *and for $\overline{S}_\delta := \mathbb{Z}_q^d \setminus B_\delta(S)$, it holds that $(S + D') \cap (\overline{S}_\delta + D') = \emptyset$. Moreover, the membership problem for D' and $(S + D')$ are easy. We call* negl *the* soundness error.

It is easy to see that uniformly structure-preserving sets sets are special cases of structure-preserving sets.

Lemma 3.5. *Let S be an uniformly structure-preserving set. Then S is a structure-preserving set with noise growth* 0 *and soundness error* 0.

Proof. By setting $\mathcal{D}$ to be the uniform distribution on D, $\mathsf{success}$ to be the constant function 1, $\alpha := 1$ and $D' = D$, we directly obtain that S is a structure-preserving with noise growth 0 and soundness error 0. □

Let us complete this section by providing an example of a structure-preserving set which is not uniformly structure-preserving.

Example 3.6 (Close vectors). Every set $S \subseteq \mathbb{Z}_q^d$ where $S - S$ is T-bounded (i.e., $S - S \subseteq B_T(\{\mathbf{0}\})$) is structure-preserving with noise growth $4Td + 1$, when d grows polynomially with the security parameter.

Proof. Pick $\mathcal{D} := \mathcal{D}_{\mathbb{Z}^d,\sigma}$ with $\sigma := T\sqrt{d}$. For all $\mathbf{s}, \mathbf{s}' \in S$, by Lemma 2.7, the distribution that outputs $\mathbf{s} - \mathbf{s}' + \mathbf{d}$ for $\mathbf{d} \leftarrow \mathcal{D}_{\mathbb{Z}^d,\sigma}$ with probability $\mathsf{success}(\mathbf{s}, \mathbf{s}', \mathbf{d}) := \min\left(\frac{\rho_{\mathbb{Z}^d,\sigma}(\mathbf{s}-\mathbf{s}'+\mathbf{d})}{M\rho_{\mathbb{Z}^d,\sigma}(\mathbf{d})}, 1\right)$ is statistically close to outputting $\mathbf{d} \leftarrow \mathcal{D}_{\mathbb{Z}^d,\sigma}$ with probability $\alpha := 1/M$ for a constant M. By adding $\mathbf{s}'$ to the output of these two distributions, we get that the first condition for a structure-preserving set is satisfied.

Pick $D' := B_{2Td}(\{\mathbf{0}\})$ as smudging set. By the tail bound for Gaussian distributions (Lemma 2.1) we have $\Pr_{\mathbf{d}\leftarrow\mathcal{D}_{\mathbb{Z}^d,\sigma}}[\|\mathbf{d}\| > 2Td] < 2^d e^{\frac{-3d}{2}} = \left(2e^{-3/2}\right)^d < \frac{1}{2^d}$, which shows that this choice is valid. For $\mathbf{x} \in \overline{S}_\delta := \mathbb{Z}_q^d \setminus B_{4Td+1}(S)$ and $\mathbf{d} \in D'$ we have $\mathbf{x} + \mathbf{d} \in \mathbb{Z}_q^d \setminus B_{2Td}(S)$. On the other hand, for $\mathbf{s} \in S$ we have $\mathbf{s} + \mathbf{d} \in B_{2Td}(S)$. This implies that $(S + D') \cap (\overline{S}_\delta + D') = \emptyset$ which is the second condition for a structure-preserving set. □

Remark 3.7. This example, in particular, implies that sets of small vectors are structure-preserving. Namely, let $S \subseteq \mathbb{Z}_q^d$ be a T-bounded set. Then by triangular inequality, $S - S$ is $2T$-bounded and hence S structure-preserving with noise growth $8Td + 1$.

Next, we show that structure-preserving sets are closed under the cartesian product.

Example 3.8. When $S_1 \subseteq \mathbb{Z}_q^{d_1}$ is a structure-preserving set with noise growth δ_1 and $S_2 \subseteq \mathbb{Z}_q^{d_2}$ is a structure-preserving set with noise growth δ_2, then $S_1 \times S_2 \subseteq \mathbb{Z}_q^{d_1+d_2}$ is structure-preserving with noise $\max\{\delta_1, \delta_2\}$.

Proof. Let $\mathcal{D}_1, \mathsf{success}_1, \alpha_1$ be the distribution, abort function and abort constant that make S_1 $+$-structure-preserving with noise δ_1 and $\mathcal{D}_2, \mathsf{success}_2, \alpha_2$ be the distribution, abort function and abort constant that make S_2 structure-preserving with noise δ_2. Then the distribution $\mathcal{D}_1 \times \mathcal{D}_2$ with the success function $\mathsf{success}((\mathbf{m}_1, \mathbf{m}_2), (\mathbf{m}_1', \mathbf{m}_2'), \mathbf{d}) := \mathsf{success}_1(\mathbf{m}_1, \mathbf{m}_1', \mathbf{d}) \cdot \mathsf{success}_2(\mathbf{m}_2, \mathbf{m}_2', \mathbf{d})$ and success probability constant $\alpha := \alpha_1\alpha_2$ makes the set $S_1 \times S_2$ structure-preserving with noise $\max\{\delta_1, \delta_2\}$. □

We complete this section with an alternative formulation of the structure-preserving set property that is easier to use in some of the proofs.

Lemma 3.9. *For a structure-preserving set S with noise growth δ and smudging set D' we have $S + D' - D' \subseteq B_\delta(S)$.*

Proof. We prove this Lemma by contradiction. Suppose there exist $\mathbf{s} \in S$ and $\mathbf{d}, \mathbf{d}' \in \mathcal{D}$ such that $\mathbf{x} := \mathbf{s} + \mathbf{d} - \mathbf{d}' \notin B_\delta(S)$, i.e. $\mathbf{x} \in \overline{S}_\delta := \mathbb{Z}_q^d \setminus B_\delta(S)$. But then

$$S + D' \ni \mathbf{s} + \mathbf{d} = \mathbf{x} + \mathbf{d}' \in \overline{S}_\delta + D',$$

which is in contradiction to part (ii) of Definition 3.4. □

4 Lattice-Based Structure-Preserving Signatures

A lattice-based structure-preserving signature (SPS) scheme Σ expresses its verification algorithm in the framework of structure-preserving sets. Namely, a signature σ can be split into two separate parts $\sigma = (\mathsf{core}, \mathsf{tag})$. In order to verify that σ is valid, the Σ verification algorithm checks whether $f(\mathsf{core})$ belongs to a structure-preserving set S. The function f is publicly computable from tag, along with public verification key vk and the message m.

The requirement to use tag arises from specific properties of known lattice-based SPS schemes. The tag is publicly samplable and, for example, it could be a random string. At a technical level, the tag is usually required in all known lattice-based signatures that satisfy strong-unforgeability, and can remain unused in some schemes that are only existentially-unforgeable.

Definition 4.1 (Lattice SPS). *A* lattice-based $\mathcal{F}$-structure-preserving signature Σ *for a family $\mathcal{F}$ of functions $f : \mathbb{S} \to \mathbb{Z}_q^{d'}$ is a digital signature with signature space $\mathbb{S} \times \mathbb{T}$ where verification of a signature* $(\mathsf{core}, \mathsf{tag})$ *is functionally equivalent to returning*

$$f(\mathsf{core}) \in S$$

where $f \in \mathcal{F}$ and $S \subseteq \mathbb{Z}_q^{d'}$ are derived from vk, msg *and* tag. *Furthermore, S is a structure-preserving set. Finally, we require that tags are publicly samplable. That is, there exists an algorithm* TagGen *that, given the verification key* vk *and a message m generates a tag* tag *that has the same distribution as the tag part of the signatures generate with the signing algorithm.*

Remark 4.2. Since we do not require the membership problem for the sets S to be easy, this definition does not give immediately rise to an alternative verification procedure.

We are particularly interested in the cases where $\mathcal{F}$ is the set of linear functions or the set of functions that can be computed by bounded-depth Boolean circuits after encoding the signature as a binary string.

For structure-preserving signatures we require a slightly stronger security notion (defined below) than standard (strong) existential unforgability under chosen message attacks ((s)EUF-CMA). Compared to (s)EUF-CMA, we relax the verification of the forged signature.

Definition 4.3 (SPS-(s)EUF-CMA)**.** *A structure-preserving signature scheme* (KeyGen, Sign, Ver) *is* SPS-EUF-CMA *or* SPS-sEUF-CMA*-secure, if every PPT adversary can win the respective game in Fig. 1 with at most negligible probability.*

```
$(\mathsf{vk}, \mathsf{sk}) \leftarrow_R \mathsf{KeyGen}(1^\lambda)$
$Q := \emptyset$
$(m^\star, \mathsf{sig}^\star) \leftarrow_R \mathcal{A}^{\mathcal{O}_{\mathsf{sign}}}(\mathsf{vk})$
$b \leftarrow \mathsf{Ver}'(\mathsf{vk}, m^\star, \mathsf{sig}^\star)$
$\boxed{\textbf{return } b \wedge m^\star \notin Q}$
$\textbf{return } b \wedge (m^\star, \mathsf{sig}^\star) \notin Q$

$\underline{\mathcal{O}_{\mathsf{sign}}(m):}$
$\mathsf{sig} \leftarrow_R \mathsf{Sign}(\mathsf{sk}, \mathsf{msg})$
$\boxed{Q \leftarrow Q \cup \{m\}}$
$Q \leftarrow Q \cup \{(m, \mathsf{sig})\}$
$\textbf{return } \mathsf{sig}$

$\underline{\mathsf{Ver}'(\mathsf{vk}, m, \mathsf{sig} = (\mathsf{core}, \mathsf{tag})):}$
$\textbf{return } (f(\mathsf{core}) \in B_{\delta_S}(S))$
```

Fig. 1. Security experiment for $\boxed{\mathsf{SPS\text{-}EUF\text{-}CMA}}$ and $\mathsf{SPS\text{-}sEUF\text{-}CMA}$ security of lattice-based structure-preserving signatures.

4.1 SPS instantiation

Examples of structure-preserving signatures are Boyen's signature scheme [14], Rückert's signature scheme [49] and a new scheme, that combines the advantages of these two schemes. Namely, it achieves strong unforgeablity and has a simpler verification (because it does not need the non-zero signature check). Furthermore, it is more efficient (due to shorter signatures) than Rückert's scheme. We only show that the new scheme satisfies Definition 4.1 here and present the remaining details in Appendix A.

As a prerequisite, we state some facts that are needed in the signature scheme description, and define and construct chameleon hash functions.

Fact 1 ([14, Fact 5]) *There is a PPT algorithm* TrapGen *that, on input the security parameter λ, an odd prime $q = \mathsf{poly}(\lambda)$, and two integers $n = \Theta(\lambda)$ and $m \geq 6n \log q$, outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ statistically close to uniform, and a basis $\mathbf{T_A}$ for $\Lambda^\perp(\mathbf{A})$ such that $\|\bar{\mathbf{T}}_\mathbf{A}\| \leq \tilde{\Theta}(\sqrt{m}) \leq L$ with overwhelming probability. We assume $L = \tilde{\Omega}(\sqrt{m})$.*

Fact 2 ([14, Lemma 22]) *For a security parameter λ, let $q = \mathsf{poly}(\lambda)$ be an odd prime, $n = \Theta(\lambda)$, $m \geq 6n \log q$, $L = \tilde{\Omega}(\sqrt{m})$ and $\sigma \geq L\omega(\sqrt{\log m})$. Then there exist a PPT algorithm* SamplePre *that on input a Gaussian parameter σ, a modulus q, a matrix $\mathbf{F} := [\mathbf{A}|\mathbf{B}] \leftarrow_R \mathbb{Z}_q^{n\times 2m}$, and a basis $\mathbf{T_A} \subset \Lambda^\perp(\mathbf{A})$ of norm $\|\bar{\mathbf{T}}_\mathbf{A}\| \leq L$, and a vector $\mathbf{u}$, outputs $\mathbf{d} \in \Lambda^\perp(\mathbf{F})$ from the distribution $\mathcal{D}_{\mathbb{Z}^m,\sigma}$ conditioned on $\mathbf{Fd} = \mathbf{u}$.*

Fact 3 ([4, Section 4.2]) *Given matrices $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{B}$ needs to have rank n, a short basis $\mathbf{T_B}$ for $\mathbf{B}$ and a short matrix $\mathbf{R} \in \mathbb{Z}_q^{m\times m}$, one can compute efficiently a short basis $\mathbf{T_F}$ for $\mathbf{F} := (\mathbf{A}|\mathbf{AR} + \mathbf{B})$ with $\|\widetilde{\mathbf{T_F}}\| \leq \|\widetilde{\mathbf{T_B}}\|(\|\mathbf{R}\| + 1)$.*

Definition 4.4 (Chameleon hash function). *A* chameleon hash function *with message space $\mathcal{M}$ and hash space $\mathcal{N}$ consists of an efficiently samplable distribution $\mathcal{R}$ on some randomness space R and two PPT algorithms* (GenCH, TrapColl) *with the following syntax*

$\mathsf{GenCH}(1^\lambda)$**:** *Given an unary encoded security parameter λ as input, it outputs a chameleon hash function* ch $: \mathcal{M} \times R \to \mathcal{N}$ *and a trapdoor τ.*

$\mathsf{TrapColl}(\tau, m \in \mathcal{M}, r \in R, m^* \in \mathcal{M})$**:** *Given the trapdoor τ for a chameleon hash function* ch*, two messages m, m^* and one randomness r this algorithm outputs r^* such that $\mathsf{ch}(m, r) = \mathsf{ch}(m^*, r^*)$ and r^* is distributed according to $\mathcal{R}$.*

The security property we require for chameleon hash functions is *collision resistance*. That is, for every PPT adversary $\mathcal{A}$, the following probability is negligible

$$\Pr[(\mathsf{ch}, \tau) \leftarrow_R \mathsf{GenCH}, (m, r, m^*, r^*) \leftarrow_R \mathcal{A}(1^\lambda, \mathsf{ch}) : \mathsf{ch}(m, r) = \mathsf{ch}(m^*, r^*) \\ \wedge (m, r) \neq (m^*, r^*)].$$

An example of a chameleon hash function based on the SIS assumption is by [19]. It has message space $\mathcal{M} := \{0,1\}^k$ and randomness space $R := \{\mathbf{r} \in \mathbb{Z}^m \mid \|\mathbf{r}\| < s\sqrt{m}\}$ with a tail-truncated discrete Gaussian distribution $\mathcal{D}_{R,s}$ where $s = L \cdot \omega(\sqrt{\log m})$ and n, m, and L are as in Fact 1. It works as follows:

$\mathsf{GenCH}(1^\lambda)$ samples $\mathbf{A}_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times k}$ and $\mathbf{A}_1 \in \mathbb{Z}_q^{n\times m}$ with short basis $\mathbf{S}$ using $\mathsf{TrapGen}$. Output $\mathbf{A} := (\mathbf{A}_0|\mathbf{A}_1)$ to describe the chameleon hash function

$$\mathsf{ch}_{\mathbf{A}} : \{0,1\}^k \times R \to \mathbb{Z}_q^n$$
$$(\mathbf{m}, \mathbf{r}) \mapsto \mathbf{A} \cdot \begin{pmatrix} \mathbf{m} \\ \mathbf{r} \end{pmatrix}$$

$\mathsf{TrapColl}(\tau, \mathbf{m} \in \mathcal{M}, \mathbf{r} \in R, \mathbf{m}^* \in \mathcal{M})$ samples and outputs a vector $\mathbf{r}^*$ according to (a distribution statistically close to) $\mathcal{D}_{R,s}$ condition on $\mathsf{ch}_{\mathbf{A}}(\mathbf{m}^*, \mathbf{r}^*) = \mathsf{ch}_{\mathbf{A}}(\mathbf{m}, \mathbf{r})$ using Fact 2.

Lemma 4.5 ([19, Lemma 4.1]). *The above chameleon hash function is collision-resistant under the* $\mathsf{SIS}_{m,n,q,\beta}$ *problem where* $\beta := \sqrt{k + 4s^2 m}$.

The ISIS-based signature scheme uses a chameleon hash function $(\mathsf{GenCH}, \mathsf{TrapColl})$ with message space $\mathcal{M}$, randomness space R and hash space $\mathcal{N} = \{0,1\}^\ell$ and is described as follows:

$\mathsf{KeyGen}(1^\lambda)$**:** Given unary encoded security parameter λ as input, proceed as follows:

1. Execute the $\mathsf{TrapGen}$ algorithm to obtain a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a basis $\mathbf{T_A} \in \Lambda^\top(\mathbf{A})$ such that $\|\bar{\mathbf{T}}_{\mathbf{A}}\| \leq L$.
2. Sample $\mathbf{y} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$, $(\mathbf{C}_0, \dots, \mathbf{C}_\ell) \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m} \times \dots, \mathbb{Z}_q^{n\times m}$.
3. Sample $(\mathsf{ch}, \tau) \leftarrow_{\mathrm{R}} \mathsf{GenCH}(1^\lambda)$.
4. Output $\mathsf{vk} := (\mathbf{A}, \mathbf{C}_0, \dots, \mathbf{C}_\ell, \mathbf{y}, \mathsf{ch})$ and $\mathsf{sk} := \mathbf{T_A}$.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{msg})$**:** Given a signing key $\mathsf{sk} = \mathbf{T_A}$ and a message $\mathsf{msg} \in \mathcal{M}$ as input proceed as follows:

1. Sample $r \leftarrow \mathcal{R}$ and set $\mathsf{msg}' := \mathsf{ch}(\mathsf{msg}, r)$.
2. Compute $\mathbf{C}_{\mathsf{msg}} := \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}'_i \mathbf{C}_i$ and set $\mathbf{F}_{\mathsf{msg}} := [\mathbf{A} \mid \mathbf{C}_{\mathsf{msg}}] \in \mathbb{Z}_q^{n\times 2m}$.
3. Execute the algorithm $\mathsf{SamplePre}$ on $\mathbf{F}_{\mathsf{msg}}$, $\mathbf{T_A}$ and $\sigma \geq 2L\omega(\sqrt{\log m})$ to obtain a short non-zero random point $\mathbf{d}$ with $\mathbf{F}_{\mathsf{msg}}\mathbf{d} = \mathbf{y}$.
4. Output the signature $\mathsf{sig} := (\mathsf{core} = \mathbf{d}, \mathsf{tag} = r)$.

$\mathsf{Ver}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig})$**:** Given a verification key $\mathsf{vk} = (\mathbf{A}, \mathbf{C}_0, \dots, \mathbf{C}_\ell, \mathbf{y}, \mathsf{ch})$, a message $\mathsf{msg} \in \mathcal{M}$ and signature $\mathsf{sig} = (\mathbf{d} \in \mathbb{Z}_q^{2m}, r)$ as input, set $\mathsf{msg}' := \mathsf{ch}(\mathsf{msg}, r)$ and output 1 if (1) $\|\mathbf{d}\| \leq \sqrt{2m} \cdot \sigma$ and (2) $[\mathbf{A} \mid \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}'_i \mathbf{C}_i]\mathbf{d} = \mathbf{y} \mod q$. Otherwise, output 0.

Lemma 4.6. *The ISIS-based signature scheme from above is a SPS scheme.*

Proof. A signature sig is of the form $(\mathsf{core}, \mathsf{tag}) = (\mathbf{d}, r)$. Clearly, these tags are publicly samplable.

According to definition Definition 4.1, what remains to show is that the signature verification can be expressed as $f(\mathsf{core}) \in S$ for some function $f : \mathbb{Z}_q^{2m} \to \mathbb{Z}_q^{d'}$ and some set $S \subseteq \mathbb{Z}_q^{d'}$ which is structure-preserving. Both the function f and the set S might depend on the message being signed, the verification key and the public parameters of the scheme. We show that the signature verification can be expressed as two checks of the type $f_i(\mathsf{core}) \in S_i$ $(i \in \{1,2\})$. These check can then be combined to a single check by setting $f(\mathsf{core}) := (f_1(\mathsf{core}), f_2(\mathsf{core}))$ and $S := S_1 \times S_2$. The set S is structure-preserving when S_1 and S_2 are structure-preserving by Example 3.8.

The first check is $\|\mathsf{core}\| \leq \sqrt{2m} \cdot \sigma$, i.e., that core is a small vector. For this, we can set $n'_1 := 2m$ and

$$f_1(\mathsf{core}) := \mathsf{core}, \quad \text{and} \quad S_1 := \{\mathbf{x} \in \mathbb{Z}_q^{2m} \mid \|\mathbf{x}\| \leq \sqrt{2m} \cdot \sigma\} = B_{\sqrt{2m}\cdot\sigma}(\{0\}).$$

By triangular inequality, we have that $S_1 - S_1 \subseteq B_{2\sqrt{2m}\cdot\sigma}(\{0\})$. By Remark 3.7, we can conclude that S_1 is structure-preserving with noise growth $16m\sigma + 1$.

For the second check, we can set $n'_2 := n$ and

$$f_2(\mathsf{core}) := \left[\mathbf{A} \middle| \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i\right] \mathsf{core} \quad \text{and} \quad S_2 := \{\mathbf{y}\} \subset \mathbb{Z}_q^n.$$

Note that the function f_2 is defined by the message and the verification key. Moreover, S_2 is a singleton set and hence by Remark 3.3 and Lemma 3.5, we know that it is structure-preserving with noise growth 0. □

We prove SPS-sEUF-CMA-security of our scheme in Appendix A.

