Efficient Laconic Cryptography from Learning With Errors

Nico Döttling[1], Dimitris Kolonelos[2,3], Russell W. F. Lai[4], Chuanwei Lin[1], Giulio Malavolta[5], and Ahmadreza Rahimi[5]

[1] CISPA Helmholtz Center for Information Security
nico.doettling@gmail.com, chuanwei.lin@cispa.de
[2] IMDEA Software Institute
dimitris.kolonelos@imdea.org
[3] Universidad Politecnica de Madrid
[4] Aalto University
russell.lai@aalto.fi
[5] Max Planck Institute for Security and Privacy
giulio.malavolta@hotmail.it, ahmadreza.rahimi@mpi-sp.org

Abstract. Laconic cryptography is an emerging paradigm that enables cryptographic primitives with sublinear communication complexity in just two messages. In particular, a two-message protocol between Alice and Bob is called *laconic* if its communication and computation complexity are essentially independent of the size of Alice's input. This can be thought of as a dual notion of fully-homomorphic encryption, as it enables "Bob-optimized" protocols. This paradigm has led to tremendous progress in recent years. However, all existing constructions of laconic primitives are considered only of *theoretical interest*: They all rely on non-black-box cryptographic techniques, which are highly impractical.

This work shows that non-black-box techniques are not necessary for basic laconic cryptography primitives. We propose a *completely algebraic* construction of laconic encryption, a notion that we introduce in this work, which serves as the cornerstone of our framework. We prove that the scheme is secure under the standard Learning With Errors assumption (with polynomial modulus-to-noise ratio). We provide proof-of-concept implementations for the first time for laconic primitives, demonstrating the construction is indeed practical: For a database size of 2^{50}, encryption and decryption are in the order of single digit *milliseconds*.

Laconic encryption can be used as a black box to construct other laconic primitives. Specifically, we show how to construct:

- Laconic oblivious transfer
- Registration-based encryption scheme
- Laconic private-set intersection protocol

All of the above have essentially optimal parameters and similar practical efficiency. Furthermore, our laconic encryption can be preprocessed such that the online encryption step is entirely combinatorial and therefore much more efficient. Using similar techniques, we also obtain identity-based encryption with an unbounded identity space and tight security proof (in the standard model).

1 Introduction

Laconic cryptography [17,40,22,20] is an emerging paradigm to securely compute on large amounts of data in just two messages, while incurring very small communication. Specifically, in the laconic setting the receiver Alice has an input of very large size, whereas we typically think of the sender Bob's input as smaller in size. In the first message, Alice publishes a succinct hash h of her input D, which may be thought of as a large database $D \in \{0,1\}^n$. Such a compressing hash function cannot be unkeyed, therefore laconic protocols also rely on public parameters, which are typically also required to be succinct[6]. Given the hash h, Bob can encrypt his input x with respect to h, obtaining a succinct ciphertext ctxt. Importantly, the workload of Bob should also be independent of n. Such a ciphertext ctxt enables Alice to compute a joint function of her input D and Bob's input x, while Bob has the guarantee that Alice learns nothing but the legitimate function output. The specific choice of the function f computed by such a protocol leads to different laconic primitives:

- In laconic OT [17], Bob's input consists of an index i and two messages m_0 and m_1. The function f is given by $f(D,(i,m_0,m_1)) = (i, m_{D[i]})$, i.e. Alice learns the index i, and if the i-th bit of the database D is 0 she learns m_0, otherwise m_1. The setting of laconic OT typically imposes an additional efficiency requirement concerning Alice. Concretely, we require Alice's second phase to have a runtime essentially independent of n.
- In laconic function evaluation (LFE) [40], Alice's input D is a (large) boolean circuit C, and the function computed by an LFE protocol is $f(C,x) = C(x)$. The construction provided in [40] satisfies a somewhat relaxed succinctness guarantee: While the size of the communication does not scale with the size of the circuit C, it scales polynomially with the depth of C. Furthermore, the runtime of the second phase of Alice scales linearly with the size of C.

Implications. The notion of laconic OT in particular has had broader implications: The core-ideas underlying laconic OT led to a series of constructions of identity-based encryption (IBE) from weaker assumptions [19,18,21,15] and gave rise to the notion of registration-based encryption (RBE) [24,25,33]. These constructions make essential use of the above-mentioned more stringent efficiency-property of the laconic OT constructions they are based on. Consequently, these primitives are not known to be generically constructible from LFE.

Furthermore, the techniques developed in the context of laconic cryptography were key to making progress on a broad range of problems: trapdoor functions from the computational Diffie-Hellman assumption [23], private-information retrieval (PIR) from the decisional Diffie-Hellman assumption [22], two-round multi-party computation protocols from minimal assumptions [26,28,8], adaptively secure garbled circuits [27], laconic conditional disclosure of secrets [20], and laconic private set intersection [3,7].

[6] That is, independent or at least sublinear in n.

Reverse Delegation. Laconic cryptography can be seen as enabling *reverse delegation* without requiring additional rounds of communication. In a standard delegation scheme, a user outsources its computation to an untrusted server with the goal of learning the output while keeping its input private. The canonical cryptographic tool that enables delegation is fully-homomorphic encryption (FHE) [29], since it allows the server to perform the computation without knowing the user's input. Reverse delegation allows a user (Bob, in our previous example) to delegate the computation completely to the server (Alice) while also letting her learn the output of the computation and nothing beyond that. For instance, [17] provided a protocol to let Bob reverse-delegate RAM computations to Alice, such that Bob's overhead and the size of the communication scales only with runtime of the RAM program, but not with the size of Alice's (large) input. Likewise, the laconic function evaluation scheme of [40] allows to reverse-delegate circuit computations to Alice, while incurring a communication overhead that only scales with the depth of the circuit.

A Non-Blackbox "Barrier" for Practicality. So far, the aforementioned progress in designing new cryptographic primitives has been almost exclusively of *theoretical interest*. In essence, the lack of practicality of these new solutions can be explained by their *non-blackbox* use of underlying cryptographic building blocks. For example, essentially all known constructions of laconic OT involve a re-encryption step, also called *deferred encryption* [15], which gives the receiver Alice the ability to produce ciphertexts under keys that were not known to the sender Bob at the time of encryption. In the above-mentioned constructions, this re-encryption step is implemented using garbled circuits [42] for circuits which perform public-key cryptographic operations. The non-black box use of cryptographic primitives is such a grave source of inefficiency that, to the best of our knowledge, not even the basic laconic OT has ever been implemented as a proof of concept. On a slightly different note, we remark that while the LFE scheme of [40] does not make use of garbled circuits, it relies on a different non-blackbox mechanism based on FHE to bootstrap a weaker notion called attribute-based LFE into fully-fledged LFE.

In summary, the present state of affairs sees laconic cryptography as a powerful theoretical tool for enabling new cryptographic primitives and realizing powerful notions from weaker assumptions. However, the resulting schemes are practically inefficient, thus calling into question the relevance of this framework beyond theoretical feasibility results. Motivated by this gap, we ask:

Can we realize truly efficient laconic cryptography?

Towards a positive resolution to this question, it seems insufficient to optimize existing techniques. Instead, a conceptual reworking of basic laconic primitives will be required.

1.1 Our Results

This work shows that garbled circuits (and other non-black box cryptographic techniques) are not needed to construct laconic cryptography. We establish a

new paradigm for constructing *concretely efficient* laconic cryptographic schemes based on the hardness of the standard learning with errors (LWE) problem with a polynomial modulus-to-noise ratio. In contrast to prior works, we show that our schemes are practical with a proof of concept implementation. In the following, we discuss our contributions in more detail.