5 Lattice-Based Structure-Preserving Encryption

Our notion of a structure-preserving encryption (SPE) captures the common properties of known lattice-bases encryption schemes which are compatible with efficient lattice-based sigma protocols and NIZKs that prove statements about ciphertexts. In particular, the randomness space needs to be a structure-preserving set (Definition 3.4) and ciphertexts are of the form $\mathsf{ct} = \mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg})$, where $\mathbf{B}_\alpha$ is a public matrix depending on the message dimension α, and g_α is an invertible encoding function.

In addition, SPE needs to satisfy a series of technical properties on the noise, which provides bounds on the noise levels. This is a crucial property that allows for compatibility with the sigma protocols in later sections.

Definition 5.1 (Lattice SPE). *A PKE scheme* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a* lattice-based structure-preserving encryption *scheme if it satisfies the following properties:*

- *It has message space $\mathcal{M}^*$ for some base set $\mathcal{M}$. That is, we can encrypt arbitrary dimensional vectors of some base set $\mathcal{M}$. The ciphertexts will reveal the dimensions of the vectors.*
- *Public key: The public key implicitly defines matrices $(\mathbf{B}_\alpha \in \mathbb{Z}_q^{d(\alpha)\times r(\alpha)})_{\alpha\in\mathbb{N}_+}$ and efficiently sampleable distribution $(\mathcal{R}_\alpha)_{\alpha\in\mathbb{N}_+}$ such that $\mathbf{r} \leftarrow \mathcal{R}_\alpha$ lies with overwhelming probability in a structure-preserving set $R_\alpha \subseteq \mathbb{Z}_q^r$. The parameter α denotes the dimension of the message, i.e. to encrypt a message $\mathsf{msg} \in \mathcal{M}^\alpha$ we will use $\mathbf{B}_\alpha$ and $\mathcal{R}_\alpha$.*
- *Message encoding: The public key implicitly defines for every $\alpha \in \mathbb{N}_+$ an additively homomorphic invertible function $g_\alpha \colon \mathcal{M}^\alpha \to \mathbb{Z}_q^{d(\alpha)}$ such that Enc is equivalent to an algorithm that samples a vector $\mathbf{r} \leftarrow \mathcal{R}_\alpha$ and outputs $\mathsf{ct} = \mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg})$.*
- *Noise Levels: There exists a polynomial time algorithm $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct})$ that computes a noise level $\nu \in \mathbb{N}_0$ for each ciphertext and satisfies the following:*
  - *Initial noise level: For every security parameter λ there is a constant $\nu_{\mathsf{init}} \in \mathbb{N}_0$ such that for every key pair $(\mathsf{pk}, \mathsf{sk})$ in the range of $\mathsf{KeyGen}(1^\lambda)$ and every ciphertext ct in the range of $\mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$ for a message $\mathsf{msg} \in \mathcal{M}^\alpha$ we have $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}) \le \nu_{\mathsf{init}}$.*
  - *Maximum noise level: For every security parameter λ there is a constant $\nu_{\mathsf{max}} \ge 2\nu_{\mathsf{init}}$ such that for every key pair $(\mathsf{pk}, \mathsf{sk})$ in the range of $\mathsf{KeyGen}(1^\lambda)$ and every ciphertext $\mathsf{ct} = \mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg})$ with $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}) \le \nu_{\mathsf{max}}$ we have $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) = \mathsf{msg}$.*
  - *Symmetry: For every secret key sk and ciphertext ct*

    $$\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}) = \mathsf{NoiseLevel}(\mathsf{sk}, -\mathsf{ct}).$$

  - *Subadditivity: For every secret key sk and any two ciphertexts $\mathsf{ct}_1, \mathsf{ct}_2$ with $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}_1), \mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}_2) \le \nu_{\mathsf{max}}/2$ satisfy*

    $$\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}_1 + \mathsf{ct}_2) \le \mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}_1) + \mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}_2).$$

  - *Boundedness: For every security parameter λ there exists an efficiently computable function $\mathsf{MaxNoiseLevel} : \mathbb{N}_0 \to \mathbb{N}_0$ such that for every message dimension α and vector $\mathbf{r}$ of suitable length*

    $$\|\mathbf{r}\| < \delta \to \mathsf{NoiseLevel}(\mathsf{sk}, \mathbf{B}_\alpha \mathbf{r}) \le \mathsf{MaxNoiseLevel}(\delta)$$

    holds with overwhelming probability over the choice of the secret key sk.

Definition 5.2. *We say that a lattice-based SPE scheme is $\mathcal{F}$-homomorphic for a family of functions $\mathcal{F}$ if for all $f \in \mathcal{F}$, $f : \mathcal{M}^{\alpha_{\mathsf{in}}} \to \mathcal{M}^{\alpha_{\mathsf{out}}}$ when there exists a maximum noise level $\nu_{\mathsf{in}} \ge \nu_{\mathsf{init}}$ and a deterministic polynomial time algorithm Eval_f that takes pk and a ciphertext $\mathsf{ct} = \mathbf{B}_{\alpha_{\mathsf{in}}} \mathbf{r} + g_{\alpha_{\mathsf{in}}}(\mathsf{msg})$ that encrypts a α_{in}-dimensional message msg under pk with noise level $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}) \le \nu_{\mathsf{in}}$. It outputs a new ciphertext $\mathbf{B}_{\alpha_{\mathsf{out}}} \mathbf{r}_f + g_{\alpha_{\mathsf{out}}}(f(\mathsf{msg}))$ with $\mathbf{r}_f \in R_f$, where R_f is a structure-preserving set with noise growth δ_{R_f} such that every ciphertext $\mathsf{ct} = \mathbf{B}_{\alpha_{\mathsf{out}}} \mathbf{r} + g_{\alpha_{\mathsf{out}}}(\mathsf{msg})$ with $\mathbf{r} \in B_{\delta_{R_f}}(R_f)$ and $\mathsf{msg} \in \mathcal{M}^{\alpha_{\mathsf{out}}}$ has $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}) \le \nu_{\mathsf{max}}$.*

We further require that there is a deterministic polynomial time algorithm $\mathsf{Eval}_f^{\mathsf{rand}}$ that takes the public key pk and $\mathbf{r} \in R$ and outputs $\mathbf{r}_f$ such that

$$\mathbf{B}_{\alpha_{\mathsf{out}}} \mathbf{r}_f + g(f(\mathsf{msg})) = \mathsf{Eval}_f(\mathsf{pk}, \mathbf{B}_{\alpha_{\mathsf{in}}} \mathbf{r} + g(\mathsf{msg}))$$

Note that every SPE scheme is linearly homomorphic. In more detail, given two ciphertexts $\mathsf{ct}_1 = \mathbf{B}_\alpha \mathbf{r}_1 + g_\alpha(\mathsf{msg}_1)$ and $\mathsf{ct}_2 = \mathbf{B}_\alpha \mathbf{r}_2 + g_\alpha(\mathsf{msg}_2)$ with $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}_1)$, $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}_2) \leq \nu_{\mathsf{max}}/2$, the ciphertext $\mathsf{Eval}_+(\mathsf{pk}, \mathsf{ct}_1, \mathsf{ct}_2) := \mathsf{ct}_1 + \mathsf{ct}_2$ is a valid ciphertext for $\mathsf{msg}_1 + \mathsf{msg}_2$ with randomness $\mathsf{Eval}_f^{\mathsf{rand}}(\mathsf{pk}, \mathbf{r}_1, \mathbf{r}_2) := \mathbf{r}_1 + \mathbf{r}_2$, since g_α is additively homomorphic. This can be extended to linear functions (with sufficiently small coefficients) of multiple ciphertexts.

5.1 SPE instantiation

Examples of SPE schemes are Regev's encryption scheme, the Dual Regev encryption scheme and the GSW encryption scheme. We only prove that Regev's scheme is a SPE scheme here and present the proof for the remaining two schemes in Appendix B. As Regev's original scheme [48] allows to encrypt a single bit only, we recall its variant, put forward by Peikert et al. [47], that allows to encrypt messages from the message space $\mathcal{M} = \mathbb{Z}_p$ for p s.t. $\frac{q}{p}$ is sufficiently large. We assume that $q = p^k$, for a sufficiently large $k \in \mathbb{N}$, and we denote $c := \frac{q}{p} = p^{k-1}$. In addition to the LWE modulus q, the scheme is parametrized by a dimension n, number of samples $m \geq n \log q$ and an error distribution $\chi = \mathcal{D}_{\mathbb{Z},\sigma}$. We recall this scheme with $\alpha = 1$. To encrypt a higher-dimensional message $(\mathsf{msg}_1, \ldots, \mathsf{msg}_\alpha)^\top \in \mathcal{M}^\alpha$, we encrypt each component individually, i.e. generate $\mathsf{ct}_i = \mathsf{Enc}(\mathsf{pk}, \mathsf{msg}_i)$ for $i \in \{1, \ldots, \alpha\}$ and chain the ciphertext together, i.e. $\mathsf{ct}^\top = (\mathsf{ct}_1^\top, \ldots, \mathsf{ct}_\alpha^\top)$.

$\mathsf{KeyGen}(1^\lambda)$: Sample $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$ and $\mathbf{e} \leftarrow \chi^m$. Output the secret key $\mathsf{sk} := \mathbf{s}$ and the public key $\mathsf{pk} = (\mathbf{A}, \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^{1 \times m}$.

$\mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$: Parse pk as $(\mathbf{A}, \mathbf{x})$. Sample $\mathbf{z} \leftarrow_{\mathrm{R}} \{-1, 0, 1\}^m$ and compute $\mathbf{c}_0 := \mathbf{A}\mathbf{z} \in \mathbb{Z}_q^n$ and $c_1 := \mathbf{x}\mathbf{z} + c \cdot \mathsf{msg} \in \mathbb{Z}_q$. Then output the ciphertext $\mathsf{ct} := (\mathbf{c}_0, c_1) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: Parse ct as $(\mathbf{c}_0, c_1)$ and set $\mathbf{s} := \mathsf{sk}$. Compute $d := c_1 - \mathbf{s}^\top \mathbf{c_0} \in \mathbb{Z}_q$ and output $x \in \mathbb{Z}_p$, such that $d - c \cdot x \mod q$ is closest to 0.

Lemma 5.3. *Regev's encryption scheme is a lattice-based SPE scheme.*

Proof. For a public key $\mathsf{pk} = (\mathbf{A}, \mathbf{x}) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^{1 \times m}$, dimension α, and a message $\mathsf{msg} \in \mathcal{M}^\alpha$, let us define the matrix $\mathbf{B} \in \mathbb{Z}_q^{\alpha(n+1) \times \alpha m}$ and the function $g_\alpha \colon \mathcal{M}^\alpha \to \mathbb{Z}_q^{\alpha(n+1)}$ as follows :

$$\mathbf{B} := \mathbf{I}_\alpha \otimes \begin{pmatrix} \mathbf{A} \\ \mathbf{x} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & & \\ \mathbf{x} & & \\ & \ddots & \\ & & \mathbf{A} \\ & & \mathbf{x} \end{pmatrix}, \quad g_\alpha \begin{pmatrix} \mathsf{msg}_1 \\ \vdots \\ \mathsf{msg}_\alpha \end{pmatrix} := \begin{pmatrix} \mathbf{0} \\ c \cdot \mathsf{msg}_1 \\ \vdots \\ \mathbf{0} \\ c \cdot \mathsf{msg}_\alpha \end{pmatrix}.$$

Let $\mathcal{R}$ be the uniform distribution over $R := \{-1, 0, 1\}^{\alpha m}$. Clearly, $\mathbf{r} \leftarrow \mathcal{R}$ lies in R with probability 1. We need to show that R is a structure-preserving set. $R = \{-1, 0, 1\}^{\alpha m} \subseteq \mathbb{Z}_q^{\alpha m}$ is a $\sqrt{\alpha m}$-bounded set which, by Remark 3.7, implies that R is structure-preserving with noise growth $\delta_R := 8m + 1$.

As a next set, we need to argue that g is invertible and additively homomorphic. Let $g_\alpha^{-1} \colon \mathrm{Img}(g_\alpha) \to \mathbb{Z}_p$ be a function that on input $\mathbf{y} = (\mathbf{0}^\top, y_1, \ldots, \mathbf{0}^\top, y_\alpha)^\top \in \mathrm{Img}(g_\alpha)$, outputs $\mathbf{x} \in \mathbb{Z}_p^\alpha$, such that $y_i - cx_i \mod q = 0$ for all $i \in \{1, \ldots, \alpha\}$. It is easy to see that g^{-1} is the inverse of g. It is easy to see that g_α is additively homorphic, because it is composed of additively homomorphic functions.

Furthermore, we need to prove that the encryption algorithm is equivalent to sampling $\mathbf{r} \leftarrow \mathcal{R}_\alpha$ and computing $\mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg})$. For $\mathsf{msg} \in \mathbb{Z}_p^\alpha$ and $\mathbf{r} \leftarrow \mathcal{R}_\alpha$, we have, for $\mathbf{r}^\top = (\mathbf{r}_1^\top, \ldots, \mathbf{r}_\alpha^\top)$ with $\mathbf{r}_i \in \mathbb{Z}_q^m$,

$$\mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg}) = \begin{pmatrix} \mathbf{A}\mathbf{r}_1 \\ \mathbf{x}\mathbf{r}_1 + c \cdot \mathsf{msg}_1 \\ \vdots \\ \mathbf{A}\mathbf{r}_\alpha \\ \mathbf{x}\mathbf{r}_\alpha + c \cdot \mathsf{msg}_\alpha \end{pmatrix} = \begin{pmatrix} \mathsf{ct}_1 \\ \vdots \\ \mathsf{ct}_\alpha \end{pmatrix} = \mathsf{ct}$$

which shows that this procedure indeed gives us a well-distributed ciphertext.

Finally, we need to prove that the existence of the $\mathsf{NoiseLevel}(\mathsf{sk},\mathsf{ct})$ algorithm. Let us define $\mathsf{NoiseLevel}(\mathsf{sk},\mathsf{ct})$ as follows: Parse ct as $(\mathsf{ct}_1,\ldots,\mathsf{ct}_\alpha)$ and each ct_i as $(\mathbf{c}_{i,0},c_{i,1})$ and set $\mathbf{s} := \mathsf{sk}$. Compute $d_i := c_{1,i} - \mathbf{s}^\top \mathbf{c}_{i,0} \in \mathbb{Z}_q$ and $\nu_i := |d_i - c \cdot \mathsf{Dec}(\mathsf{sk},\mathsf{ct}_i)|$. Output $\max_{1\leq i \leq \alpha} \nu_i$.

To show that this definition satisfies the desired properties, it suffices to prove it for dimension $\alpha = 1$, because all these properties only talk about upper bounds[11] of the noise level and the noise level of a ciphertext for $\alpha > 1$ is simply the maximum of the noise levels of the ciphertexts for each component of the message.

To show boundedness, define $\mathsf{MaxNoiseLevel}(\delta) := 2\sigma\sqrt{m}\delta$. Then, for $\|\mathbf{z}\| < \delta$, we have

$$\mathsf{NoiseLevel}(\mathsf{sk},\mathsf{ct} = (\mathbf{A}\mathbf{z},((\mathbf{s}^\top\mathbf{A}+\mathbf{e}^\top)\mathbf{z}+c\mathsf{msg})) = |\mathbf{e}^\top\mathbf{z}| \overset{(1)}{\leq} \|\mathbf{e}\|\|\mathbf{z}\| \overset{(2)}{\leq} 2\sigma\sqrt{m}\delta,$$

where inequality (1) follows from the Cauchy-Schwartz inequality and inequality (2) follows from the Gaussian tail bound (Lemma 2.1).

The maximal initial noise level is $\nu_{\mathsf{init}} := 2\sigma m$: An honestly generated ciphertext has randomness $\mathbf{z} \in \{0,1\}^m$ and thus $\|\mathbf{z}\| \leq \sqrt{m}$. Plugging this in the $\mathsf{MaxNoiseLevel}$ function yields the desired bound.

The maximum noise level is $\nu_{\mathsf{max}} := \lceil c/2 \rceil$, because then for a ciphertext $\mathsf{ct} = (\mathbf{c}_0, c_1)$ for msg, the value $d := c_1 - \mathbf{s}^\top\mathbf{c_0}$ deviates at most by $\lceil c/2 \rceil$ from $c\mathsf{msg}$ and so the Dec algorithm will round to msg.

The Symmetry property of $\mathsf{NoiseLevel}$ follows immediately from the definition and the subadditivity property follows immediately from the triangle inequality. □

6 *Σ*-Protocol Constructions

In this section, we describe a generalization of the sigma protocols in [38] that, at a high level, allow to prove that the value encrypted in an SPE scheme belongs to a structure-preserving set S (up to an additional inherent error that comes from the noises of the encryption scheme and the structure-preserving set S).

More formally, we construct a trapdoor gap Σ-protocol that can prove for a lattice-based SPE scheme $\Pi = (\mathsf{KeyGen},\mathsf{Enc},\mathsf{Dec}^\star)$ that a ciphertext encrypts a message $\mathsf{msg} \in S$ where S is a structure-preserving set with noise growth δ_S and $B_{\delta_S}(S) \subseteq \mathcal{M}^\alpha$. Let:

- α be the dimension of the message in the ciphertext
- $\mathbf{B}_\alpha \in \mathbb{Z}_q^{d(\alpha)\times r(\alpha)}$ be the matrix defined by the public key for messages of length α,
- g_α be the message encoding function for messages of length α,
- R_α be the randomness space with maximum noise level ν_R (i.e. for all $\mathbf{r} \in R_\alpha$ and messages msg we have $\mathsf{NoiseLevel}(\mathsf{sk},\mathbf{B}_\alpha\mathbf{r} + g_\alpha(\mathsf{msg})) \leq \nu_R$). We also require R_α to be additively structure-preserving with noise growth δ_R using the distribution $\mathcal{D}_R$, smudging set D'_R, no-abort function $\mathsf{success}_R$ and no-abort constant α_R.
- S be a structure-preserving set with noise growth δ_S using distribution $\mathcal{D}_S$, smudging set D'_S with $S, D'_S, S + D'_S \subseteq \mathcal{M}$, no-abort function $\mathsf{success}_S$ and no-abort constant α_S,
- $\mathbf{r}' \in R_\alpha$ be a fixed representative of R_α,
- and $\mathsf{msg}' \in S$ be a fixed representative of S.
- And assume that the parameters of the SPE scheme are selected such that

$$\nu_{\mathsf{init}} + \nu_R + \mathsf{MaxNoiseLevel}(\delta_R) < \nu_{\mathsf{max}}/2. \tag{1}$$

We construct a gap Σ-protocol for:

$$\begin{aligned}\mathcal{L}_{\mathsf{zk}} &= \{\mathbf{B}_\alpha\mathbf{r} + g_\alpha(\mathsf{msg}) \mid \mathbf{r} \in R_\alpha, \mathsf{msg} \in S\}\\ \mathcal{L}_{\mathsf{sound}} &= \{\mathsf{ct} \mid \mathsf{NoiseLevel}(\mathsf{sk},\mathsf{ct}) \leq 2\cdot\nu_{\mathsf{init}} + \nu_R + 2\cdot\mathsf{MaxNoiseLevel}(\delta_R),\\ &\qquad \mathsf{Dec}(\mathsf{sk},\mathsf{ct}) \in B_{\delta_S}(S)\}\end{aligned}$$

From the SPE definition we get $\mathcal{L}_{\mathsf{zk}} \subseteq \mathcal{L}_{\mathsf{sound}}$.

[11] Note that the symmetry property is equivalent to $\mathsf{NoiseLevel}(\mathsf{sk},\mathsf{ct}) \leq \mathsf{NoiseLevel}(\mathsf{sk},-\mathsf{ct})$.