Laconic Encryption. We propose the notion of *laconic encryption* as the central abstraction of our framework. Laconic encryption allows Alice to construct a binary tree whose leaves are public keys $(\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$ and sends the root of the tree to Bob. Given only the root of the tree and an index ind, Bob can then encrypt a message with respect to $\mathsf{pk}_{\mathsf{ind}}$, which can only be decrypted with the corresponding secret key $\mathsf{sk}_{\mathsf{ind}}$. Such a scheme is called laconic since Alice's message is independent of n, as she only sends the root of the tree.

We then show how to construct laconic encryption *efficiently* and with *(asymptotically) optimal parameters* without relying on garbled circuits or other non-black box cryptographic techniques. At a technical level, our construction relies on the algebraic properties of the SIS-based hash tree. It exploits the gadget matrix to efficiently re-encrypt the message layer-by-layer. In order to demonstrate the security of the scheme, we introduce a new variant of the (ring/module) LWE problem, in which the adversary is also given a leakage on the error. Then we prove that this problem is as hard as the standard (ring/module) LWE problem, with an essentially tight reduction. Our proof relies on spectral analysis of positive definite matrices, a subject of independent interest.

Applications. We show how laconic encryption enables a wide range of laconic cryptographic primitives with minimal overhead. The following constructions use laconic encryption in a black-box sense, and the additional methods required are combinatorial. That is, all of the resulting schemes are *concretely efficient* and have near-optimal parameters. Specifically, we show how to construct:

- **Laconic OT**: As an immediate application of laconic encryption, we construct a laconic OT protocol with essentially optimal parameters.
- **Registration-Based Encryption**: Registration-based encryption (RBE) is a notion recently introduced in [24] to solve the key-escrow problem for identity-based encryption (IBE) while preserving the "encrypt with respect to identity" functionality. Laconic encryption enables the first concretely efficient RBE construction that the size of the public parameters scales *logarithmically* with the number of users.
- **Laconic Private-Set Intersection**: Private-set intersection (PSI) allows Alice and Bob to check whether they have a common item in their database without revealing anything about other items. Laconic encryption allows us to construct an efficient *laconic* PSI protocol where the communication complexity is independent of the size of Alice's database.

Optimizations and Extensions. We explore a number of optimizations and extensions for our laconic encryption construction. First, we show that the encryption algorithm can be *pre-processed*: In an input-independent offline phase,

the encryptor can prepare auxiliary information at essentially the same cost as the encryption algorithm. In an online phase, where the message msg and the index ind are known, the encryptor can use the auxiliary information prepared earlier to produce a correctly-formed ciphertext. Importantly, the online phase is entirely combinatorial, and all public-key operations happen in the offline phase.

Second, we explore the possibility of plugging-in different encryption schemes in our construction. Natively, our laconic encryption supports only dual-Regev ciphertexts [30], whereas for some applications it may be desirable to use support other encryption schemes. We show how our scheme can be adapted to support a large class of algorithms, which includes LPN-based encryption [5] and recently NIST-standardized lattice-based schemes [10]. To solve this challenge, we develop a new special-purpose randomized encoding scheme, which may be of independent interest.

Finally, we show that our construction of laconic encryption can be turned into that of identity-based encryption (IBE) [9] with similar efficiency properties. Our IBE is the first scheme that simultaneously achieves: (i) Constant-size public parameters, (ii) an *unbounded* identity space, (iii) a tight proof of (adaptive) security against a standard assumption (specifically, LWE).

Implementation and Benchmark. To demonstrate the practicality of our laconic encryption scheme, we implemented a proof of concept in Go (see the full version for more details). We ran the benchmarks for the scheme with a database size/index space of 2^{50} and achieved encryption and decryption times below 10 milliseconds on a personal computer. We believe these times can be improved using further optimizations, which are beyond the scope of this work.

1.2 Related Work

We mention prior works that study practical variants of laconic cryptographic primitives. In [7] the authors show a variant of laconic private-set intersection that is practically efficient and leads to substantial improvements in real-world protocols. However, the variant that is implemented has a long common reference string, linear in the size of Alice's database D; thus it is not fully laconic.

In [35] the authors propose the notion of registered attribute-based encryption, as an extension of the notion of RBE, and they show a constructions based on bilinear pairings. Compared to our work, their scheme has a long common reference string (in fact, quadratic in n), the runtime of the key generation and registration algorithms is linear in n, and they have an a-priori bound on the number of users. On the flip-side, they achieve the attribute-based functionality, that we do not consider in this work.

Another recent work [31] proposes the first practically efficient registration-based encryption scheme, and shows the first proof of concept implementation. Contrary to this work, their scheme is asymptotically only sublinear in the size of D (specifically, $\sqrt{n}$ as opposed to $\mathsf{polylog}(n)$), and requires an a-priori bound on n. Furthermore, they rely on the hardness of problems over bilinear pairings and thus their scheme is immediately insecure in the quantum settings.

2 Technical Overview

We give a brief overview of the new ideas and the technical innovations introduced in this work. We start by describing the new notion of laconic encryption, how to construct it efficiently from the standard LWE assumptions, and the challenges that arise during the security proofs. Then we outline the new cryptographic schemes that are enabled by this new notion and possible optimizations and extensions. In favor of a more intuitive description, the following outline considers the special case of $\mathbb{Z}$-lattices; however, in the technical sections, we prove all of our statements for the more general $\mathcal{R}$-module settings.

2.1 Laconic Encryption

Before delving into the description of our scheme, we introduce the syntax of laconic encryption, and we recall how prior work (implicitly) addresses the challenges needed to build this notion.

Syntax and Properties. A laconic encryption scheme allows Alice to (iteratively) construct a digest (e.g., via a Merkle hash tree) of public keys $(\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$ where $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KGen}(\mathsf{pp})$ and pp can be thought of as a uniformly random string, which is common to all participants. We denote by st the message that Alice sends to Bob, which consists of the digest (e.g., the root of the Merkle tree). Importantly, the size of pp and st is only polynomial in the security parameter, and in particular, it does not depend on n. On input a message msg and an index $\mathsf{ind} \in [n]$, Bob can then compute a ciphertext $\mathsf{ctxt} \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{msg})$. Correctness requires that anyone possessing the corresponding secret key can decrypt ctxt, more specifically:

$$\mathsf{msg} = \mathsf{Dec}(\mathsf{sk}_{\mathsf{ind}}, \mathsf{wit}_{\mathsf{ind}}, \mathsf{ctxt})$$

where $\mathsf{wit}_{\mathsf{ind}}$ is some (public) auxiliary information, whose size is logarithmic in n. The reader can think of this information as being the Merkle tree opening, i.e., the root-to-leaf path, of the key $\mathsf{pk}_{\mathsf{ind}}$. For security, we require that if the adversary does not know the secret key associated with index ind (or if no key is added to the tree at that particular index), then:

$$\mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{msg}_0) \approx \mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{msg}_1).$$

In fact, we will require (and prove) a slight strengthening of this property, i.e., that ciphertexts should look pseudorandom to anyone who cannot decrypt them.

Prior Works. To gain some intuition on why constructing laconic encryption is a challenging problem, it is useful to recall how prior works [17] (implicitly) build this cryptographic primitive. Loosely speaking, their main leverage is a construction of a structured two-to-one hash function Hash (which can be constructed from a variety of computational assumptions) that supports an *encryption* functionality. More specifically, given a digest $d \leftarrow \mathsf{Hash}(D)$, Bob can compute a

ciphertext $\mathsf{ctxt} \leftarrow \mathsf{Enc}(d, \mathsf{ind}, (\mathsf{msg}_0, \mathsf{msg}_1))$ that allows Alice (who knows the database D) to recover $\mathsf{msg}_{D_{\mathsf{ind}}}$, whereas the message $\mathsf{msg}_{D_{1-\mathsf{ind}}}$ remains computationally hidden. While this looks like a promising start, it should be noted that the hash function is only two-to-one, and therefore the size of the digest d is only half that of the original database D. If one were to naively recurse this scheme, the encryption algorithm would quickly start running in exponential time.