The language is described by the modulus q, the matrix $\mathbf{B}_\alpha$ and the structure-preserving sets R_α and S and the message encoding function g_α. The Setup algorithm will output as crs simply the language description, i.e. $\mathsf{crs} = (q, \mathbf{B}_\alpha, R_\alpha, S, g_\alpha)$. The membership testing trapdoor for the language is the secret key sk of the structure-preserving encryption scheme and TrapSetup will simply output as trapdoor this secret key, i.e. $\tau_\Sigma = \mathsf{sk}$. The definition of the prover and verfier can be found in Fig. 2.

```
Prover $\mathsf{P} = (\mathsf{P}_1, \mathsf{P}_2)$                                   Verifier $\mathsf{V}$

Input: $(\mathsf{crs} = (q, \mathbf{B}_\alpha, R_\alpha, S, g_\alpha),$          Input: $(\mathsf{crs} = (q, \mathbf{B}_\alpha, R_\alpha, S, g_\alpha), x)$
$x = \mathbf{Br} + g(\mathsf{msg}), w = \mathbf{r})$

$\mathbf{r}_R \leftarrow \mathcal{D}_R;\ m_S \leftarrow \mathcal{D}_S$
$\mathsf{a} := \mathbf{B}_\alpha \mathbf{r}_R + g_\alpha(m_S)$          $\xrightarrow{\mathsf{a}}$
                                              $\xleftarrow{\mathsf{Chal}}$     $\mathsf{Chal} \leftarrow_R \{0,1\}$
if $\mathsf{Chal} = 0$ then
    $\mathsf{z} := \mathbf{r}_R$
else
    $\mathsf{z} := \mathbf{r} + \mathbf{r}_R$
$\theta_1 := \mathsf{success}_R(\mathbf{r}, \mathsf{Chal} \cdot \mathbf{r}' + (1 - \mathsf{Chal}) \cdot \mathbf{r}, \mathbf{r}_R)$
Abort with probability $1 - \theta_1$
$\theta_2 := \mathsf{success}_S(\mathsf{msg}, \mathsf{Chal} \cdot \mathsf{msg}' + (1 - \mathsf{Chal}) \cdot \mathsf{msg}, m_S)$
Abort with probability $1 - \theta_2$          $\xrightarrow{\mathsf{z}}$
                                              if $\mathsf{Chal} = 0$ then
                                                  Output: $\mathsf{z} \in D'_R \wedge g_\alpha^{-1}(\mathsf{a} - \mathbf{B}\mathsf{z}) \in D'_S$
                                              else
                                                  Output: $\mathsf{z} \in R_\alpha + D'_R \wedge g_\alpha^{-1}(\mathsf{a} + x - \mathbf{B}\mathsf{z}) \in S + D'_S$
```

Fig. 2. The interaction between Prover and Verfier in our Σ-protocol.

Theorem 6.1. *The above construction is a trapdoor gap Σ-protocol for $(\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$.*

Proof. **Completeness:** Suppose that $\mathbf{r}_R \in D'_R$ and $m_S \in D'_S$. Both of these events happens with overwhelming probability by the second part of the structure-preserving set definition. Given this, it is easy to verify that the protocol accepts for both $\mathsf{Chal} = 0$ and $\mathsf{Chal} = 1$ when $x \in \mathcal{L}_{\mathsf{zk}}$.

Special Soundness: Suppose that for a statement x and a first flow message a there exist responses z_0 and z_1 that an honest verifier accepts for challenge $\mathsf{Chal} = 0$ resp. $\mathsf{Chal} = 1$. Then

$$\mathsf{z}_0 \in D'_R \tag{2}$$

$$\mathsf{z}_1 \in D'_R + R_\alpha \tag{3}$$

$$g_\alpha^{-1}(\mathsf{a} - \mathbf{B}_\alpha \mathsf{z}_0) \in D'_S \tag{4}$$

$$g_\alpha^{-1}(x + \mathsf{a} - \mathbf{B}_\alpha \mathsf{z}_1) \in D'_S + S \tag{5}$$

holds. By subtracting Eq. (4) from Eq. (5) and using the additive homomorphism of g_α, we get

$$g_\alpha^{-1}(x + \mathsf{a} - \mathbf{B}_\alpha \mathsf{z}_1 - (\mathsf{a} - \mathbf{B}_\alpha \mathsf{z}_0)) = g_\alpha^{-1}(x - \mathbf{B}_\alpha(\mathsf{z}_1 - \mathsf{z}_0)) \in S + D'_S - D'_S \subseteq B_{\delta_S}(S),$$

where the last relation follows using Lemma 3.9. Since we also have $\mathsf{z}_1 - \mathsf{z}_0 \in R_\alpha + D'_R - D'_R \subseteq B_{\delta_R}(R_\alpha)$ (again using Lemma 3.9) this proves $x \in \{\mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg}) \mid \mathbf{r} \in B_{\delta_R}(R_\alpha), \mathsf{msg} \in B_{\delta_S}(S)\} \subseteq \{\mathsf{ct} \mid \mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct}) \leq \nu_R +$

$\mathsf{MaxNoiseLevel}(\delta_R)$, $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \in B_{\delta_S}(S)\} \subseteq \mathcal{L}_{\mathsf{sound}}$. For the first subset relationship we use that we can write $\mathbf{r} = \mathbf{r}' + \mathbf{y}$ with $\mathbf{r}' \in R_\alpha$ and $\|\mathbf{y}\| \leq \delta_R$ since $\mathbf{r} \in B_{\delta_R}(R_\alpha)$. The statement then follows from using $\mathsf{NoiseLevel}(\mathsf{sk}, \mathbf{B}_\alpha \mathbf{r}' + g_\alpha(\mathsf{msg})) \leq \nu_R$, $\mathsf{NoiseLevel}(\mathsf{sk}, \mathbf{B}_\alpha \mathbf{y}) \leq \mathsf{MaxNoiseLevel}(\delta_R)$ (boundedness property of the $\mathsf{NoiseLevel}$ function) and combining this with the subadditivity property of the $\mathsf{NoiseLevel}$ function, which we can use due to Eq. (1).

Special Zero-Knowledge: We show that there exists a zero-knowledge simulator, that outputs statistically close transcripts and has statistically close aborting behavior as the real protocol. The simulator ZKSim works as follows on input ($\mathsf{crs} = (q, \mathbf{B}_\alpha, R_\alpha, S, g_\alpha), x \in \mathcal{L}_{\mathsf{zk}}, \mathsf{Chal}^\star \in \{0, 1\}$):

1. Sample $\mathbf{r}_R^\star \leftarrow \mathcal{D}_R$; $m_S^\star \leftarrow \mathcal{D}_S$.
2. Compute $\mathsf{a}^\star := \mathbf{B}_\alpha \mathbf{r}_R^\star + \mathsf{Chal}^\star(\mathbf{B}_\alpha \mathbf{r}' + g_\alpha(\mathsf{msg}') - x) + g_\alpha(m_S^\star)$.
3. Compute $\mathsf{z}^\star := \mathbf{r}_R^\star + \mathsf{Chal}^\star \cdot \mathbf{r}'$.
4. Abort with probability $1 - \alpha_R$.
5. Abort with probability $1 - \alpha_S$.
6. Output $(\mathsf{a}^\star, \mathsf{z}^\star)$.

For $x \in \mathcal{L}_{\mathsf{zk}}$, we have $x = \mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg})$ for $\mathbf{r} \in R_\alpha$ and $\mathsf{msg} \in S$.

First, we will focus on the case $\mathsf{Chal}^\star = 0$. In the real protocol, the randomness $\mathbf{r}_R$ of the first flow a is sampled from $\mathcal{D}_R$ and the protocol continues with probability $\theta_1 := \mathsf{success}_R(\mathbf{r}, \mathbf{r}, \mathbf{r}_R)$. The zero-knowledge simulator samples the first flow randomness from the same distribution, but continues with probability α_R. We use now that R_α is a structure-preserving set. The first part of the structure-preserving set definition with $\mathbf{r}$ and $\mathbf{r}$ guarantees that the distribution of the first flow randomness in real and the simulated protocol is statistically close.

Similarly, the distribution of the message part of the first flow is $\mathcal{D}_S$ both in the real protocol and the simulated one, but the real protocol continues with probability $\theta_2 := \mathsf{success}_S(\mathsf{msg}, \mathsf{msg}, m_S)$ while the simulated one continues with probability α_S. By using that S is a structure-preserving and plugging in msg and msg in the first part of the definition, it follows that the distribution of the first flow message in real and the simulated protocol is statistically close.

Next, we will discuss the remaining case $\mathsf{Chal}^\star = 1$. In the real protocol, the randomness part $\mathbf{r}_R$ of the first flow a is sampled again from $\mathcal{D}_R$ and the protocol continues with probability $\theta_1 := \mathsf{success}_R(\mathbf{r}, \mathbf{r}', \mathbf{r}_R)$. The simulated protocol samples $\mathbf{r}_R^\star \leftarrow \mathcal{D}_R$ and uses $\mathbf{r}_R^\star + \mathbf{r}' - \mathbf{r}$ as randomness and continues with probability α_R. We use again that R_α is a structure-preserving set, but plug in $\mathbf{r}$ and $\mathbf{r}'$ in the first part of the structure-preserving set definition. This gives us that outputting $\mathbf{r} + \mathbf{r}_R$ with probability $\mathsf{success}_R(\mathbf{r}, \mathbf{r}', \mathbf{r}_R)$ is statistically close to outputting $\mathbf{r}_R^\star + \mathbf{r}'$ with probability α_R.

The message part of the first flow is m_S, sampled from $\mathcal{D}_S$ in the real protocol and the protocol aborts with probability $\mathsf{success}_S(\mathsf{msg}, \mathsf{msg}', m_S)$. The simulator samples $m_S^\star \leftarrow \mathcal{D}_S$ and uses $\mathsf{msg}' - \mathsf{msg} + m_S^\star$ as message part of the first flow. Furthermore, the simulator aborts with probability α_S. Using that S is a structure-preserving set and plugging in msg and msg' in the first part of the definition, we get that these two distributions are also statistically close.

Putting this together, we see that the simulated first flow is statistically close to an honest first flow. And the third flow outputted by ZKSim is always the correct third flow with respect to the first flow and challenge, so ZKSim is a correct simulator. Furthermore, the zero knowledge simulator only aborts with a constant probability, so the real protocol also aborts only with constant probability.

Correctness of $\mathsf{BadChallenge}$**:** We show that the following $\mathsf{BadChallenge}$ algorithm outputs for any $x \notin \mathcal{L}_{\mathsf{sound}}$ a bad challenge. The $\mathsf{BadChallenge}$ algorithm proceeds on input ($\tau_\Sigma = \mathsf{sk}, \mathsf{crs}, x, \mathsf{a}$) as follows:

1. If $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{a}) > \nu_{\mathsf{init}} + \mathsf{MaxNoiseLevel}(\delta_R) \vee \mathsf{Dec}(\mathsf{sk}, \mathsf{a}) \notin D'_S$, output $\mathsf{Chal} = 0$ (indicating that the first flow is invalid).
2. Otherwise, if $\mathsf{NoiseLevel}(\mathsf{sk}, x + \mathsf{a}) > \nu_{\mathsf{init}} + \nu_R + \mathsf{MaxNoiseLevel}(\delta_R) \vee \mathsf{Dec}(\mathsf{sk}, x + \mathsf{a}) \notin S + D'_S$, output $\mathsf{Chal} = 1$.
3. Otherwise, output $\perp$.

First, assume that $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{a}) > \nu_{\mathsf{init}} + \mathsf{MaxNoiseLevel}(\delta_R)$ or $\mathsf{Dec}(\mathsf{a}) \notin D'_S$ holds. Then a can not be written as $\mathsf{a} = \mathbf{B}_\alpha \mathbf{r}_R + g_\alpha(m_S)$ with $\mathbf{r}_R \in D'_R, m_S \in D'_S$ because then it would have both of the above properties. In this scenario there is no third flow that would make the Verifier accept for $\mathsf{Chal} = 0$, so the $\mathsf{BadChallenge}$ correctly returns 0.

Second, assume that $\mathsf{NoiseLevel}(\mathsf{sk}, x + \mathsf{a}) > \nu_{\mathsf{init}} + \nu_R + \mathsf{MaxNoiseLevel}(\delta_R)$ or $\mathsf{Dec}(x + \mathsf{a}) \notin S + D'_S$ holds. Then $x + \mathsf{a}$ can not be written as $x + \mathsf{a} = \mathbf{B}_\alpha \mathbf{r} + g_\alpha(\mathsf{msg})$ with $\mathbf{r} \in R_\alpha + D'_R, \mathsf{msg} \in S + D'_S$ because then it would have both of the above properties. In this scenario there is no third flow that would make the Verifier accept for $\mathsf{Chal} = 1$, so the $\mathsf{BadChallenge}$ correctly returns 1 (if the first case does not apply as well).

Finally, assume that neither of the two cases above applies. Then

$$\begin{aligned}\mathsf{NoiseLevel}(\mathsf{sk}, x) &= \mathsf{NoiseLevel}(\mathsf{sk}, x + \mathsf{a} - \mathsf{a}) \\ &\leq \mathsf{NoiseLevel}(\mathsf{sk}, x + \mathsf{a}) + \mathsf{NoiseLevel}(\mathsf{sk}, -\mathsf{a}) \\ &= \mathsf{NoiseLevel}(\mathsf{sk}, x + \mathsf{a}) + \mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{a}) \\ &\leq 2 \cdot \nu_{\mathsf{init}} + \nu_R + 2 \cdot \mathsf{MaxNoiseLevel}(\delta_R).\end{aligned}$$

The inequality follows from subadditivity of the NoiseLevel-function which we can use due to Eq. (1). This guarantees that

$$\mathsf{Dec}(\mathsf{sk}, x) = \mathsf{Dec}(\mathsf{sk}, x + \mathsf{a}) - \mathsf{Dec}(\mathsf{sk}, \mathsf{a}) \in S + D'_S - D'_S \subseteq B_{\delta_S}(S)$$

which shows that $x \in \mathcal{L}_{\mathsf{sound}}$, in contradiction to our initial assumption. □

7 Lattice-Based Structure-Preserving NIZK Arguments

Definition 7.1 (Structure-Preserving NIZK (SPNIZK) Argument). *Let S be a structure-preserving set with noise growth δ_S and* SPE *be a structure-preserving public key encryption scheme with message space $\mathcal{M}^\alpha$ and randomness distribution $\mathcal{R}_\alpha$, where $\mathbf{r} \leftarrow_{\mathrm{R}} \mathcal{R}$ lies with overwhelming probability in a structure-preserving set $R_\alpha \subseteq \mathbb{Z}_q^r$ with noise growth δ_R. A NIZK argument system $(\mathsf{Gen}_{\mathsf{par}}, \mathsf{Gen}_{\mathcal{L}}, \mathsf{P}, \mathsf{V})$ is a structure-preserving NIZK (SPNIZK) with respect to S and* SPE *if for any $(\mathsf{pk}, \cdot) \leftarrow \mathsf{SPE.Setup}(1^\lambda)$, encryption randomness $\mathbf{r} \leftarrow_{\mathrm{R}} \mathcal{R}$ and $m \in S$,* SPNIZK *supports the following functionality:*

- $\mathsf{ProveMembershipS}_S(\mathsf{crs}, \mathsf{pk}, m, \mathsf{ct}, \mathbf{r})$ *outputs a proof π that* ct *encrypts a message m which belongs to the structure-preserving set S.*
- $\mathsf{VerifyMembershipS}_S(\mathsf{crs}, \mathsf{pk}, \mathsf{ct}, \pi)$ *verifies that* ct *indeed encrypts a message m which belongs to the structure-preserving set S.*

As in Definition 2.10, the SPNIZK must satisfy completeness, computational soundness, and zero-knowledge. Moreover, we require our SPNIZK argument system to satisfy unbounded simulation soundness [23, 51]. We refer the reader to Appendix C for the definition of these properties.

We discuss how to compile the sigma protocol from Section 6 into an SPNIZK argument with unbounded simulation soundness and multi-theorem zero-knowledge in Appendix D.

8 Verifiably Encrypted Signatures (VES)

Using a verifiable encrypted signature (VES), a signer can encrypt a signature under the public key of a trusted-third party (the *adjudicator*) and then generate a proof that the ciphertext encrypts a valid signature for a known message.

The main application of VES is online contract signing, in which two parties Alice and Bob agree on a contract by using the help of a trusted third party called an adjudicator. Alice and Bob start the protocol by producing a VES Ω_{Alice}, Ω_{Bob} on the agreed contract m, using the public key apk of the adjudicator. Upon receipt of the VES Ω_{Alice}, Ω_{Bob}, both Alice and Bob reveal the unencrypted versions $\sigma_{\text{Alice}}, \sigma_{\text{Bob}}$ of their signatures, agreeing to the contract. If any one of the parties, for example Bob, refuses to release his signature σ_{Bob}, Alice can contact the adjudicator and ask them to extract σ_{Bob} from Ω_{Bob}. This prevents Bob from not completing the protocol and using σ_{Alice} to negotiate a better contract elsewhere.

We recall the formal definition of VES in Appendix F. We discuss it here only informally. A VES is a tuple of PPT algorithms $(\mathsf{Kg}, \mathsf{AdjKg}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf})$, where Kg, Sig and Vf are defined similarly to a digital signature scheme. AdjKg generates a key pair $(\mathsf{apk}, \mathsf{ask})$ for the adjudicator, Create computes a VES on a given message,

	Our ISIS-based signature scheme	Rückert's scheme	Boyen's scheme
Regev	✓	✓	✗
Dual Regev	✓	✓	✗
GSW	✓	✓	✓

Table 1. The table indicates which of the SPE schemes can be combined with which SPS scheme to obtain VES.

and VesVf allows to verify that a given VES is a encryption of a valid signature on a given message. In addition to completeness, VES is required to satisfy four security properties: unforgeability, abuse freeness, extractability and opacity.

Unforgeability guarantees that no PPT adversary given the public key and oracle access Create and Adj, is able to compute a VES Ω for a message m that they have never queried to its oracles. Abuse freeness requires that no malicious, PPT adjudicator with access to a Create oracle is able to output a valid VES for a message that they have never queried. Extractability requires that no malicious signer which can create their own vk and is granted oracle access to Adj is able to efficiently output a valid VES Ω, from which the Adj algorithm is unable to extract a valid signature. Opacity requires that no PPT adversary, given public keys vk and apk and oracle access to Create and Adj, can return a valid signature σ^* for some message m^*, provided it has not queried Adj on m^*.

8.1 The VES Construction

We are now ready to show how to use our notions of structure-preserving signatures, encryptions and NIZK arguments to obtain verifiably encrypted signatures. Our construction is given in Fig. 3 and informally discussed below.

The starting point of our construction is any structure-preserving SPS (see Definition 4.1), over a modulus q. Recall that signatures are tuples $\sigma = (\mathsf{core}, \mathsf{tag})$, which consist of a vector $\mathsf{core} \in \mathbb{Z}_q^\gamma$ and a public string $\mathsf{tag} \in \{0,1\}^\varsigma$. To compute a VES Ω, we encrypt the core part of the signature core and obtain a ciphertext ct^1. The public tag is not encrypted, and is revealed together with ct^1 as part of Ω.

If we stop at this point, the verifier has no way of checking if core is valid, as it is only given in its encrypted form. Therefore, we now want to convince the verifier that the ciphertexts encrypt a vector core that is part of a valid signature. To this end, we first compute efficiently the structure-preserving set and function (S, f) that correspond to signature verification in the sense of Definition 4.1. Note that in our notation, f is a function that takes γ inputs and outputs a vector in $\mathbb{Z}_q^\tau$. We then compute ciphertexts ct^2 that correspond to homomorphic evaluation using function f over ct^1. Then, we use our SPNIZK argument to compute a proof π that ct^2 actually encrypts a vector that belongs to the structure-preserving set S. The resulting VES is hence $\Omega = (\mathsf{ct}^1, \pi, \mathsf{tag})$.

We can combine an SPE scheme with an SPS scheme if the SPE scheme is $\mathcal{F}$-homomorphic where $\mathcal{F}$ is the set of all functions f that can appear in the signature verification procedure in the sense of Definition 4.1. Table 1 summarizes which SPE scheme can be combined with which SPS scheme.

Verification is now straightforward. Namely, we recompute (S, f) using vk, m and the public tag, and check that the SPNIZK proof π is indeed valid. Finally, adjudication is performed by simply decrypting ciphertexts ct^1 and revealing the vector core.

We present the concrete parameters of our VES scheme in Appendix E and refer the reader to Appendix G for the security proof.

9 Acknowledgements

All authors were partially supported by ERC PREP-CRYPTO Grant Agreement ID 724307.

References

1. Abe, M., Fuchsbauer, G., Groth, J., Haralambiev, K. & Ohkubo, M. *Structure-Preserving Signatures and Commitments to Group Elements* in *CRYPTO 2010* (2010).