To circumvent the runtime issue, the strategy of [17] is to rely on garbled circuits [42]. More specifically, to boost the compression of the hash function, they define a binary tree of hash values and use garbled circuits to (asymptotically efficiently) implement a re-encryption gadget from one layer to another. Given a digest $d_i \leftarrow \mathsf{Hash}(D_{i+1})$, where $D_{i+1} = (d_{i+1,0}, d_{i+1,1})$ are the digests at a lower layer, the encryption algorithm uses the above procedure to encrypt the *labels* of a garbled circuit, that internally runs the encryption Enc for the layer below. Crucially, the size of the labels is independent of the size of the garbled circuit (except for its input) and therefore this encryption strategy can be recursed without incurring an exponential blow-up. Although this framework achieves asymptotically optimal parameters, it is prohibitively expensive to use garbled circuits for public-key operations. In contrast, our strategy (described below in detail) will bypass this barrier by leveraging the algebraic properties of a particular hash function.

Our Approach. As hinted above, a strategy of constructing laconic encryption is to design a mechanism allowing to "encrypt with respect to a Merkle tree opening", and successfully executing this strategy requires an "encryption-friendly" hash function. Our starting point is the following variant of Ajtai's [2,32] collision-resistant hash function based on the short integer solution (SIS) assumption:

$$f(\mathbf{x}_0, \mathbf{x}_1) \coloneqq \mathbf{A}_0(-\mathbf{G}^{-1}(\mathbf{x}_0)) + \mathbf{A}_1(-\mathbf{G}^{-1}(\mathbf{x}_1)) \bmod q$$

where $\mathbf{A}_0, \mathbf{A}_1 \in \mathbb{Z}_q^{n \times m}$ are uniformly random matrices with $m \approx n \log q$, $\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{Z}_q^n$ are vectors, and $\mathbf{G}^{-1}$ denote the binary-decomposition operator (so that for any $\mathbf{x} \in \mathbb{Z}_q^n$ we have $\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{x}) = \mathbf{x}$). A very similar hash function was used in [36] to build lattice-based Merkle-tree accumulators, ring signatures, and group signatures. At first glance, it may seem that the hash function f is not encryption-friendly since the binary-decomposition operation $\mathbf{G}^{-1}$ is highly non-linear. What enables us to encrypt with respect to a Merkle tree opening is the crucial observation that a hash chain formed by f *induces* a linear relation.

More concretely, consider the Merkle tree built using the hash function f where the node indexed by $\mathsf{str} \in \{0,1\}^*$ is labeled by $\mathbf{y}_{\mathsf{str}}$. Suppose that $\mathbf{u}_{\mathsf{str}} = -\mathbf{G}^{-1}(\mathbf{y}_{\mathsf{str}})$ for each $\mathsf{str} \in \{0,1\}^*$. Closing into the top of the tree, we observe that $(\mathbf{u}_0, \mathbf{u}_1)$ is a short (in fact binary) vector satisfying the linear relation:

$$\begin{pmatrix} \mathbf{A}_0 & \mathbf{A}_1 \\ \mathbf{G} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \end{pmatrix} = \begin{pmatrix} \mathbf{y}_\epsilon \\ -\mathbf{y}_0 \end{pmatrix} \bmod q.$$

Where $\mathbf{y}_\epsilon$ is the node denoting the root of the tree. In other words, the vector $(\mathbf{u}_0, \mathbf{u}_1)$ is a valid solution to the (inhomogeneous) SIS instance

$$\left(\begin{pmatrix}\mathbf{A}_0 & \mathbf{A}_1\\ \mathbf{G} & \mathbf{0}\end{pmatrix}, \begin{pmatrix}\mathbf{y}_\epsilon\\ -\mathbf{y}_0\end{pmatrix}\right).$$