Generic Construction of a Verifiable Encrypted Signature Scheme VES based on any Structure-Preserving Signature SPS

```
$\mathsf{VES.Kg}(1^\lambda)$:
    Return $(\mathsf{vk}, \mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{SPS.KeyGen}(1^\lambda)$.

$\mathsf{VES.Sig}(\mathsf{sk}, m)$:
    Return $\sigma \leftarrow_{\mathrm{R}} \mathsf{SPS.Sign}(\mathsf{sk}, m)$.

$\mathsf{VES.Ver}(\mathsf{vk}, m, \sigma)$:
    Return $(\mathsf{SPS.Ver}(\mathsf{vk}, m, \sigma) \overset{?}{=} 1)$.

$\mathsf{VES.AdjKg}(1^\lambda)$:
    Return $(\mathsf{apk}, \mathsf{ask}) \leftarrow_{\mathrm{R}} \mathsf{SPE.KeyGen}(1^\lambda)$.

$\mathsf{VES.Create}(\mathsf{sk}, \mathsf{apk}, m)$:
    $\sigma = (\mathsf{core}, \mathsf{tag}) \leftarrow_{\mathrm{R}} \mathsf{SPS.Sig}(\mathsf{sk}, m) \in \mathbb{Z}_q^\gamma \times \{0,1\}^\zeta$
    $\mathbf{r}^1 \leftarrow_{\mathrm{R}} \mathcal{R}_\gamma$
    $\mathsf{ct}^1 \leftarrow \mathsf{SPE.Enc}(\mathsf{apk}, \mathsf{core}; \mathbf{r}^1)$
    $(S, f) \leftarrow \mathsf{ComputeSPSetsAndFunctions}(\mathsf{vk}, m, \mathsf{tag})$
    $\mathsf{val} \leftarrow f(\mathsf{core}) \in \mathbb{Z}_q^\tau$
    $\mathsf{ct}^2 \leftarrow \mathsf{Eval}_f(\mathsf{apk}, \mathsf{ct}^1)$
    $\mathbf{r}^2 \leftarrow \mathsf{Eval}_f^{\mathsf{rand}}(\mathsf{apk}, \mathbf{r}^1)$
    $\pi \leftarrow_{\mathrm{R}} \mathsf{SPNIZK.ProveMembershipS}_S(\mathsf{crs}, \mathsf{apk}, \mathsf{val}, \mathsf{ct}^2, \mathbf{r}^2)$
    Return $\Omega \leftarrow (\mathsf{ct}^1, \pi, \mathsf{tag})$

$\mathsf{VES.VesVf}(\mathsf{apk}, \mathsf{vk}, \Omega, m)$:
    Parse $\Omega$ as $(\mathsf{ct}^1, \pi, \mathsf{tag})$
    $(S, f) \leftarrow \mathsf{ComputeSPSetsAndFunctions}(\mathsf{vk}, m, \mathsf{tag})$
    $\mathsf{ct}^2 \leftarrow \mathsf{Eval}_f(\mathsf{apk}, \mathsf{ct}^1)$
    If $\mathsf{SPNIZK.VerifyMembershipS}_S(\mathsf{crs}, \mathsf{apk}, \mathsf{ct}^2, \pi) = 0$, then return $0$
        Else, return $1$

$\mathsf{VES.Adj}(\mathsf{ask}, \mathsf{apk}, \mathsf{vk}, \Omega, m)$:
    Parse $\Omega$ as $(\mathsf{ct}^1, \pi, \mathsf{tag})$
    $(S, f) \leftarrow \mathsf{ComputeSPSetsAndFunctions}(\mathsf{vk}, m, \mathsf{tag})$
    $\mathsf{ct}^2 \leftarrow \mathsf{Eval}_f(\mathsf{apk}, \mathsf{ct}^1)$
    If $\mathsf{SPNIZK.VerifyMembershipS}_S(\mathsf{crs}, \mathsf{apk}, \mathsf{ct}^2, \pi) = 0$, then return $\perp$
    $\mathsf{core}_i \leftarrow \mathsf{SPE.Dec}(\mathsf{ask}, \mathsf{ct}_i^1)$
    Return $\sigma = (\mathsf{core}, \mathsf{tag})$
```

Fig. 3. A verifiably-encrypted signature (VES) scheme $(\mathsf{Kg}, \mathsf{AdjKg}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf})$. SPS denotes a structure-preserving signature scheme, while SPE is a lattice-based structure-preserving encryption. SPNIZK is a structure-preserving NIZK argument for SPE, allowing to prove that encryptions encode plaintexts that belong to a structure-preserving set S. The parameters of SPS, SPE and SPNIZK are described in Appendix E.

2. Abe, M., Groth, J., Haralambiev, K. & Ohkubo, M. *Optimal Structure-Preserving Signatures in Asymmetric Bilinear Groups* in *CRYPTO 2011* (2011).
3. Abe, M. *et al. Constant-Size Structure-Preserving Signatures: Generic Constructions and Simple Assumptions* in *ASIACRYPT 2012* (2012).
4. Agrawal, S., Boneh, D. & Boyen, X. *Efficient Lattice (H)IBE in the Standard Model* in *EUROCRYPT 2010* (2010).
5. Agrawal, S., Boneh, D. & Boyen, X. *Lattice Basis Delegation in Fixed Dimension and Shorter-Ciphertext Hierarchical IBE* in *CRYPTO 2010* (2010).
6. Applebaum, B., Cash, D., Peikert, C. & Sahai, A. *Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems* in *CRYPTO 2009* (2009).
7. Asokan, N., Shoup, V. & Waidner, M. *Optimistic Fair Exchange of Digital Signatures (Extended Abstract)* in *EUROCRYPT'98* (1998).
8. Attema, T., Lyubashevsky, V. & Seiler, G. *Practical Product Proofs for Lattice Commitments* in *CRYPTO 2020, Part II* (2020).
9. Beame, P., Cook, S. & Hoover, H. *Log Depth Circuits For Division And Related Problems* in *25th Annual Symposium on Foundations of Computer Science, 1984.* (1984).
10. Belenkiy, M., Chase, M., Kohlweiss, M. & Lysyanskaya, A. *P-signatures and Noninteractive Anonymous Credentials* in *TCC 2008* (2008).
11. Bellare, M., Micciancio, D. & Warinschi, B. *Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions* in *EUROCRYPT 2003* (2003).
12. Blazy, O. & Chevalier, C. *Structure-Preserving Smooth Projective Hashing* in *ASIACRYPT 2016, Part II* (2016).
13. Boneh, D., Gentry, C., Lynn, B. & Shacham, H. *Aggregate and Verifiably Encrypted Signatures from Bilinear Maps* in *EUROCRYPT 2003* (2003).
14. Boyen, X. *Lattice Mixing and Vanishing Trapdoors: A Framework for Fully Secure Short Signatures and More* in *PKC 2010* (2010).
15. Camenisch, J., Haralambiev, K., Kohlweiss, M., Lapon, J. & Naessens, V. *Structure Preserving CCA Secure Encryption and Applications* in *ASIACRYPT 2011* (2011).
16. Camenisch, J. & Shoup, V. *Practical Verifiable Encryption and Decryption of Discrete Logarithms* in *CRYPTO 2003* (2003).
17. Canetti, R., Goldreich, O. & Halevi, S. The Random Oracle Methodology, Revisited. *J. ACM* (2004).
18. Canetti, R. *et al. Fiat-Shamir: from practice to theory* in *51st ACM STOC* (2019).
19. Cash, D., Hofheinz, D., Kiltz, E. & Peikert, C. *Bonsai Trees, or How to Delegate a Lattice Basis* in *EUROCRYPT 2010* (2010).
20. Cathalo, J., Libert, B. & Yung, M. *Group Encryption: Non-interactive Realization in the Standard Model* in *ASIACRYPT 2009* (2009).
21. Chase, M. & Kohlweiss, M. *A New Hash-and-Sign Approach and Structure-Preserving Signatures from DLIN* in *SCN 12* (2012).
22. Cramer, R., Damgård, I. & Schoenmakers, B. *Proofs of Partial Knowledge and Simplified Design of Witness Hiding Protocols* in *CRYPTO'94* (1994).
23. De Santis, A., Di Crescenzo, G., Ostrovsky, R., Persiano, G. & Sahai, A. *Robust Non-interactive Zero Knowledge* in *CRYPTO 2001* (2001).
24. ElGamal, T. *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms* in *CRYPTO'84* (1984).
25. Esgin, M. F., Nguyen, N. K. & Seiler, G. *Practical Exact Proofs from Lattices: New Techniques to Exploit Fully-Splitting Rings* in *ASIACRYPT 2020, Part II* (2020).
26. Faonio, A., Fiore, D., Herranz, J. & Ràfols, C. *Structure-Preserving and Re-randomizable RCCA-Secure Public Key Encryption and Its Applications* in *ASIACRYPT 2019, Part III* (2019).
27. Feige, U., Lapidot, D. & Shamir, A. *Multiple Non-Interactive Zero Knowledge Proofs Based on a Single Random String (Extended Abstract)* in *31st FOCS* (1990).
28. Fiat, A. & Shamir, A. *How to Prove Yourself: Practical Solutions to Identification and Signature Problems* in *CRYPTO'86* (1987).
29. Fuchsbauer, G. *Automorphic Signatures and Applications* PhD thesis (ENS Paris and Université Paris 7, 2011).
30. Fuchsbauer, G. *Commuting Signatures and Verifiable Encryption* in *EUROCRYPT 2011* (2011).
31. Gentry, C., Peikert, C. & Vaikuntanathan, V. *Trapdoors for hard lattices and new cryptographic constructions* in *40th ACM STOC* (2008).
32. Gentry, C., Sahai, A. & Waters, B. *Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based* in *CRYPTO 2013, Part I* (2013).
33. Groth, J. *Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures* in *ASIACRYPT 2006* (2006).
34. Groth, J. *Optimal Structure-Preserving Signatures (Invited Talk)* in *ProvSec 2011* (2011).

35. Groth, J. & Sahai, A. *Efficient Non-interactive Proof Systems for Bilinear Groups* in *EUROCRYPT 2008* (2008).
36. Hofheinz, D. & Kiltz, E. *Programmable Hash Functions and Their Applications* in *CRYPTO 2008* (2008).
37. Kilian, J. & Petrank, E. *Identity Escrow* in *CRYPTO'98* (1998).
38. Libert, B., Nguyen, K., Passelègue, A. & Titiu, R. *Simulation-Sound Arguments for LWE and Applications to KDM-CCA2 Security* in *ASIACRYPT 2020, Part I* (2020).
39. Libert, B., Peters, T. & Qian, C. *Structure-Preserving Chosen-Ciphertext Security with Shorter Verifiable Ciphertexts* in *PKC 2017, Part I* (2017).
40. Lyubashevsky, V. *Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures* in *ASIACRYPT 2009* (2009).
41. Lyubashevsky, V. *Lattice Signatures without Trapdoors* in *EUROCRYPT 2012* (2012).
42. Lyubashevsky, V., Nguyen, N. K. & Seiler, G. *Practical Lattice-Based Zero-Knowledge Proofs for Integer Relations* in *ACM CCS 2020* (2020).
43. Lyubashevsky, V., Nguyen, N. K. & Seiler, G. *Shorter Lattice-Based Zero-Knowledge Proofs via One-Time Commitments* in *PKC 2021, Part I* (2021).
44. Lyubashevsky, V., Nguyen, N. K. & Seiler, G. *SMILE: Set Membership from Ideal Lattices with Applications to Ring Signatures and Confidential Transactions* in *CRYPTO 2021, Part II* (2021).
45. Micciancio, D. & Peikert, C. *Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller* in *EUROCRYPT 2012* (2012).
46. Peikert, C. & Shiehian, S. *Noninteractive Zero Knowledge for NP from (Plain) Learning with Errors* in *CRYPTO 2019, Part I* (2019).
47. Peikert, C., Vaikuntanathan, V. & Waters, B. *A Framework for Efficient and Composable Oblivious Transfer* in *CRYPTO 2008* (2008).
48. Regev, O. *On lattices, learning with errors, random linear codes, and cryptography* in *37th ACM STOC* (2005).
49. Rückert, M. *Strongly Unforgeable Signatures and Hierarchical Identity-Based Signatures from Lattices without Random Oracles* in *The Third International Workshop on Post-Quantum Cryptography, PQCRYPTO 2010* (2010).
50. Rückert, M. & Schröder, D. *Security of Verifiably Encrypted Signatures and a Construction without Random Oracles* in *PAIRING 2009* (2009).
51. Sahai, A. *Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security* in *40th FOCS* (1999).
52. Schnorr, C.-P. *Efficient Identification and Signatures for Smart Cards* in *CRYPTO'89* (1990).
53. Zhang, T., Wu, H. & Chow, S. S. M. *Structure-Preserving Certificateless Encryption and Its Application* in *CT-RSA 2019* (2019).

A Deferred instantiations of Lattice-Based Structure-Preserving Signatures

A.1 SIS-based instantiation

In this section, we show that the SIS-based signature scheme put forward by Boyen [14] is a SPS scheme. To this end, we briefly recall Boyen's construction. Parts of the description below are taken verbatim from the work of Boyen. For the definition of algorithms $\mathsf{TrapGen}$ and $\mathsf{SamplePre}$ used in the construction, see Section 4.

$\mathsf{KeyGen}(1^\lambda)$**:** Given unary encoded security parameter λ as input, proceed as follows:
1. Execute the $\mathsf{TrapGen}$ algorithm to obtain a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a basis $\mathbf{T_A} \in \Lambda^\top(\mathbf{A})$ such that $\|\bar{\mathbf{T}}_\mathbf{A}\| \leq L$.
2. Sample $(\mathbf{C}_0, \ldots, \mathbf{C}_\ell) \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m} \times \cdots \times \mathbb{Z}_q^{n\times m}$.
3. Output as verification key $\mathsf{vk} := (\mathbf{A}, \mathbf{C}_0, \ldots, \mathbf{C}_\ell)$ and as signing key $\mathsf{sk} := \mathbf{T_A}$.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{msg})$**:** Given a signing key $\mathsf{sk} = \mathbf{T_A}$ and a message $\mathsf{msg} \in \{0,1\}^\ell$ as input proceed as follows:
1. Compute $\mathbf{C}_{\mathsf{msg}} := \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i$.
2. Set $\mathbf{F}_{\mathsf{msg}} := [\mathbf{A} \mid \mathbf{C}_{\mathsf{msg}}] \in \mathbb{Z}_q^{n\times 2m}$.
3. Execute the algorithm $\mathsf{SamplePre}$ on $\mathbf{F}_{\mathsf{msg}}$, $\mathbf{T_A}$ and $\sigma \geq 2L\omega(\sqrt{\log m})$ to obtain a short non-zero random point $\mathbf{d} \in \Lambda^\perp(\mathbf{F}_{\mathsf{msg}})$.
4. Output the signature $\mathsf{sig} := (\mathsf{core} = \mathbf{d}, \mathsf{tag} = \emptyset)$.

$\mathsf{Ver}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig})$**:** Given a verification key $\mathsf{vk} = (\mathbf{A}, \mathbf{C}_0, \ldots, \mathbf{C}_\ell)$, a message $\mathsf{msg} \in \{0,1\}^\ell$ and signature $\mathsf{sig} = (\mathsf{core}, \mathsf{tag})$ where $\mathsf{core} \in \mathbb{Z}_q^{2m}$ as input, output 1 if
1. $0 < \|\mathsf{core}\| \leq \sqrt{2m} \cdot \sigma$ and
2. $[\mathbf{A} \mid \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i]\mathsf{core} = \mathbf{0} \mod q$.

Otherwise, output 0.

Lemma A.1. *The SIS-based signature scheme of Boyen [14] is a SPS scheme.*

Proof. This signature does not use a tag (formally we set the tag to always be the empty string).

According to definition Definition 4.1, what remains to show is that the signature verification can be expressed as $f(\mathsf{core}) \in S$ for some function $f : \mathbb{Z}_q^{2m} \to \mathbb{Z}_q^{d'}$ and some set $S \subseteq \mathbb{Z}_q^{d'}$ which is structure-preserving. Both the function f and the set S might depend on the message being signed, the verification key and the public parameters of the scheme. We show that the signature verification can be expressed as three checks of the type $f_i(\mathsf{core}) \in S_i$ ($i \in \{1, 2, 3\}$). These check can then be combined to a single check by setting $f(\mathsf{core}) := (f_1(\mathsf{core}), f_2(\mathsf{core}), f_3(\mathsf{core}))$ and $S := S_1 \times S_2 \times S_3$. The set S is structure-preserving when S_1, S_2, and S_3 are structure-preserving by Example 3.8.

Let us first focus on the check $0 < \|\mathsf{core}\|$. Equivalently, we need to verify that core is a non-zero vector. For this, we can set $n'_1 := 1$ and

$$f_1(\mathsf{core}) := \begin{cases} 1, & \text{if } \mathsf{core} = \mathbf{0} \\ 0, & \text{otherwise.} \end{cases} \quad \text{and} \quad S_1 := \{0\}.$$

By Remark 3.3 and Lemma 3.5, we know that S_1 is structure-preserving with a noise growth 0. Let $(s_1, \ldots, s_k) \in \{0, 1\}^k$, for $k = 2m(\lfloor \log q \rfloor + 1)$, be the binary representation of core. Then f_1 can be expressed as $\bigwedge_{i=1}^{k} \neg s_i$ which can be computed by a Boolean circuit of depth $\lfloor \log k \rfloor + 2$.

Secondly, we need to express the check $\|\mathsf{core}\| \le \sqrt{2m} \cdot \sigma$, i.e., that core is a small vector. For this, we can set $n'_2 := 2m$ and

$$f_2(\mathsf{core}) := \mathsf{core}, \quad \text{and} \quad S_2 := \{\mathbf{y} \in \mathbb{Z}_q^{2m} \mid \|\mathbf{y}\| \le \sqrt{2m} \cdot \sigma\} = B_{\sqrt{2m} \cdot \sigma}(\{0\}).$$

By triangular inequality, we have that $S_2 - S_2 \in B_{2\sqrt{2m} \cdot \sigma}(\{0\})$. By Remark 3.7, we can conclude that S_2 is structure-preserving with noise growth $16m\sigma + 1$.

For the final check, we can set $n'_3 := n$ and

$$f_3(\mathsf{core}) := \left[\mathbf{A} \middle| \mathbf{C}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{C}_i \right] \mathsf{core} \quad \text{and} \quad S_3 := \{\mathbf{0}\} \subset \mathbb{Z}_q^n.$$

Note that the function f_3 is defined by the message and the verification key. Moreover, S_3 is a singleton set and hence by Remark 3.3 and Lemma 3.5, we know that it is structure-preserving with noise growth 0. □

Remark A.2. One may wonder whether we could not express the non-zero check as a negation of a zero check (i.e., $f_1(\mathsf{core}) = \mathsf{core}$ and $S_1 = \{0\}$). This would avoid the need of Boolean circuits as f_1, and hence f, would be linear function. Unfortunately, this does not work as structure-preserving sets (and also languages we can prove with our NIZK) are not closed under negations.

The original security proof of Boyen showed only that this scheme is secure when the number of signing queries in the UF-CMA security game is a priori bounded. Namely, their reduction had a security loss of $\mathcal{O}(q)$ (so the modulus q has to be polynomial) and they restricted the adversary makes to make at most $q/2$ signing queries. We give an improved security proof that is tighter and has no restriction on the number of signing queries.

Theorem A.3. *The SIS-based signature scheme of Boyen [14] is* SPS-EUF-CMA*-secure under the* $\mathsf{SIS}_{m,n,q,\beta}$ *problem where β grows polynomial in the security parameter.*

Proof. The reduction gets as input a uniformly random matrix $\mathbf{A}_0 \in \mathbb{Z}_q^{n \times m}$ and is supposed to output a short vector $\mathbf{e}_0 \neq \mathbf{0}$ with $\|\mathbf{e}_0\| \le \beta$ and $\mathbf{A}_0\mathbf{e}_0 = \mathbf{0}$. Let Q be the number of signing queries of the adversary. The reduction proceeds as follows:

1. Sample a $n \times m$ matrix with a short basis: $(\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, n, m)$.
2. Sample short $m \times m$ matrices $\mathbf{R}_0, \ldots, \mathbf{R}_\ell \leftarrow \{-1, 0, 1\}^{m \times m}$.

3. Sample h_i as results of random walks of length L. In more detail, sample for $i \in \{1, \ldots, \ell\}$ and $j \in \{1, \ldots, L\}$ for $L \in \mathcal{O}(Q^2)$ $h_{i,j} \leftarrow_{\mathrm{R}} \{-1, 0, 1\}$ and set $h_i := \sum_{j=1}^{L} h_{i,j}$.[12]
4. Set $\mathbf{C}_i := \mathbf{A}_0\mathbf{R}_i + h_i\mathbf{B}_0$ for all $i \in \{0, \ldots, \ell\}$.
5. Give the verification key $\mathsf{vk} := (\mathbf{A}_0, (\mathbf{C}_i)_{0 \leq i \leq \ell})$ to the adversary.

The reduction answers each of the adversaries signing queries for a message msg as follows:

1. Compute $h_{\mathsf{msg}} := h_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i h_i$.
2. Abort, if $h_{\mathsf{msg}} = 0$.
3. Define $\mathbf{F}_{\mathsf{msg}} := (\mathbf{A}_0|\mathbf{A}_0\mathbf{R}_{\mathsf{msg}} + h_{\mathsf{msg}}\mathbf{B}_0$ with $\mathbf{R}_{\mathsf{msg}} := \mathbf{R}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i \mathbf{R}_i$.
4. Compute a short basis $\mathbf{T}_{\mathbf{F}_{\mathsf{msg}}}$ for $\mathbf{F}_{\mathsf{msg}}$ using the short basis $\mathbf{T}_{\mathbf{B}_0}$ for $\mathbf{B}_0$ via Fact 3. This basis will have $\|\widetilde{\mathbf{T}_{\mathbf{F}_{\mathsf{msg}}}}\| \leq \widetilde{\mathbf{T}_{\mathbf{B}_0}}\|(\|\mathbf{R}\| + 1) \leq 2L$.
5. Sample a short vector $\mathbf{d} \in \Lambda^{\perp}(\mathbf{F}_{\mathsf{msg}})$ using the $\mathsf{SamplePre}$ algorithm from Fact 2 and $\mathbf{T}_{\mathbf{F}_{\mathsf{msg}}}$ with $\sigma \geq 2L\omega(\sqrt{\log m})$.
6. Give $\mathbf{d}$ as signature for msg to the adversary.

When the adversary outputs a forgery $(\mathsf{msg}^\star, \mathbf{d}^\star)$, the reduction solves the SIS instance as follows:

1. Compute $h_{\mathsf{msg}^\star} := h_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i^\star h_i$ and $\mathbf{R}_{\mathsf{msg}^\star} := \mathbf{R}_0 + \sum_{i=1}^{\ell} \mathsf{msg}_i^\star \mathbf{R}_i$.
2. Abort, if $h_{\mathsf{msg}^\star} \neq 0$.
3. Define $((\mathbf{d}_1^\star)^\top | (\mathbf{d}_2^\star)^\top) := (\mathbf{d}^\star)^\top$ with $\mathbf{d}_1^\star, \mathbf{d}_2^\star \in \mathbb{Z}_q^m$.
4. Output $\mathbf{e}_0 := \mathbf{d}_1^\star + \mathbf{R}_{\mathsf{msg}^\star}\mathbf{d}_2^\star$ as solution to the SIS instance.

First, we verify that the reduction correctly simulates the game. Therefore we need that the matrices $\mathbf{C}_i := \mathbf{A}_0\mathbf{R}_i + h_i\mathbf{B}_0$ look uniformly random to the adversary. The matrix $\mathbf{A}_0$ is uniformly random and thus, by the left over hash lemma, $\mathbf{A}_0\mathbf{R}_i$ is statistically close to uniformly random because $\mathbf{R}_i$ has at least $nm \log q + \lambda$ bits of min-entropy. Thus also the coefficients h_i are hidden from the adversary.

Next, we verify that the reduction solves the SIS instance when the adversary successfully forges a signature and the reduction does not abort. In this case we have

$$\mathbf{A}_0\mathbf{e}_0 = \mathbf{A}_0\mathbf{d}_1^\star + \mathbf{A}_0\mathbf{R}_{\mathsf{msg}^\star}\mathbf{d}_2^\star = (\mathbf{A}_0|\mathbf{A}_0\mathbf{R}_{\mathsf{msg}^\star})\mathbf{d}^\star = \mathbf{0},$$

where the last inequality follows from the third signature check. From the second check we know that $\|\mathbf{d}^\star\| \leq \sqrt{2m} \cdot \sigma + 16m\sigma + 1$ and from the first check we know that $\mathbf{d}^\star \neq \mathbf{0}$. Then with high probability also $\mathbf{e}_0$ is a short and non-zero vector. Details for this step can be found in [14, Lemma 26].

Finally, we need to analyze the probability of an abort. This argument follows [36], and we only give a brief summary here. For any two messages $\mathsf{msg}, \mathsf{msg}^\star$ we have

$$\Pr[h_{\mathsf{msg}} \neq 0 \mid h_{\mathsf{msg}^\star} = 0] \geq 1 - 1/\Theta(Q)$$

because h_{msg} differs from $h_{\mathsf{msg}^\star}$ by a random walk of length at least Q^2 and random walk with n steps is back at its origin with probability $1/\Theta(\sqrt{n})$. Let $\mathsf{msg}_1, \ldots, \mathsf{msg}_Q$ be the messages the adversary queried a signature for. By the union bound we get

$$\Pr[h_{\mathsf{msg}_1}, \ldots, h_{\mathsf{msg}_Q} \neq 0 \mid h_{\mathsf{msg}^\star} = 0] \geq \Theta(1).$$

Furthermore, since $h_{\mathsf{msg}^\star}$ is a random walk of length at most $Q^2\ell$ we have

$$\Pr[h_{\mathsf{msg}^\star} = 0] \geq 1/\Theta(Q\sqrt{\ell})$$

and thus

$$\Pr[\text{no abort}] = \Pr[h_{\mathsf{msg}_1}, \ldots, h_{\mathsf{msg}_Q} \neq 0 \wedge h_{\mathsf{msg}^\star} = 0] \geq 1/\Theta(Q\sqrt{\ell}).$$

□

[12] This is the part where our proof differs from Boyen's original proof. There the coefficients h_i are chosen uniformly random over $\mathbb{Z}_q$.

A.2 ISIS-based instantiation

We presented a new signature scheme that combines the techniques of Boyen's and Rückert's signature scheme in Section 4.1, where we also proved that it is a SPS scheme. What remains to prove is that it satisfies $\mathsf{SPS\text{-}sEUF\text{-}CMA}$-security.

Theorem A.4. *The signature scheme presented in Section 4.1 is* $\mathsf{SPS\text{-}sEUF\text{-}CMA}$*-secure under the* $\mathsf{SIS}_{m,n,q,\beta}$ *where* β *grows polynomial in the security parameter.*

Proof. The reduction starts by guessing a bit $b \leftarrow_{\mathrm{R}} \{0,1\}$. $b = 0$ indicates that the reduction hopes that the adversary outputs a forgery $(\mathsf{msg}^\star, (r^\star, \mathbf{d}^\star))$ where $(\mathsf{msg}')^\star := \mathsf{ch}(\mathsf{msg}^\star, r^\star)$ is fresh, i.e. does not match with one of the signatures outputted by the signing oracle. $b = 1$ indicates that the reduction hopes for the opposite event.

In this case $b = 0$, the reduction works very similar to the one for the SIS-based signature, but reduces to the ISIS problem instead (by Remark 2.6 we can reduce the ISIS problem to the SIS problem in the end). The reduction gets as input a uniformly random matrix $\mathbf{A}_0 \in \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{y} \in \mathbb{Z}_q^n$ and is supposed to output a short vector with $\|\mathbf{e}_0\| \leq \beta$ and $\mathbf{A}_0\mathbf{e}_0 = \mathbf{y}$.

In the case $b = 1$, the reduction reduces to the SIS problem. Here the reduction gets as input a uniformly random matrix $\mathbf{A}_0 \in \mathbb{Z}_q^{n \times m}$ and is supposed to output a short vector $\mathbf{e}_0 \neq \mathbf{0}$ with $\|\mathbf{e}_0\| \leq \beta$ and $\mathbf{A}_0\mathbf{e}_0 = \mathbf{0}$. Let Q be the number of signing queries of the adversary. In this case the reduction also guesses an index $i^\star \in \{1, \ldots, Q\}$ and hopes that the adversary uses the message and randomness of the $i^\star$-th signing query for the forgery. The reduction proceeds as follows:

1. Sample a $n \times m$ matrix with a short basis: $(\mathbf{B}_0, \mathbf{T}_{\mathbf{B}_0}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, n, m)$.
2. Sample short $m \times m$ matrices $\mathbf{R}_0, \ldots, \mathbf{R}_\ell \leftarrow \{-1, 0, 1\}^{m \times m}$.
3. Sample h_i as results of random walks of length L. In more detail, sample for $i \in \{1, \ldots, \ell\}$ and $j \in \{1, \ldots, L\}$ for $L \in \mathcal{O}(Q^2)$ $h_{i,j} \leftarrow_{\mathrm{R}} \{-1, 0, 1\}$ and set $h_i := \sum_{j=1}^{L} h_{i,j}$.[13]
4. Set $\mathbf{C}_i := \mathbf{A}_0\mathbf{R}_i + h_i\mathbf{B}_0$ for all $i \in \{0, \ldots, \ell\}$.
5. If $b = 1$, compute $\mathbf{y}$ as follows:
   (a) Sample $\widehat{\mathsf{msg}} \leftarrow_{\mathrm{R}} \mathcal{M}, \widehat{r} \leftarrow_{\mathrm{R}} R$.
   (b) Set $\widehat{\mathsf{msg}}' := \mathsf{ch}(\widehat{\mathsf{msg}}, \widehat{r})$.
   (c) Compute $\mathbf{F}_{\widehat{\mathsf{msg}}'} := \mathbf{C}_0 + \sum_{i=1}^{\ell} \widehat{\mathsf{msg}}'\mathbf{C}_i$.
   (d) Sample $\widehat{\mathbf{d}} \leftarrow_{\mathrm{R}} \mathcal{D}_\sigma^{2m}$, where $\mathcal{D}_\sigma^{2m}$ is the distribution of $2m$-dimensional vectors where each entry is sampled according to a discrete Gaussian distribution.
   (e) Set $\mathbf{y} := \mathbf{F}_{\widehat{\mathsf{msg}}'}\widehat{\mathbf{d}}$.
6. Give the verification key $\mathsf{vk} := (\mathbf{A}_0, (\mathbf{C}_i)_{0 \leq i \leq \ell}, \mathbf{y})$ to the adversary.

The reduction answers each of the adversaries signing queries, except the $i^\star$-th signing query if $b = 1$, for a message msg as follows:

1. Sample $r \in \{0,1\}^{\ell/2}$ and set $\mathsf{msg}' := \mathsf{msg}||r$.
2. Compute $h_{\mathsf{msg}'} := h_0 + \sum_{i=1}^{\ell} \mathsf{msg}'_i h_i$.
3. Abort, if $h_{\mathsf{msg}'} = 0$.
4. Define $\mathbf{F}_{\mathsf{msg}'} := (\mathbf{A}_0 | \mathbf{A}_0\mathbf{R}_{\mathsf{msg}'} + h_{\mathsf{msg}'}\mathbf{B}_0$ with $\mathbf{R}_{\mathsf{msg}'} := \mathbf{R}_0 + \sum_{i=1}^{\ell} \mathsf{msg}'_i\mathbf{R}_i$.
5. Compute a short basis $\mathbf{T}_{\mathbf{F}_{\mathsf{msg}'}}$ for $\mathbf{F}_{\mathsf{msg}'}$ using the short basis $\mathbf{T}_{\mathbf{B}_0}$ for $\mathbf{B}_0$ via Fact 3. This basis will have $\|\widetilde{\mathbf{T}_{\mathbf{F}_{\mathsf{msg}'}}}\| \leq \widetilde{\mathbf{T}_{\mathbf{B}_0}}\|(\|\mathbf{R}\| + 1) \leq 2L$.
6. Sample a short vector $\mathbf{d}$ with $\mathbf{F}_{\mathsf{msg}'}\mathbf{d} = \mathbf{y}$ using the $\mathsf{SamplePre}$ algorithm from Fact 2 and $\mathbf{T}_{\mathbf{F}_{\mathsf{msg}'}}$ with $\sigma \geq 2L\omega(\sqrt{\log m})$.
7. Give $(\mathbf{d}, r)$ as signature for msg to the adversary.

If $b = 1$, the $i^\star$-th signing query for a message msg is answered as follows:

[13] This is the part where our proof differs from Boyen's original proof. There the coefficients h_i are chosen uniformly random over $\mathbb{Z}_q$.

1. Compute $r := \mathsf{TrapColl}(\tau, \widehat{\mathsf{msg}}, \widehat{r}, \mathsf{msg})$.
2. Give $(\widehat{\mathbf{d}}, r)$ as signature for msg to the adversary.

If $b = 0$, when the adversary outputs a forgery $(\mathsf{msg}^\star, (\mathbf{d}^\star, r^\star))$, the reduction solves the ISIS instance as follows:

1. Set $(\mathsf{msg}')^\star := \mathsf{ch}(\mathsf{msg}^\star, r^\star)$.
2. Compute $h_{(\mathsf{msg}')^\star} := h_0 + \sum_{i=1}^{\ell} (\mathsf{msg}')_i^\star h_i$ and $\mathbf{R}_{(\mathsf{msg}')^\star} := \mathbf{R}_0 + \sum_{i=1}^{\ell} (\mathsf{msg}')_i^\star \mathbf{R}_i$.
3. Abort, if $h_{(\mathsf{msg}')^\star} \neq 0$.
4. Define $((\mathbf{d}_1^\star)^\top | (\mathbf{d}_2^\star)^\top) := (\mathbf{d}^\star)^\top$ with $\mathbf{d}_1^\star, \mathbf{d}_2^\star \in \mathbb{Z}_q^m$.
5. Output $\mathbf{e}_0 := \mathbf{d}_1^\star + \mathbf{R}_{(\mathsf{msg}')^\star} \mathbf{d}_2^\star$ as solution to the ISIS instance.

If $b = 1$, when the adversary outputs a forgery $(\mathsf{msg}^\star, (\mathbf{d}^\star, r^\star))$, the reduction solves the ISIS instance as follows:

1. Set $(\mathsf{msg}')^\star := \mathsf{ch}(\mathsf{msg}^\star, r^\star)$.
2. Abort if $(\mathsf{msg}')^\star \neq \widehat{\mathsf{msg}}'$.
3. Compute $h_{(\mathsf{msg}')^\star} := h_0 + \sum_{i=1}^{\ell} (\mathsf{msg}')_i^\star h_i$ and $\mathbf{R}_{(\mathsf{msg}')^\star} := \mathbf{R}_0 + \sum_{i=1}^{\ell} (\mathsf{msg}')_i^\star \mathbf{R}_i$.
4. Abort, if $h_{(\mathsf{msg}')^\star} \neq 0$.
5. Define $\mathbf{d}' := \mathbf{d}^\star - \widehat{\mathbf{d}}$.
6. Define $((\mathbf{d}_1')^\top | (\mathbf{d}_2')^\top) := (\mathbf{d}')^\top$ with $\mathbf{d}_1', \mathbf{d}_2' \in \mathbb{Z}_q^m$.
7. Output $\mathbf{e}_0 := \mathbf{d}_1' + \mathbf{R}_{(\mathsf{msg}')^\star} \mathbf{d}_2'$ as solution to the SIS instance.

First, we verify that the reduction correctly simulates the game. Therefore we need that the matrices $\mathbf{C}_i := \mathbf{A}_0 \mathbf{R}_i + h_i \mathbf{B}_0$ look uniformly random to the adversary. The matrix $\mathbf{A}_0$ is uniformly random and thus, by the left over hash lemma, $\mathbf{A}_0 \mathbf{R}_i$ is statistically close to uniformly random because $\mathbf{R}_i$ has at least $nm \log q + \lambda$ bits of min-entropy. Thus also the coefficients h_i are hidden from the adversary. By a similar argument, we can also argue that in the $b = 1$ case, the vector $\mathbf{y}$ is statistically close to uniformly random using the entropy of $\widehat{\mathbf{d}}$.

Next, we verify that the reduction solves in the $b = 0$ case the ISIS instance when the adversary successfully forges a signature and the reduction does not abort. In this case we have

$$\mathbf{A}_0 \mathbf{e}_0 = \mathbf{A}_0 \mathbf{d}_1^\star + \mathbf{A}_0 \mathbf{R}_{\mathsf{msg}^\star} \mathbf{d}_2^\star = (\mathbf{A}_0 | \mathbf{A}_0 \mathbf{R}_{\mathsf{msg}^\star}) \mathbf{d}^\star = \mathbf{y},$$

where the last inequality follows from the third signature check. From the second check we know that $\|\mathbf{d}^\star\| \leq \sqrt{2m} \cdot \sigma + 16m\sigma + 1$. Then $\|\mathbf{e}_0\| \leq 2\|\mathbf{d}^\star\| \leq 2\sqrt{2m} \cdot \sigma + 32m\sigma + 2$ and thus $\mathbf{e}_0$ is a solution to the ISIS problem.

Similarly, the reduction solves in the $b = 1$ case the SIS instance when the adversary successfully forges a signature and the reduction does not abort. In this case we have

$$\mathbf{F}_{(\mathsf{msg}')^\star} \mathbf{d}^\star = \mathbf{y} = \mathbf{F}_{(\mathsf{msg}')^\star} \widehat{\mathbf{d}}$$

and thus

$$\mathbf{F}_{(\mathsf{msg}')^\star} (\mathbf{d}^\star - \widehat{\mathbf{d}}) = \mathbf{F}_{(\mathsf{msg}')^\star} \mathbf{d}' = \mathbf{0}$$

which we can use to argue that

$$\mathbf{A}_0 \mathbf{e}_0 = \mathbf{A}_0 \mathbf{d}_1' + \mathbf{A}_0 \mathbf{R}_{(\mathsf{msg}')^\star} \mathbf{d}_2' = (\mathbf{A}_0 | \mathbf{A}_0 \mathbf{R}_{(\mathsf{msg}')^\star}) \mathbf{d}' = \mathbf{0},$$

From the second signature check we know that $\|\mathbf{d}^\star\| \leq \sqrt{2m} \cdot \sigma + 16m\sigma + 1$ and with high probability $\|\mathbf{d}'\| \leq \sqrt{2m} \cdot \sigma$ and thus $\|\mathbf{d}'\| \leq 2\sqrt{2m} \cdot \sigma + 16m\sigma + 1$. If the forgery is not trivial, we have $\mathbf{d}' \neq \mathbf{0}$. Then with high probability also $\mathbf{e}_0$ is a short and non-zero vector. Details for this step can be found in [14, Lemma 26].

Finally, we need to analyze the probability of an abort. Assume that the reduction guesses the bit b and the index $i^\star$ correctly. This happens with probability at least $1/2Q$. Also assume that no messages queried by the adversary to the signing oracle or used as forgery produce a collision with the chameleon hash function. Then we can bound the remaining abort probability as follows. The argument follows [36], and we only give a brief summary here. For any two hashed messages $\mathsf{msg}', (\mathsf{msg}')^\star$ we have

$$\Pr[h_{\mathsf{msg}'} \neq 0 \mid h_{(\mathsf{msg}')^\star} = 0] \geq 1 - 1/\Theta(Q)$$

because $h_{\mathsf{msg}'}$ differs from $h_{(\mathsf{msg}')^\star}$ by a random walk of length at least Q^2 and random walk with n steps is back at its origin with probability $1/\Theta(\sqrt{n})$. Let $\mathsf{msg}'_1, \ldots, \mathsf{msg}'_Q$ be the messages the adversary queried a signature for with appended randomness. By the union bound we get

$$\Pr[h_{\mathsf{msg}'_1}, \ldots, h_{\mathsf{msg}'_Q} \neq 0 \mid h_{(\mathsf{msg}')^\star} = 0] \geq \Theta(1).$$

Furthermore, since $h_{(\mathsf{msg}')^\star}$ is a random walk of length at most $Q^2\ell$ we have

$$\Pr[h_{(\mathsf{msg}')^\star} = 0] \geq 1/\Theta(Q\sqrt{\ell})$$

and thus

$$\Pr[\text{no abort}] = \Pr[h_{\mathsf{msg}'_1}, \ldots, h_{\mathsf{msg}'_Q} \neq 0 \wedge h_{(\mathsf{msg}')^\star} = 0] \geq 1/\Theta(Q\sqrt{\ell}).$$

□

A.3 Rückert's scheme

Rückert [49] describes a signature based on Bonsai trees [19] that is also an SPS scheme (Definition 4.1) and satisfies strong existential unforgeability. We begin by recalling the construction. The construction relies on the following facts about lattice trapdoors.

Fact 4 ([49, Proposition 2.3], [19]) *Let $\delta > 0$ be any fixed real constant and let $q \geq 3$ be odd. There is a polynomial time algorithm $\mathsf{ExtLattice}(\mathbf{A}_1, m_2)$ that, given uniformly random $\mathbf{A}_1 \in \mathbb{Z}_q^{n\times m_1}$ for any $m_1 \geq (1+\delta)n\log(q)$ and $\mathsf{poly}(n)$-bounded $m_2 \geq (4+2\delta)n\log(q)$, outputs $(\mathbf{A}_2 \in Z_q^{n\times m_2}, \mathbf{S} \in \mathbb{Z}^{m\times m})$, where $m = m_1 + m_2$, such that $\mathbf{A} = (\mathbf{A}_1|\mathbf{A}_2)$ is within negligible statistical distance of uniform, $\mathbf{S}$ is a basis of $\Lambda_q^\perp(\mathbf{A}_1|\mathbf{A}_2)$, $\|S\| \leq L = Cn\log(q)$ with overwhelming probability, and for the Gram-Schmidt orthogonalization $\tilde{\mathbf{S}}$ of $\mathbf{S}$ we have $\|\tilde{\mathbf{S}}\| \leq \tilde{L} = 1 + C\sqrt{(1+\delta)nlog(n)} \leq 1 + C\sqrt{m_1}$ with overwhelming probability.*

Fact 5 ([19, Proposition 2.4]) *There exists a deterministic polynomial time algorithm $\mathsf{ExtBasis}(\mathbf{S}_1, \mathbf{A}_1, \mathbf{A}_2)$ that takes a short basis $\mathbf{S}_1$ of $\Lambda_q^\perp(\mathbf{A}_1)$ and two matrices $\mathbf{A}_1 \in \mathbb{Z}_q^{n\times m_1}$ and $\mathbf{A}_2 \in \mathbb{Z}_q^{n\times m_2}$ with $m_1 \geq 2n\log(q)$. It outputs a short basis $\mathbf{S}$ for $\Lambda_q^\perp(\mathbf{A} := (\mathbf{A}_1|\mathbf{A}_2))$ with $\|\tilde{\mathbf{S}}\| = \|\tilde{\mathbf{S}}_2\|$, where $\tilde{\mathbf{S}}$ and $\tilde{\mathbf{S}}_2$ are the Gram-Schmidt orthogonalization of $\mathbf{S}$ and $\mathbf{S}_2$, respectively.*

The lattice trapdoor can be used to sample efficiently short preimages, as described by the following fact.

Fact 6 *The algorithm $\mathsf{SamplePre}(\mathbf{S}, s, \mathbf{y})$ takes as input a shot basis $\mathbf{S} \in \mathbb{Z}_q^{m\times m}$ of a lattice $\Lambda_q^\perp(\mathbf{A})$, a parameter s and a vector $\mathbf{y} \in \mathbb{Z}_q^n$ and outputs a vector from the set*

$$\mathbf{x} \in \mathbb{Z}_q^m | \|\mathbf{x}\| \leq s\sqrt{m}, \mathbf{x} \neq \mathbf{0}, \mathbf{A}\mathbf{x} = \mathbf{y}$$

according to Gaussian distribution.

The construction uses a chameleon hash function $(\mathsf{GenCH}, \mathsf{TrapColl})$ and is described as follows:

$\mathsf{KeyGen}(1^\lambda)$**:** Given unary encoded security parameter λ as input, proceed as follows:

1. Choose $q, \tilde{L}, m_1, m_2$ as in Fact 4.
2. Set $s := \tilde{L}\omega(\sqrt{\log(n)})$ and $d := s\sqrt{m_1 + (\lambda+1)m_2}$.
3. Sample $\mathbf{A}_1 \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m_1}$.
4. Sample $(\mathbf{A}_2, \mathbf{S}^\star) \leftarrow_{\mathrm{R}} \mathsf{ExtLattice}(\mathbf{A}_1, m_2)$.
5. Set $\mathbf{A}^\star := (\mathbf{A}_1|\mathbf{A}_2)$.
6. Sample a sequence $\langle\mathbf{B}\rangle := \left((\mathbf{B}_i^{(0)}, \mathbf{B}_i^{(1)})\right)_{1\leq i\leq\lambda}$ of uniformly random matrices in $\mathbb{Z}_q^{n\times m_2}$.
7. Sample $\mathbf{y} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$.
8. Sample $(\mathsf{ch}, \tau) \leftarrow_{\mathrm{R}} \mathsf{GenCH}(1^\lambda)$.
9. Output the signing key $\mathsf{sk} := (\mathbf{A}^\star, \langle\mathbf{B}\rangle, \mathbf{y}, \mathbf{S}^\star, \mathsf{ch})$ and the verification key $\mathsf{vk} := (\mathbf{A}^\star, \langle\mathbf{B}\rangle, \mathbf{y}, \mathsf{ch})$.

$\mathsf{Sign}(\mathsf{sk}, \mathsf{msg})$**:** Given a signing key $\mathsf{sk} = (\mathbf{A}^\star, \langle \mathbf{B} \rangle, \mathbf{y}, \mathbf{S}^\star, \mathsf{ch})$ and a message $\mathsf{msg} \in \{0,1\}^*$ as input, proceed as follows:

1. Sample $r \leftarrow \mathcal{R}$.
2. Compute $h := \mathsf{ch}(m, r)$
3. Set $\mathbf{B}_h := (\mathbf{B}_1^{(h_1)} | \cdots | \mathbf{B}_\lambda^{(h_\lambda)})$
4. $\mathbf{S}_h := \mathsf{ExtBasis}(\mathbf{S}^*, \mathbf{A}^*, \mathbf{B}_h)$
5. Sample $\mathbf{d} \leftarrow_{\mathrm{R}} \mathsf{SamplePre}(\mathbf{S}_h, s, \mathbf{y})$.
6. Output the signature $\mathsf{sig} = (\mathsf{core} = \mathbf{d}, \mathsf{tag} = r)$.

$\mathsf{Ver}(\mathsf{vk}, \mathsf{msg}, \mathsf{sig})$**:** Given a verification key $\mathsf{vk} = (\mathbf{A}^\star, \langle \mathbf{B} \rangle, \mathbf{y})$, a message $\mathsf{msg} \in \{0,1\}^*$ and signature $(\mathbf{d}, r)$ as input, proceed as follows:

1. Compute $h := \mathsf{ch}(m, r)$
2. Set $\mathbf{A}_h := (\mathbf{A}^\star | \mathbf{B}_1^{(h_1)} | \cdots | \mathbf{B}_\lambda^{(h_\lambda)})$
3. Output 1 if $\|\mathbf{d}\| \leq s\sqrt{m_1 + (\lambda+1)m_2}$ and $\mathbf{A}_h \mathbf{d} = \mathbf{y}$.
4. Otherwise, output 0.

Lemma A.5. *Rückert's signature scheme is a SPS scheme.*

Proof. For a signature $(\mathbf{d}, r)$, $\mathbf{d}$ will be the core and r will be the tag. Clearly, these tags are publicly samplable.

According to definition Definition 4.1, what remains to show is that the signature verification procedure can be expressed as $f(\mathsf{core}) \in S$ for some function $f : \mathbb{Z}_q^{m_1+(\lambda+1)m_2} \to \mathbb{Z}_q^{d'}$ and some structure-preserving set $S \subseteq \mathbb{Z}_q^{d'}$. We show that the signature verification can be expressed as two checks of the type $f_i(\mathsf{core}) \in S_i$ $(i \in \{1,2\})$. These check can then be combined to a single check by setting $f(\mathsf{core}) := (f_1(\mathsf{core}), f_2(\mathsf{core}))$ and $S := S_1 \times S_2$. The set S is structure-preserving when S_1 and S_2 are structure-preserving by Example 3.8.

First, we need to express the check $\|\mathbf{d}\| \leq s\sqrt{m_1 + (\lambda+1)m_2}$, i.e., that core is a small vector. For this, we can set

$$f_1(\mathsf{core}) := \mathsf{core}, \quad \text{and} \quad \begin{aligned} S_1 :=& \{\mathbf{x} \in \mathbb{Z}_q^{2m} \mid \|\mathbf{x}\| \leq s\sqrt{m_1 + (\lambda+1)m_2}\} \\ =& B_{s\sqrt{m_1+(\lambda+1)m_2}}(\{0\}). \end{aligned}$$

By triangular inequality, we have that $S_1 - S_1 \in B_{2s\sqrt{m_1+(\lambda+1)m_2}}(\{0\})$. By Remark 3.7, we can conclude that S_1 is structure-preserving with noise growth $8s(m_1 + (\lambda+1)m_2) + 1$.

For the other check, we can set

$$\begin{aligned} &f_2(\mathsf{core}) := \mathbf{A}_h \mathsf{core} \quad \text{for } \mathbf{A}_h := (\mathbf{A}^\star | \mathbf{B}_1^{(h_1)} | \cdots | \mathbf{B}_\lambda^{(h_\lambda)}) \text{ and } h := \mathsf{ch}(m, r) \\ &\text{and} \quad S_2 := \{\mathbf{y}\}, \end{aligned}$$

where $\mathbf{x} := \binom{0}{\mathsf{msg}}$. Note that the function f_2 is defined by the message, the signatures tag and the verification key. Moreover, S_2 is a singleton set and hence by Remark 3.3 and Lemma 3.5, we know that it is structure-preserving with noise growth 0. □

Theorem A.6. *Rückert's signature scheme achieves* SPS-sEUF-CMA *security under the* $\mathsf{SIS}_{m,n,q,\beta}$ *problem where* β *grows polynomial in the security parameter.*

Proof. The proof works exactly as for [49, Theorem 4.1]. The only difference is that for SPS-sEUF-CMA security, we allow the forged signature to be larger by a summand of $8s(m_1 + (\lambda+1)m_2) + 1$, due to the noise growth of the "short vector" structure-preserving set. But this only increases β by a polynomial summand compared to the original proof. □

B Instantiations of Lattice-Based Structure-Preserving Encryption

In Section 5.1, we showed that Regev's encryption scheme is a SPE scheme as of Definition 5.1. Here we prove that the same holds for the Dual Regev's and the GSW encryption schemes. The schemes are parametrized by a LWE modulus q, dimension n, number of samples $m \geq n \log q$ and an error distribution $\chi = \mathcal{D}_{\mathbb{Z},\sigma}$.

Dual Regev's Encryption. Gentry et al. [31] defined an LWE-based encryption scheme that is often refer to as *dual* to the one of Regev. Note that in Regev's encryption scheme, public keys have an LWE (i.e., non-uniform) distribution with a unique secret key. Moreover, given a public key, there are many choices of encryption randomness that produce the same ciphertext. At a high level, the dual encryption scheme flips these two properties around. Namely, public keys are uniformly random with many possible secret keys. But, given a public key, the encryption randomness that produce a certain ciphertext is unique.

We now present the dual Regev's encryption scheme with message space $\mathcal{M} = \mathbb{Z}_p$ for p s.t. $\frac{q}{p}$ is sufficiently large. Again, we assume $q = p^k$ and denote $c := \frac{q}{p} = p^{k-1}$. We recall this scheme with $\alpha = 1$.

$\mathsf{KeyGen}(1^\lambda)$**:** Sample $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m}$, $\mathbf{z} \leftarrow_{\mathrm{R}} \{-1,0,1\}^m$. Output the secret key $\mathsf{sk} := \mathbf{z}$ and the public key $\mathsf{pk} = (\mathbf{A}, \mathbf{A}\mathbf{z}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n$.

$\mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$**:** Parse pk as $(\mathbf{A}, \mathbf{u})$. Sample $\mathbf{s} \leftarrow \chi^n$ and $\mathbf{e} \leftarrow \chi^m$ and $e' \leftarrow \chi$. Compute $\mathbf{c}_0 := \mathbf{A}^\top \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m$ and $c_1 := \mathbf{u}^\top \mathbf{s} + c \cdot \mathsf{msg} + e' \in \mathbb{Z}_q$. Then output the ciphertext $\mathsf{ct} := (\mathbf{c}_0, c_1)$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$**:** Parse ct as $(\mathbf{c}_0, c_1)$ and set $\mathbf{z} := \mathsf{sk}$. Compute $d := c_1 - \mathbf{z}^\top \mathbf{c_0}$ and output $x \in \mathbb{Z}_p$, such that $d - c \cdot x \mod q$ is closest to 0.

To encrypt a higher-dimensional message $(\mathsf{msg}_1, \ldots, \mathsf{msg}_\alpha)^\top \in \mathcal{M}^\alpha$, we encrypt each component individually, i.e. generate $\mathsf{ct}_i = \mathsf{Enc}(\mathsf{pk}, \mathsf{msg}_i)$ for $i \in \{1, \ldots, \alpha\}$ and chain the ciphertext together, i.e. $\mathsf{ct}^\top = (\mathsf{ct}_1^\top, \ldots, \mathsf{ct}_\alpha^\top)$.

Lemma B.1. *Dual Regev's encryption scheme is a lattice-based SPE scheme.*

Proof. For a public key $\mathsf{pk} = (\mathbf{A}, \mathbf{u}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n$, and a message $\mathsf{msg} \in \mathbb{Z}_p$, let us set $r := n + m + 1$, and define the matrix $\mathbf{B}$ and the function g follows:

$$\mathbf{B} := \mathbf{I}_\alpha \otimes \left(\begin{array}{c|c|c} \mathbf{A}^\top & \mathbf{I}_m & \mathbf{0} \\ \mathbf{u}^\top & 0 & 1 \end{array}\right) \in \mathbb{Z}_q^{\alpha(m+1)\times \alpha r},$$

$$g_\alpha \begin{pmatrix} \mathsf{msg}_1 \\ \vdots \\ \mathsf{msg}_\alpha \end{pmatrix} := \begin{pmatrix} \mathbf{0} \\ c \cdot \mathsf{msg}_1 \\ \vdots \\ \mathbf{0} \\ c \cdot \mathsf{msg}_\alpha \end{pmatrix} \in \mathbb{Z}_q^{\alpha(m+1)}$$

Let $\mathcal{R} := \mathcal{D}_{\mathbb{Z}^{\alpha r}, \sigma}$ and $R = B_{2\sigma\sqrt{\alpha r}}(\{0\})$. Then, by Lemma 2.1, we have that

$$\Pr_{\mathbf{r} \leftarrow \mathcal{R}}[\mathbf{r} \notin R] = \Pr_{\mathbf{r} \leftarrow \mathcal{D}_{\mathbb{Z}^{\alpha r}, \sigma}}[\|\mathbf{r}\| > 2\sigma\sqrt{\alpha r}] < 2^{\alpha r} e^{-\frac{3\alpha r}{2}} < \frac{1}{2^{\alpha r}}.$$

Hence, with overwhelming probability, $\mathbf{r} \leftarrow \mathcal{R}$ lies in R. Moreover, by Remark 3.7, we know that R is a structure-preserving set with noise growth $\delta_R := 16\sigma\alpha r + 1$.

We can argue as in the proof of Lemma 5.3 that g is invertible and additively homomorphic.

Next, we need to prove that the encryption algorithm is equivalent to sampling $\mathbf{r} \leftarrow \mathcal{R}$ and computing $\mathbf{Br}+g(\mathsf{msg})$. For $\mathsf{msg} = (\mathsf{msg}_1, \ldots, \mathsf{msg}_\alpha)^\top \in \mathbb{Z}_p^\alpha$, and $\mathbf{r}^\top = (\mathbf{s}_1, \mathbf{e}_1, e'_1, \ldots, \mathbf{s}_\alpha, \mathbf{e}_\alpha, e'_\alpha)$, we have

$$\mathbf{Br} + g(\mathsf{msg}) = \left(\mathbf{I}_\alpha \otimes \left(\begin{array}{c|c|c} \mathbf{A}^\top & \mathbf{I}_m & \mathbf{0} \\ \mathbf{u}^\top & 0 & 1 \end{array}\right)\right) \begin{pmatrix} \mathbf{s}_1 \\ \mathbf{e}_1 \\ e'_1 \\ \vdots \\ \mathbf{s}_\alpha \\ \mathbf{e}_\alpha \\ e'_\alpha \end{pmatrix} + \begin{pmatrix} \mathbf{0} \\ c \cdot \mathsf{msg}_1 \\ \vdots \\ \mathbf{0} \\ c \cdot \mathsf{msg}_\alpha \end{pmatrix}$$

$$= \begin{pmatrix} \mathbf{A}^\top \mathbf{s}_1 + \mathbf{e}_1 \\ \mathbf{u}^\top \mathbf{s}_1 + e'_1 + c \cdot \mathsf{msg}_1 \\ \vdots \\ \mathbf{A}^\top \mathbf{s}_\alpha + \mathbf{e}_\alpha \\ \mathbf{u}^\top \mathbf{s}_\alpha + e'_\alpha + c \cdot \mathsf{msg}_\alpha \end{pmatrix} = \begin{pmatrix} \mathbf{c}_{1,0} \\ c_{1,1} \\ \vdots \\ \mathbf{c}_{\alpha,0} \\ c_{\alpha,1} \end{pmatrix} = \mathsf{ct}$$

Finally, we need to prove that the existence of the $\mathsf{NoiseLevel}$ algorithm. Similar to Regev's encryption, let use define $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct})$ as follows: Parse ct as $(\mathsf{ct}_1, \ldots, \mathsf{ct}_\alpha)$ and each ct_i as $(\mathbf{c}_{i,0}, c_{i,1})$ and set $\mathbf{z} := \mathsf{sk}$. Compute $d_i := c_{i,1} - \mathbf{z}^\top \mathbf{c}_{i,0} \in \mathbb{Z}_q$ and $\nu_i := |d_i - c \cdot \mathsf{Dec}(\mathsf{sk}, \mathsf{ct}_i)|$. Output $\max_{1 \le i \le \alpha} \nu_i$.

As for Regev's encryption, we show that this definition of the $\mathsf{NoiseLevel}$ function has the desired properties for $\alpha = 1$. This implies that it also has the desired properties for $\alpha > 1$, because all these properties only talk about upper bounds of the noise level and the noise level of a ciphertext for $\alpha > 1$ is simply the maximum of the noise levels of the ciphertexts for each component of the message. To show boundedness, define $\mathsf{MaxNoiseLevel}(\delta) := (\sqrt{m+1})\delta$. Then for $\|\mathbf{z}\| < \delta$ we have

$$\begin{aligned} &\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct} = (\mathbf{A}^\top \mathbf{s} + \mathbf{e}, \mathbf{z}^\top \mathbf{A}^\top \mathbf{s} + c \cdot \mathsf{msg} + e')) \\ &= |(\mathbf{z}^\top \mathbf{A}^\top \mathbf{s} + e' - \mathbf{z}^\top (\mathbf{A}^\top \mathbf{s} + \mathbf{e})| \\ &= |e' - \mathbf{z}^\top \mathbf{e}| \le \left| \begin{pmatrix} 1 \\ -\mathbf{z} \end{pmatrix}^\top \begin{pmatrix} e' \\ \mathbf{e} \end{pmatrix} \right| \overset{(1)}{\le} \left\| \begin{pmatrix} 1 \\ -\mathbf{z} \end{pmatrix} \right\| \left\| \begin{pmatrix} e' \\ \mathbf{e} \end{pmatrix} \right\| \\ &\overset{(2)}{\le} (\sqrt{m+1})\delta, \end{aligned}$$

where inequality (1) follows from the Cauchy-Schwartz inequality.

The maximal initial noise level is $\nu_{\mathsf{init}} := 2\sigma(m+1)$: An honestly generated ciphertext uses $\binom{e'}{\mathbf{e}} \leftarrow \chi^{m+1}$ and thus has $\|\binom{e'}{\mathbf{e}}\| \le 2\sigma\sqrt{m+1}$ with overwhelming probability by the Gaussian tail bound (Lemma 2.1). Plugging this in the $\mathsf{MaxNoiseLevel}$ function yields the desired bound.

The maximum noise level is $\nu_{\mathsf{max}} := \lceil c/2 \rceil$, because then for a ciphertext $\mathsf{ct} = (\mathbf{c}_0, c_1)$ for msg, the value $d := c_1 - \mathbf{z}^\top \mathbf{c_0}$ deviates at most by $\lceil c/2 \rceil$ from $c\mathsf{msg}$ and so the Dec algorithm will round to msg.

The Symmetry property of $\mathsf{NoiseLevel}$ follows immediately from the definition and the subadditivity property follows immediately from the triangle inequality.

□

GSW Encryption. The third lattice based scheme that we recall here is the FHE scheme put forward by Gentry, Sahai and Waters in 2013 [32]. We refer to this scheme as GSW for short.

Let $L := \lfloor \log q \rfloor + 1$ and $m := (n+1) \cdot L$. We describe the GSW construction using a *gadget matrix* $\mathbf{G}$ [45] defined as $\mathbf{G} := \mathbf{I}_{n+1} \otimes \mathbf{g}$ for $\mathbf{g} = (2^0, 2^1, \ldots, 2^{L-1})$. This means that $\mathbf{G}$ is a $(n+1) \times m$ matrix whose rows consist of shifts of the vector $\mathbf{g}$.

Pick $j \in \{0, \ldots, L-1\}$. This parameter controls the tradeoff between message space size and the maximum tolerated noise level. The base message space is $\mathcal{M} = \{0, \ldots, \lfloor q/2^j \rfloor\}$. We recall this scheme with $\alpha = 1$.

$\mathsf{KeyGen}(1^\lambda)$**:** Sample $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m}$, $\mathbf{e} \leftarrow \chi^m$. Output secret key sk and public key pk defined as

$$\mathsf{sk} := \begin{pmatrix} -\mathbf{s} \\ 1 \end{pmatrix} \in \mathbb{Z}_q^{n+1}, \quad \mathsf{pk} := \begin{pmatrix} \mathbf{A} \\ \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \end{pmatrix} \in \mathbb{Z}_q^{(n+1)\times m}.$$

$\mathsf{Enc}(\mathsf{pk}, \mathsf{msg})$**:** Let $\mathbf{U} := \mathsf{pk}$. Sample $\mathbf{R} \leftarrow_{\mathrm{R}} \{-1, 0, 1\}^{m\times N}$. Then output the ciphertext

$$\mathsf{ct} := \mathbf{U}\mathbf{R} + \mathsf{msg} \cdot \mathbf{G} \in \mathbb{Z}_q^{(n+1)\times m}$$

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$**:** Let $\mathbf{t} := \mathsf{sk}$ and $\mathbf{v}$ be the $(m - j)$-th column of ct. Output $x \in \mathbb{Z}_q$ such that $\mathbf{t}^\top \mathbf{v} - x \cdot 2^j \mod q$ is closest to 0.

To encrypt a higher-dimensional message $(\mathsf{msg}_1, \ldots, \mathsf{msg}_\alpha)^\top \in \mathcal{M}^\alpha$, we encrypt each component individually, i.e. generate $\mathsf{ct}_i = \mathsf{Enc}(\mathsf{pk}, \mathsf{msg}_i)$ for $i \in \{1, \ldots, \alpha\}$ and chain the ciphertext together, i.e. $\mathsf{ct} = (\mathsf{ct}_1, \ldots, \mathsf{ct}_\alpha)$.

Lemma B.2. *GSW encryption scheme is a lattice-based SPE scheme.*

Before we prove the lemma, let us define an auxiliary algorithm $\mathsf{ToVector}$ that takes as input a matrix $\mathbf{M}$ and outputs a column vector $\mathbf{m}$ obtained by "stacking" all columns of $\mathbf{M}$. More precisely, let $\mathbf{M} = (m_{i,j})_{i\in[n_0], j\in[n_1]}$ for some $n_0, n_1 \in \mathbb{N}$. Then

$$\mathsf{ToVector}(\mathbf{M}) = (m_{1,1}, \ldots, m_{n_0,1}, m_{1,2}, \ldots, m_{n_0,2}, \ldots, m_{1,n_1}, \ldots, m_{n_0,n_1})^\top.$$

Since according to the SPE definition the ciphertexts have to be vectors, we will apply $\mathsf{ToVector}(\mathsf{ct})$ to the ciphertexts for this definition.

Proof. For a public key $\mathbf{U} := \mathsf{pk} \in \mathbb{Z}_q^{(n+1)\times m}$, and a message $\mathsf{msg} \in \mathcal{M} = \mathbb{Z}_q$, let define the matrix $\mathbf{B}$ and the function g as follows:

$$\mathbf{B} := \mathbf{I}_\alpha \otimes (\mathbf{I}_m \otimes \mathbf{U}) \in \mathbb{Z}_q^{\alpha(n+1)m\times \alpha mm},$$
$$g(\mathsf{msg}) := \mathsf{ToVector}(\mathsf{msg}^\top \otimes \mathbf{G}) \in \mathbb{Z}_q^{\alpha(n+1)m}.$$

Let $\mathcal{R}$ be the uniform distribution over $R := \{-1, 0, 1\}^{\alpha mm}$.

Clearly, $\mathbf{r} \leftarrow \mathcal{R}$ lies in R with probability 1. We need to show that R is a structure-preserving set. $R = \{-1, 0, 1\}^{\alpha mm} \subseteq \mathbb{Z}_q^{\alpha mm}$ is a $\sqrt{\alpha mm}$-bounded set which, by Remark 3.7, implies that R is structure-preserving with noise growth $\delta_R := 8\alpha mm + 1$.

As a next step, we need to prove that g is invertible and additively homomorphic. It is easy to recover $\mathsf{msg} \in \mathcal{M}^\alpha$ from $g(\mathsf{msg})$, for example by taking every $(n + 1)m$-th entry. Thus g is invertible. Let us fix $\mathsf{msg}_1, \mathsf{msg}_2 \in \mathbb{Z}_q$. Then we have

$$\begin{aligned} g(\mathsf{msg}_1 + \mathsf{msg}_2) &= \cdot\mathsf{ToVector}((\mathsf{msg}_1 + \mathsf{msg}_2)^\top \otimes \mathbf{G}) \\ &= \mathsf{ToVector}(\mathsf{msg}_1^\top \otimes \mathbf{G} + \mathsf{msg}_2^\top \otimes \mathbf{G}) \\ &= \mathsf{ToVector}(\mathsf{msg}_1^\top \otimes \mathbf{G}) + \mathsf{ToVector}(\mathsf{msg}_2^\top \otimes \mathbf{G}) \\ &= g(\mathsf{msg}_1) + g(\mathsf{msg}_2) \end{aligned}$$

proving that g is additively homomorphic.

Next, we need to prove that the encryption algorithm is equivalent to sampling $\mathbf{r} \in R$ and computing $\mathbf{B}\mathbf{r} + g(\mathsf{msg})$. For $\mathbf{r} \leftarrow_{\mathrm{R}} R$, let us write $\mathbf{r}^\top =: (\mathbf{r}_1^\top, \ldots, \mathbf{r}_\alpha^\top)$ such that for each $i \in \{1, \ldots, \alpha\}$ $\mathbf{r}_i \in \mathbb{Z}_q^{(n+1)m}$ and let $\mathbf{R}_i$ be the

$(n+1) \times m$ matrix such that $\mathbf{r}_i = \mathsf{ToVector}(\mathbf{R}_i)$. Then for $\mathsf{msg} \in \mathcal{M}^\alpha$ we have

$$\mathbf{B}\mathbf{r} + g(\mathsf{msg}) = (\mathbf{I}_\alpha \otimes (\mathbf{I}_m \otimes \mathbf{U}))\mathbf{r} + \mathsf{ToVector}(\mathsf{msg}^\top \otimes \mathbf{G})$$

$$= \begin{pmatrix} (\mathbf{I}_m \otimes \mathbf{U})\mathbf{r}_1 + \mathsf{ToVector}(\mathsf{msg}_1 \cdot \mathbf{G}) \\ \vdots \\ (\mathbf{I}_m \otimes \mathbf{U})\mathbf{r}_\alpha + \mathsf{ToVector}(\mathsf{msg}_\alpha \cdot \mathbf{G}) \end{pmatrix} = \begin{pmatrix} \mathsf{ToVector}(\mathbf{U}\mathbf{R}_1 + \mathsf{msg}_1 \cdot \mathbf{G}) \\ \vdots \\ \mathsf{ToVector}(\mathbf{U}\mathbf{R}_\alpha + \mathsf{msg}_\alpha \cdot \mathbf{G}) \end{pmatrix}$$

$$= \begin{pmatrix} \mathsf{ct}_1 \\ \vdots \\ \mathsf{ct}_\alpha \end{pmatrix} = \mathsf{ct}$$

Finally, we need to prove that the existence of the NoiseLevel algorithm. Let us define $\mathsf{NoiseLevel}(\mathsf{sk}, \mathsf{ct})$ as follows: Parse ct as $(\mathsf{ct}_1, \dots, \mathsf{ct}_\alpha)$ and for each $i \in \{1, \dots, \alpha\}$ let $\mathbf{v}_i$ be the $(m-j)$-th column of ct_i (in matrix form) and $\mathbf{t} := \mathsf{sk}$. Then define $\nu_i := |\mathbf{t}^\top \mathbf{v}_i - \mathsf{Dec}(\mathsf{sk}, \mathsf{ct}_i)2^j|$ and output $\max_{1 \leq i \leq \alpha} \nu_i$.

As before, we show that this definition of the NoiseLevel function has the desired properties for $\alpha = 1$. This implies that it also has the desired properties for $\alpha > 1$, because all these properties only talk about upper bounds of the noise level and the noise level of a ciphertext for $\alpha > 1$ is simply the maximum of the noise levels of the ciphertexts for each component of the message. To show boundedness, define $\mathsf{MaxNoiseLevel}(\delta) := 2\sigma\sqrt{m}\delta$. Then, for $\|\mathsf{ToVector}(\mathbf{R})\| < \delta$, we have

$$\mathsf{NoiseLevel}(\mathsf{sk} = \begin{pmatrix} -\mathbf{s} \\ 1 \end{pmatrix}, \mathsf{ct} = \begin{pmatrix} \mathbf{A} \\ \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \end{pmatrix} \mathbf{R} + \mathsf{msg} \cdot \mathbf{G})$$

$$= |\mathbf{e}^\top \mathbf{R}_j| \overset{(1)}{\leq} \|\mathbf{e}\| \|\mathbf{R}_j\| \leq \|\mathbf{e}\|\delta \overset{(2)}{\leq} 2\sigma\sqrt{m}\delta,$$

where inequality (1) follows from the Cauchy-Schwartz inequality and inequality (2) follows from the Gaussian tail bound (Lemma 2.1).

The maximal initial noise level is $\nu_{\mathsf{init}} := 2\sigma m^{3/2}\sqrt{m}$: An honestly generated ciphertext has randomness $\mathbf{R} \in \{-1, 0, 1\}^{m \times N}$ and thus

$$\|\mathsf{ToVector}(\mathbf{R})\| = \sqrt{\|\mathbf{R}_1\|^2 + \cdots + \|\mathbf{R}_m\|^2} \leq \sqrt{m^2 m} = m\sqrt{m}.$$

Plugging this in the MaxNoiseLevel function yields the desired bound.

The maximum noise level is $\nu_{\mathsf{max}} := 2^{j-1}$, because then for a ciphertext ct for msg, the value $\mathbf{t}^\top \mathbf{v}$ (where $\mathbf{v}$ is the $(m-j)$-th column of ct) deviates at most by 2^{j-1} from $2^j\mathsf{msg}$ and so the Dec algorithm will round to msg.

The Symmetry property of NoiseLevel follows immediately from the definition and the subadditivity property follows immediately from the triangle inequality. □

C Formal definition of SPNIZK

Definition C.1 (Structure-Preserving NIZK (SPNIZK) Argument). *Let S be a structure-preserving set with noise growth δ_S and* SPE *be a structure-preserving public key encryption scheme with message space $\mathcal{M}^\alpha$ and randomness distribution $\mathcal{R}_\alpha$, where $\mathbf{r} \leftarrow_{\mathrm{R}} \mathcal{R}_\alpha$ lies with overwhelming probability in a structure-preserving set $R_\alpha \subseteq \mathbb{Z}_q^r$ with noise growth δ_R. A NIZK argument system* $(\mathsf{Gen}_{\mathsf{par}}, \mathsf{Gen}_{\mathcal{L}}, \mathsf{P}, \mathsf{V})$ *is a structure-preserving NIZK (SPNIZK) with respect to S and* SPE *if for any* $(\mathsf{pk}, \cdot) \leftarrow \mathsf{SPE.Setup}(1^\lambda)$*, encryption randomness $\mathbf{r} \leftarrow_{\mathrm{R}} \mathcal{R}$ and $m \in S$,* SPNIZK *supports the following functionality:*

- $\mathsf{ProveMembershipS}_S(\mathsf{crs}, \mathsf{pk}, m, \mathsf{ct}, \mathbf{r})$ *outputs a proof π that* ct *encrypts a message m which belongs to the structure-preserving set S.*
- $\mathsf{VerifyMembershipS}_S(\mathsf{crs}, \mathsf{pk}, \mathsf{ct}, \pi)$ *verifies that* ct *indeed encrypts a message m which belongs to the structure-preserving set S.*

As in Definition 2.10, the SPNIZK *must satisfy completeness, computational soundness, and zero-knowledge:*

1. *Completeness, meaning that for every* $(\mathsf{pk}, \cdot) \leftarrow \mathsf{SPE.Setup}(1^\lambda)$, $\mathbf{r} \leftarrow_{\mathrm{R}} \mathcal{R}_\alpha$ *and* $m \in S$, *we have:* $\mathsf{VerifyMembershipS}_S(\mathsf{crs}, \mathsf{pk}, \mathsf{ct}, \mathsf{P}$ $\mathsf{pk}, m, \mathsf{ct}, \mathbf{r})) \geq 1 - \mathsf{negl}(\lambda)$.
2. *Computational soundness holds only with respect to slightly larger message sets* S' *and randomness space* R', *with* $S' = B_{\delta_S}(S)$ *and* $R' = B_{\delta_R}(R)$. *This means that if* $\mathsf{VerifyMembershipS}_S(\mathsf{pk}, \mathsf{ct}, \pi) = 1$, *it holds with overwhelming probability that* ct *encodes a message in* S', *encrypted with randomness in the set* R'.
3. *Statistical zero-knowledge: let* $\mathcal{L} = (\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$ *be the language of* SPE *ciphertexts from Section 6. The zero-knowledge property allows us to simulate proofs computed for messages in* S *and honestly-generated randomness in* $\mathcal{R}$, *for statements in the language* $\mathcal{L}$ *(formally expressed in Definition 2.10).*

The following definition follows the write-up of [38], and is the security notion we require from our SPNIZK argument system.

Definition C.2 (Unbounded Simulation Soundness [23, 51]). *Consider a language* $\mathcal{L} = (\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$. *A NIZK argument system for* $\mathcal{L}$ *satisfies unbounded simulation soundness if no PPT adversary* $\mathcal{A}$ *wins the following game with non-negligible advantage:*

1. *The challenger chooses a membership testing trapdoor* τ_{zk} *for* $\mathcal{L}_{\mathsf{zk}}$. *Let* $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ *be the efficient NIZK simulator for* $\mathcal{L}$. *The challenger computes* $(\mathsf{crs}, \tau_{zk}) \leftarrow_{\mathrm{R}} \mathsf{Sim}_0(1^\lambda, \mathcal{L})$. *Then, it sends* $(\mathsf{crs}, \tau_{zk})$ *to adversary* $\mathcal{A}$.
2. *Adversary* $\mathcal{A}$ *is given oracle access to* $\mathsf{Sim}_1(\mathsf{crs}, \tau_{zk}, \cdot)$. *At each oracle query,* $\mathcal{A}$ *chooses a statement* x, *and obtains a simulated proof* $\pi \leftarrow_{\mathrm{R}} \mathsf{Sim}_1(\mathsf{crs}, \tau_{zk}, x)$.
3. *Finally,* $\mathcal{A}$ *outputs* (x^*, π^*).

We denote by Q *the set of all simulation queries and responses* (x_i, π_i) *made by* $\mathcal{A}$ *to* $\mathsf{Sim}_1(\mathsf{crs}, \tau_{zk}, \cdot)$. *Adversary* $\mathcal{A}$ *wins if all the following conditions hold:*

1. $(x^*, \pi^*) \notin Q$, *meaning that* x^* *was not queried.*
2. $x^* \notin \mathcal{L}_{\mathsf{sound}}$.
3. $\mathsf{V}(\mathsf{crs}, x^*, \pi^*) = 1$, *meaning that* π^* *is an accepting proof.*

D Compiling the Sigma Protocol of Section 6 into an SPNIZK Argument with Unbounded Simulation Soundness

The following results show how to compile the sigma protocol from section Section 6 into an SPNIZK with unbounded simulation soundness, by using correlation-intractable hashing and a construction by [38].

Definition D.1 (Searchable Relation [18]). *A relation* $R \subseteq \mathcal{X} \times \mathcal{Y}$ *is said to be searchable in time* T *if there exists a function* $f : \mathcal{X} \to \mathcal{Y}$, *which is computable in time* T, *and if there exists* y *with* $(x, y) \in \mathcal{R}$, *we have that* $f(x) = y$.

Note that for every $x \in \mathcal{X}$, Definition D.1 ensures that there exists at most one $y \in \mathcal{Y}$ with $(x, y) \in \mathcal{R}$.

Definition D.2 (Correlation-Intractable Hash Function (CI-Hash) [17]). *Let* $\mathcal{R} = \{\mathcal{R}_\lambda \subseteq I_\lambda \times O_\lambda\}$ *be a set of relations for each security parameter* λ. *A collection* $\mathcal{H} = \{H_\lambda : K_\lambda \times I_\lambda \mapsto O_\lambda\}_{\lambda \in \mathbb{N}}$ *of (efficient) keyed hash functions is a* $\mathcal{R}$-correlation intractable hash *(CIH) family for* $\mathcal{R}$, *if for every (non-uniform) PPT adversary* $\mathcal{A}$, *it holds that*

$$\Pr_{\substack{k \leftarrow_{\mathrm{R}} K_\lambda \\ x \leftarrow_{\mathrm{R}} \mathcal{A}(k)}} [(x, H_\lambda(K, x)) \in \mathcal{R}_\lambda] = \mathsf{negl}(\lambda).$$

Theorem D.3 (CI-Hashing based on SIS, from [46]). *Consider* $\mathcal{C}$ *to be the class of functions that have output length* m, *and which can be implemented by boolean circuits of size* S *and depth* d. *Assuming that* $\mathsf{SIS}_{m,n,q,\beta}$ *is hard for sufficiently large* $\beta = m^{O(d)}$, *one can construct a correlation-intractable hash family for class* $\mathcal{C}$.

Lemma D.4 (Bad-Challenge Relation is Efficiently Searchable). *Consider sigma protocol Σ for language $\mathcal{L}$ given in Section 6. Let x denote the first flow of the sigma protocol Σ,* Chal *denote the challenge, and consider the relation R_{bad}, defined as follows:*

$$R_{\text{bad}} = \{(x, \mathsf{Chal}) : x \notin \mathcal{L} \text{ and there exists a third flow } z \text{ s.t } \Sigma.\mathsf{V}(x, \mathsf{Chal}, z) = 1\}$$

Then, it holds that relation R_{bad} is efficiently searchable.

The proof of Lemma D.4 is straightforward. R_{bad} is efficiently searchable due to the $\mathsf{BadChallenge}$ function being efficiently computable. Moreover, we note that decryption of SPE ciphertexts is in NC_1.

Theorem D.5 ([38], Theorems 3.2, 3.4). *Let* SPE *be a structure-preserving encryption scheme and consider the language $\mathcal{L} = (\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$ for the sigma protocol from Section 6 defined with respect to* SPE*. There exists a SPNIZK argument system for $\mathcal{L}$ with unbounded simulation soundness, assuming that the $\mathsf{SIS}_{m,n,q,\beta}$ is hard (where n grows with the security parameter, $m = n \log q$ and β grows polynomial with the security parameter), and relying on the security of the following primitives:*

- *A trapdoor Σ-protocol $\Pi' = (\mathsf{Gen}'_{\mathsf{par}}, \mathsf{Gen}'_{\mathcal{L}}, \mathsf{P}', \mathsf{V}')$ with challenge space $\mathcal{C}$ for the same language $\mathcal{L} = (\mathcal{L}_{\mathsf{zk}}, \mathcal{L}_{\mathsf{sound}})$, where the* $\mathsf{BadChallenge}$ *function of Π' runs in time at most T. Protocol Σ must have statistically special zero-knowledge.*
- *A correlation-intractable hash family $\mathcal{H} = (\mathsf{Gen}, \mathsf{Hash})$ with output length $\kappa \in \mathsf{poly}(\lambda)$, for the class of relations $\mathcal{R}_{\text{CI}}$, efficiently searchable in time at most T, where T denotes the maximal running time of algorithm* $\mathsf{BadChallenge}(\cdot,\cdot,\cdot,\cdot)$.

The proof of this theorem follows the steps in [38], and uses the construction of a CI-Hash function from Theorem D.3.

E Parameters for the VES Construction

SPS is an $\mathcal{F}$-structure-preserving signature with parameters f, S that can be efficiently computed using $\mathsf{ComputeSPSetsAndFunctions}$ and depend on vk, m and tag. Our framework supports this general case, although in all our concrete instantiations the set S is part of the vk and does not depend on m or tag. Recall that the verification functions f belong to a set $\mathcal{F}$.

$$\delta_{\max} = \max_{\mathsf{vk},\mathsf{tag},m} \left\{ \delta \middle| \begin{array}{c} \delta \text{ is the noise growth of } S, \text{ where} \\ (S, \cdot) \leftarrow \mathsf{ComputeSPSetsAndFunctions}(\mathsf{vk}, m, \mathsf{tag}) \end{array} \right\}$$

SPE is a structure-preserving encryption scheme as in Definition 5.1 and message space $\mathbb{Z}_q$. The SPE parameters comprise a structure-preserving set R with noise growth δ_R, along with noise levels $\nu_{\mathsf{init}}, \nu_{\mathsf{max}}$.

In order to have correctness, the chosen SPE must be $\mathcal{F}$-homomorphic and the maximum noise level has to be large enough to satisfy Eq. (1). For all of our concrete proposals of SPE schemes, ν_{max} can be chosen arbitrarily large (as long as it grows at most exponential in the security parameter) by increasing the size of the modulus. Thus among the SPE and SPS schemes presented in this paper the combinations shown in Table 1 are possible.

E.1 Efficiency Considerations

Let λ be the security parameter. Then SPE has dimension $n' = \lambda$ and modulus $q' = \text{poly}(\lambda)$. The CI-Hash of [46] is implemented using GSW encryption. The decryption algorithm of SPE must be expressible as an NC_1 circuit of depth $\mathcal{O}(\log \lambda)$—which is the case with the schemes analysed in this paper. Such an NC_1 circuit can be translated to a branching program of size $\mathcal{O}(\text{poly}(\lambda))$, and the GSW parameters are $q = \text{poly}(\lambda) = q'\text{poly}(\lambda)$ and $n = \lambda^{c-o(1)}$, where c is a constant that depends on the SPE decryption circuit. The output of the CI hash function consists of m bits, where $m = n\lceil \log(q) \rceil$. In addition, the compiler for obtaining an unbounded simulation-sound NIZK also contains the ciphertexts of a generalised lossy encryption scheme—and the entire construction requires a $\theta(\lambda)$ number of parallel repetitions.

While this machinery might sound daunting relative to pairing-based NIZK systems, the NIZK presented here remains the most efficient lattice-based construction which is secure in the standard model (for proving membership to structure-preserving sets). There are several reasons for this:

1. The CI-Hash requires homomorphic encryption, but no bootstrapping is required since SPE decryption circuits have low depth $c_{\mathsf{Dec}} \cdot \kappa_{\mathsf{SPE}}$, where κ_{SPE} is the size of SPE ciphertexts and c_{Dec} is a small constant $c_{\mathsf{Dec}} \leq 44$ (for example using the results of [9]).
2. It avoids expensive Karp reductions, which would be necessary if one used general purpose NIZKs such as the one of [46].

The standard model NIZK incurs a significant overhead when compared to the usage of lattice NIZKs in the ROM, which is why the proposed NIZK is only semi-efficient. For this reason, we do not provide more detailed efficiency comparisions with random-oracle implementations. At the same time, we note that a gap can also be observed between the Groth-Sahai NIZK and Fiat-Shamir compilations of more restricted sigma protocols that only lead to secure NIZKs in the ROM. Nevertheless, such a gap in the group setting appears to be smaller than in the lattice case.

F Formal definition of VES

We now present the formal definitions of VES and its security, taken almost verbatim from [50]:

Definition F.1 (Verifiably Encrypted Signatures (VES) [50]). *Let $\lambda \in \mathbb{N}$ denote the security parameter and $\mathcal{M}$ the message space. A verifiable encrypted signature (VES) is a tuple of PPT algorithms* $(\mathsf{Kg}, \mathsf{AdjKg}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf})$*, where:*

- Kg*,* Sig *and* Vf *are defined similarly to a digital signature scheme, namely:* $\mathsf{Kg}(1^\lambda)$ *generates signature keys* $(\mathsf{vk}, \mathsf{sk})$*,* $\mathsf{Sig}(\mathsf{sk}, m)$ *is the signing algorithm run on a message* $m \in \mathcal{M}$ *and* $\mathsf{Vf}(\mathsf{vk}, m, \sigma)$ *is the signature verification algorithm.*
- $\mathsf{AdjKg}(1^\lambda)$ *generates keys* $(\mathsf{apk}, \mathsf{ask})$ *for the adjudicator.*
- $\mathsf{Create}(\mathsf{sk}, \mathsf{apk}, m)$ *outputs a VES* Ω*.*
- $\mathsf{VesVf}(\mathsf{apk}, \mathsf{vk}, \Omega, m)$ *allows to verify a VES* Ω*, without knowing an unencrypted signature of* m*.*
- $\mathsf{Adj}(\mathsf{ask}, \mathsf{apk}, \mathsf{vk}, \Omega, m)$ *is given a VES* Ω*, and it computes a corresponding signature* σ *for* m *with respect to* vk*.*

Definition F.2 (Completeness of a Verifiably Encrypted Signatures (VES) [50]). *We say that a VES* $(\mathsf{Kg}, \mathsf{AdjKg}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf})$ *is complete if for all* $\lambda \in \mathbb{N}$*:*

$$\begin{aligned}
&\mathsf{VesVf}(\mathsf{apk}, \mathsf{vk}, \mathsf{Create}(\mathsf{sk}, \mathsf{apk}, m), m) = 1) \text{ and} \\
&\qquad\qquad\qquad \mathsf{Vf}(\mathsf{vk}, \mathsf{Adj}(\mathsf{ask}, \mathsf{apk}, \mathsf{vk}, \mathsf{Create}(\mathsf{sk}, \mathsf{apk}, m)), m) \geq 1 - \mathsf{negl}(\lambda) \\
&\qquad\qquad\qquad\qquad\qquad \text{for all } m \in \mathcal{M}, (\mathsf{apk}, \mathsf{ask}) \leftarrow_{\mathrm{R}} \mathsf{AdjKg}(1^\lambda) \text{ and } (\mathsf{sk}, \mathsf{vk}) \leftarrow_{\mathrm{R}} \mathsf{Kg}(1^\lambda).
\end{aligned}$$

Definition F.3 (Security of a Verifiably Encrypted Signatures (VES) [50]). *We say that a VES* $(\mathsf{Kg}, \mathsf{AdjKg}, \mathsf{Sig}, \mathsf{Vf}, \mathsf{Create}, \mathsf{VesVf})$ *is secure if the following properties hold:*

- ***Unforgeability*** *There does not exist a PPT adversary* $\mathcal{A}$ *which given access to the public keys and oracle access to* Create *and* Adj*, is able to compute a VES* Ω *for a message* m *that it has never queried to its oracles. Namely, for all* $\lambda \in \mathbb{N}$ *and for all PPT* $\mathcal{A}$*:*

$$\begin{aligned}
\Pr[&(\mathsf{apk}, \mathsf{ask}) \leftarrow_{\mathrm{R}} \mathsf{AdjKg}(1^\lambda), (\mathsf{vk}, \mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Kg}(1^\lambda), \\
&(m^*, \Omega^*) \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathsf{Create}(\mathsf{sk}, \mathsf{apk}, \cdot), \mathsf{Adj}(\mathsf{ask}, \mathsf{apk}, \mathsf{vk}, \cdot, \cdot)}(\mathsf{vk}, \mathsf{apk}) : \\
&\mathsf{VesVf}(\mathsf{apk}, \mathsf{vk}, \Omega^*, m^*) = 1 \land \\
&\mathcal{A} \text{ has not queried } \mathsf{Create}(\mathsf{sk}, \mathsf{apk}, \cdot) \text{ or } \mathsf{Adj}(\mathsf{ask}, \mathsf{apk}, \mathsf{pk}, \cdot, \cdot) \text{ on } m^*] \leq \mathsf{negl}(\lambda)
\end{aligned}$$

- ***Abuse freeness*** *requires that no malicious, PPT adjudicator with access to a* Create *oracle is able to output a valid VES for a message that it has never queried. Namely, for all* $\lambda \in \mathbb{N}$ *and for all PPT* $\mathcal{A}$*:*

$$\begin{aligned}
\Pr[&(\mathsf{apk}, \mathsf{ask}) \leftarrow_{\mathrm{R}} \mathsf{AdjKg}(1^\lambda), (\mathsf{vk}, \mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Kg}(1^\lambda), \\
&(m^*, \Omega^*) \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathsf{Create}(\mathsf{sk}, \mathsf{apk}, \cdot)}(\mathsf{apk}, \mathsf{ask}, \mathsf{vk}) : \mathsf{VesVf}(\mathsf{apk}, \mathsf{vk}, \Omega^*, m^*) = 1 \land \\
&\mathcal{A} \text{ has not queried } \mathsf{Create}(\mathsf{sk}, \mathsf{apk}, \cdot) \text{ on } m^*] \leq \mathsf{negl}(\lambda)
\end{aligned}$$

- ***Extractability*** *requires that no malicious signer which can create its own* vk *and is granted oracle access to* Adj *is able to efficiently output a valid VES Ω, from which the algorithm* Adj *is unable to extract a valid signature. Namely, for all $\lambda \in \mathbb{N}$ and for all PPT $\mathcal{A}$:*

$$\begin{aligned}&\Pr[(\mathsf{apk},\mathsf{ask}) \leftarrow_{\mathrm{R}} \mathsf{AdjKg}(1^\lambda), (m^*, \Omega^*, \mathsf{vk}^*) \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathsf{Adj}(\mathsf{ask},\mathsf{apk},\cdot,\cdot,\cdot)}(\mathsf{apk}),\\ &\quad \sigma^* \leftarrow \mathsf{Adj}(\mathsf{ask},\mathsf{apk},\mathsf{vk}^*,\Omega^*,m^*) :\\ &\quad \mathsf{VesVf}(\mathsf{apk},\mathsf{vk}^*,\Omega^*,m^*) = 1 \ \wedge \mathsf{Vf}(\mathsf{vk}^*,m^*,\sigma^*) = 0] \le \mathsf{negl}(\lambda)\end{aligned}$$

- ***Opacity*** *requires that no PPT adversary, given public keys* vk *and* apk *and oracle access to* Create *and* Adj*, can return a valid signature σ^* for some message m^*, provided it has not queried* Adj *on m^*. Namely, for all $\lambda \in \mathbb{N}$ and for all PPT $\mathcal{A}$:*

$$\begin{aligned}&\Pr[(\mathsf{apk},\mathsf{ask}) \leftarrow_{\mathrm{R}} \mathsf{AdjKg}(1^\lambda)), (\mathsf{vk},\mathsf{sk}) \leftarrow_{\mathrm{R}} \mathsf{Kg}(1^\lambda),\\ &\quad (m^*,\sigma^*) \leftarrow_{\mathrm{R}} \mathcal{A}^{\mathsf{Create}(\mathsf{sk},\mathsf{apk},\cdot),\mathsf{Adj}(\mathsf{ask},\mathsf{apk},\mathsf{vk},\cdot,\cdot)}(\mathsf{vk},\mathsf{apk}) :\\ &\quad \mathsf{Vf}(\mathsf{vk},\sigma^*,m^*) = 1 \ \wedge \mathcal{A} \text{ has not queried } \mathsf{Adj}(\mathsf{ask},\mathsf{apk},\mathsf{vk},\cdot,\cdot) \text{ on } m^*] \le \mathsf{negl}(\lambda)\end{aligned}$$

G Deferred Proofs for our VES Construction from Section 8

Lemma G.1. *Assuming the correctness of* $\mathsf{SPS},\mathsf{SPE}$ *and of the* SPNIZK *argument, the VES scheme from Fig. 3 is complete, in the sense of Definition F.2, for the choice of parameters from Appendix E.*

The proof follows directly from the correctness of Sig, SPNIZK and the properties of SPE. The verification checks for ct^2 succeed because the noise growth of the homomorphic operations does not exceed the ν_{max} parameter of SPE.

Lemma G.2. *Assuming the unbounded simulation soundness of* SPNIZK *and the unforgeability of* SPS *with parameters as in Appendix E, the VES scheme from Fig. 3 is unforgeable, in the sense of Definition F.3.*

Proof. We exhibit a hybrid argument between adversary $\mathcal{A}$ and the challenger of the unforgeability game. Game_0 is the VES unforgeability game from Definition F.3. In Game_1, we use unbounded simulation soundness to switch the crs to a simulated crs for which the challenger knows a simulation trapdoor τ_{zk}. Responses to Create queries are now VES Ω which contain simulated proofs computed with trapdoor τ_{zk}.

Game_2 is identical to Game_1, except that the challenger receives the adversary's forgery (m^*, Ω^*). It parses Ω^* as $(\mathsf{ct}^{1,*}, \pi^*, \mathsf{tag}^*)$ and decrypts $\mathsf{ct}^{1,*}$ to obtain core^*. The challenger aborts if $\sigma^* = (\mathsf{core}^*, \mathsf{tag}^*)$ is an invalid SPS signature, but Ω^* is accepted by $\mathsf{VES.VesVf}$. The probability of aborting is precisely the probability that the adversary breaks the unbounded simulation soundness of SPNIZK. Finally, we argue that the success probability of $\mathcal{A}$ in Game_2 is bounded by the probability to successfully break the existential unforgeability of SPS. This is the case because σ^* is a valid signature for m^*, but $\mathcal{A}$ has not queried m^* to its oracles, which corresponds precisely to producing a forgery for SPS. □

Lemma G.3. *Consider the choice of parameters from Section 8.1. Assuming the unbounded simulation soundness of* SPNIZK *and the unforgeability of* SPS*, the VES scheme from Fig. 3 has abuse-freeness, in the sense of Definition F.3.*

Proof. We need to show that an adversary $\mathcal{A}$ that possesses ask and has access to the Create oracle, cannot output a valid VES for a message that it hasn't queried to its oracle. The argument is similar to the proof of Lemma G.2. □

Lemma G.4. *Consider the choice of parameters from Section 8.1. Assuming the unbounded simulation soundness of* SPNIZK *and the unforgeability of* SPS*, the VES scheme from Fig. 3 satisfies extractability, in the sense of Definition F.3.*

Proof. Adversary $\mathcal{A}$ is allowed to create its own vk and has oracle access to Adj, with the objective of outputting a valid VES Ω^* from which Adj is unable to extract a valid signature σ^*. After switching to a hybrid where the crs and proofs are simulated, this directly contradicts the unbounded simulation soundness of SPNIZK. □

Lemma G.5. *Consider the choice of parameters from Section 8.1 and let* SPS *be a structure-preserving signature with superpolynomially-large tag space and uniform public tag distribution. Assuming the unbounded simulation soundness of the* SPNIZK *argument and the strong unforgeability of* SPS*, the VES scheme from Fig. 3 satisfies opacity, in the sense of Definition F.3.*

Proof. We follow the outline of the standard proof from [29, Section 5.1], but there are some differences with the group-based structure-preserving signatures in [29]. We need to account for the random tags which are part of the underlying signature, but which are not encrypted and are revealed in the VES. At the same time, the tags allow us to obtain a modified proof strategy. We proceed by a hybrid argument, where Game_0 is the opacity game from Definition F.3. We denote by $\epsilon_i = \Pr[\text{Game}_i = 1]$, the probability that the adversary successfully wins Game_i.

- Game_1 is identical to Game_0, but we switch the crs of the SPNIZK to a simulated crs. The proofs of the SPNIZK argument are now switched with simulated proofs. From the zero-knowledge property of the NIZK, we have:

$$\epsilon_0 \leq \epsilon_1 + \mathsf{negl}(\lambda)$$

- Game_2 is identical to Game_1, but we abort if any two Create queries for messages m_1, m_2 yield Ω_1 and Ω_2 that contain the same unencrypted public value tag. Since we assumed that the tag space is superpolynomial and tags are chosen uniformly at random during honest signature generation, the probability of tag collisions is negligible. We therefore have:

$$\epsilon_1 \leq \epsilon_2 + \mathsf{negl}(\lambda)$$

- Game_3 is identical to Game_2, except that we abort if the adversary makes a query to its Adj oracle on a valid VES $\Omega' = (\mathsf{ct}'^1, \pi', \mathsf{tag}')$ and message m', where ct^1 decrypts to core' and the tuple $(m', \mathsf{tag}', \mathsf{core}')$ has not been previously used to answer a previous Create oracle query on m' (meaning that at least one of m', tag' or core' is fresh). The crucial aspects of this abort condition are that the VES Ω' must be a valid signature and that the abort condition is efficiently checkable by the challenger.

  Because Game_3 has an additional abort condition, we have that $\epsilon_2 \geq \epsilon_3$, since it is now harder for the adversary to win the experiment. Nevertheless, we will show that ϵ_3 cannot decrease too much.

  In fact, the adversary can induce an abort if it either manages to find a valid SPS signature σ' for m', or by forging a proof for an invalid signature σ'. By the strong existential unforgeability of the SPS signature and unbounded simulation soundness of the NIZK, we have:

$$\epsilon_2 \leq \epsilon_3 + \mathsf{negl}(\lambda)$$

- Game_4 This game is identical to Game_3, except that we change how the Adj and Create oracles work. Notice first that the modifications in Game_2 allow us to argue that every response $\Omega = (\mathsf{ct}^1, \pi, \mathsf{tag})$ from the Create oracle can be uniquely mapped to the corresponding message and core generated during signing, but crucially without requiring decryption. This is because we have assumed in Game_2 that every auxiliary information tag appears only once, so it uniquely links every queried message m to the signature $(\mathsf{core}, \mathsf{tag})$ that was generated for it during that Create query. (Multiple queries on the same message would lead to multiple signatures, but the tags are always unique.)

  Every Create query on a message m^* first generates an SPS signature $(\mathsf{core}^*, \mathsf{tag}^*)$ and then computes $\Omega^* = (\mathsf{ct}^{1,*}, \pi^*, \mathsf{tag}^*)$. The challenger then stores the tuple $(m^*, \mathsf{tag}^*, \mathsf{core}^*)$ in a list.

  On input $(\Omega = (\mathsf{ct}^1, \pi, \mathsf{tag}), m)$, Adj doesn't decrypt. Instead Adj first checks if the queried Ω contains a valid proof π. If π is not valid, then Adj simply returns $\perp$. Otherwise, Adj uses (m, tag) to search the stored list for a tuple that contains $(m, \mathsf{tag}, \mathsf{core})$ and recover the proper vector core corresponding to m. If (m, tag) does not appear in any of the stored tuples, then the challenger aborts and the adversary fails the experiment.

  We now analyze the differences between Adj behavior in Game_3 and Game_4:

  - **Type 0 queries:** the adversary queries Adj on a VES Ω that contains an invalid proof. In both Game_3 and Game_4, Adj simply returns $\perp$.
  - **Type 1 queries:** the adversary queries Adj on a VES Ω that contains a valid proof, but for a tuple (m, tag) that hasn't appeared in a previous Create query. From the changes in Game_2, the challenger aborts in this situation and the adversary loses the game in both Game_3 and Game_4.

- **Type 2 queries:** the adversary queries Adj on a VES $\Omega = (\mathsf{ct}'^1, \pi', \mathsf{tag}'), m)$ that contains a valid proof π', but for a tuple (m', tag') that **has appeared** in a previous Create query. Since (m', tag') appears in the list, we can recover a corresponding vector core' and return it as the result of the Adj oracle. In this case, the view of the adversary might start diverging from its view in Game_3.

We analyze more carefully the difference between the adversary view in Game_4 and the differences from its view in Game_3. First we observe that Create queries have the same effect in the two games. The only oracle queries that can lead to an increase in adversary advantage are Adj queries. Let DIFF_i denote the event that the i^{th} Adj query leads to an abort in Game_3, but not in Game_4. For Adj queries of type 0 and 1, event DIFF_i always has probability 0.

Only Adj queries of type 2 have potentially non-zero probability. Denote by event DIFF the event that at least one DIFF_i occurs, for $i = 1 \dots Q_{\mathrm{Adj}}$. Then we have that:

$$\begin{aligned}\epsilon_4 &= \Pr[\mathrm{Game}_4 = 1 \mid \neg\mathsf{DIFF}] \cdot \Pr[\neg\mathsf{DIFF}] + \Pr[\mathrm{Game}_4 = 1 \mid \mathsf{DIFF}] \cdot \Pr[\mathsf{DIFF}] \\ &\overset{(*)}{=} \Pr[\mathrm{Game}_3 = 1 \mid \neg\mathsf{DIFF}] \cdot \Pr[\neg\mathsf{DIFF}] + \Pr[\mathrm{Game}_4 = 1 \mid \mathsf{DIFF}] \cdot \Pr[\mathsf{DIFF}] \\ &\geq \Pr[\mathrm{Game}_3 = 1 \mid \neg\mathsf{DIFF}] \cdot \Pr[\neg\mathsf{DIFF}] + \underbrace{\Pr[\mathrm{Game}_3 = 1 \mid \mathsf{DIFF}]}_{=0} \cdot \Pr[\mathsf{DIFF}] = \epsilon_3,\end{aligned}$$

where $(*)$ uses that Game_3 and Game_4 proceed identically *until* DIFF occurs. We have thus shown that:

$$\epsilon_3 \leq \epsilon_4$$

- Game_5 This game is identical to Game_4, except that we switch all encryptions in VES signatures to encryptions of 0. We can do this because the NIZK argument proofs are simulated using the simulation trapdoor, and we do not need to decrypt any encryption submitted through an Adj query. From the IND-CPA security of the encryption scheme,

$$\epsilon_4 \leq \epsilon_5 + \mathsf{negl}(\lambda)$$

As a technical subtlety, we remind the reader that the $\mathsf{BadChallenge}$ function requires the decryption key of the encryption scheme in order to ensure the unbounded simulation soundness of the underlying NIZK. Nevertheless, we do not require simulation soundness at this point in the hybrid argument.

We now prove that the probability to win Game_5 is negligible, by exhibiting a reduction to the strong unforgeability of the SPS signature scheme. The reduction interacts with the challenger of the strong unforgeability experiment, and it receives the verification key of the signature, which it uses to construct the public parameters of the verifiably encrypted signature.

First, our reduction guesses whether the adversary will forge on a message it has not has not queried to the Create oracle before, or on a message m_k, where m_k is the input to the k^{th} Create query. (The second situation is why we need strong unforgeability here.) The probability of a correct guess is $\frac{1}{Q_{\mathrm{Create}}+1}$.

The guessing step is necessary. Even though Create queries are answered using encryptions of 0, it is not possible to only request signatures from the strong unforgeability challenger during Adj calls. This is because in order to answer Create queries on messages m_i, we cannot simply simulate the entire verifiable encrypted signature, since the public auxiliary tag must correspond to the core part of an actual signature. During an eventual Adj query, the adversary will be able to check whether $(\mathsf{core}, \mathsf{tag})$ are valid. This is due to the adversary being able to ask Adj queries on VES Ω_i, which should output valid signatures as long as they correspond to a Create query on m_i, with $i \neq k$.

Therefore, for each Create query on message m_i with $i \neq k$, the reduction asks for a signature $(\mathsf{core}_i, \mathsf{tag}_i)$ on message m_i. It then encrypts core_i as ct_i^1 and outputs $\Omega_i = (\mathsf{ct}_i^1, \pi_i, \mathsf{tag}_i)$.

The adversary outputs a signature $\sigma^* = (\mathsf{core}^*, \mathsf{tag}^*)$ on its chosen message m^*, and the reduction forwards (σ^*, m^*) to the challenger.

To summarize, the guessing phase distinguishes between the following two scenarios:

- The adversary will forge on a message m_k, where m_k is the input to the k^{th} Create query. The challenger will generate a random tag for the response of the k^{th} Create query. The other Create queries are answered with tags that correspond to valid signatures obtained from the strong-unforgeability challenger.

- The adversary will forge on a message m^* that was never asked in a Create query. Then we can deal with Create queries by asking for valid signatures from the strong-unforgeability challenger. Existential unforgeability is sufficient in this second scenario.

Therefore, from the strong unforgeability of the SPS scheme,

$$\epsilon_5 \leq \mathsf{negl}(\lambda)$$

At a high level, the proof strategy ensures that the adversary cannot get any additional advantage by somehow manipulating the Adj and Create oracles in unintended ways. Create queries correspond to encryptions of 0, ensuring that each Adj query will correspond to a query that can be bijectively mapped to a query to the strong-unforgeability challenger. $\square$