Likewise, $(\mathbf{u}_0, \mathbf{u}_1)$ is also a valid solution to the (inhomogeneous) SIS instance

$$\left(\begin{pmatrix}\mathbf{A}_0 & \mathbf{A}_1\\ \mathbf{0} & \mathbf{G}\end{pmatrix}, \begin{pmatrix}\mathbf{y}_\epsilon\\ -\mathbf{y}_1\end{pmatrix}\right).$$

Dual-Regev Encryption. It turns out that this structure synergizes remarkably well with the dual-Regev encryption scheme [30]. Recall that in the dual-Regev encryption scheme [30], whose security is based on the standard LWE assumption, a public key is a SIS instance and the corresponding secret key is the SIS solution. Specifically, in the following assume that the matrix $\mathbf{A} = (\mathbf{A}_0\ \mathbf{A}_1)$ is part of the public parameters. Further assume that $\mathbf{y}_0$ and $\mathbf{y}_1$ are dual-Regev public keys with respect to $\mathbf{A}$. That is, for $b \in \{0,1\}$ we generate $\mathbf{y}_b$ by choosing a uniformly random $\mathbf{w}_b \in \{0,1\}^{2m}$ and set $\mathbf{y}_b = \mathbf{A} \cdot \mathbf{w}_b \bmod q$. Here, $\mathbf{w}_b$ is the secret key corresponding to $\mathbf{y}_b$. By the leftover-hash-lemma [34,41], the $\mathbf{y}_b$ are statistically close to uniform. To encrypt a message msg under $\mathbf{y}_b$, we choose an LWE secret $\mathbf{r}_1$ and compute a ciphertext $(\mathbf{c}_1, d_1)$ via

$$\begin{aligned}\mathbf{c}_1 &\approx \mathbf{r}_1^\mathsf{T} \cdot \mathbf{A} \bmod q,\\ d_1 &\approx \mathbf{r}_1^\mathsf{T} \cdot \mathbf{y}_b + \mathsf{Encode}(\mathsf{msg}) \bmod q.\end{aligned}$$

Here, we use the "$\approx$" notation to omit the LWE error. The function $\mathsf{Encode}(\cdot)$ protects the message msg against small errors, a popular choice is to encode a message bit msg in the most-significant bit, i.e. $\mathsf{Encode}(\cdot) = \frac{q}{2} \cdot \mathsf{msg}$. To decrypt a ciphertext $(\mathbf{c}_1, d_1)$ using a secret key $\mathbf{w}_b$ we compute

$$d_1 - \mathbf{c}_1^\mathsf{T} \cdot \mathbf{w}_b \approx \mathbf{r}_1^\mathsf{T} \cdot \mathbf{y}_b + \mathsf{Encode}(\mathsf{msg}) - \mathbf{r}_1^\mathsf{T} \cdot \underbrace{\mathbf{A} \cdot \mathbf{w}_b}_{=\mathbf{y}_b} = \mathsf{Encode}(\mathsf{msg}) \bmod q,$$

from which the message msg can be efficiently recovered.

Encrypting to Hash Values. Now assume that we are not given $\mathbf{y}_0$ and $\mathbf{y}_1$, but only their hash value

$$\mathbf{y} = \mathbf{A} \cdot \begin{pmatrix}-\mathbf{G}^{-1}(\mathbf{y}_0)\\ -\mathbf{G}^{-1}(\mathbf{y}_1)\end{pmatrix} \bmod q.$$

Our goal is to produce a ciphertext "for the key $\mathbf{y}_b$" given only the hash value $\mathbf{y}$. Towards this goal, let us examine what happens when we generate a dual-Regev encryption scheme with respect to the "public key"

$$\mathsf{pk} := \left(\begin{pmatrix}\mathbf{A}_0 & \mathbf{A}_1\\ \mathbf{G} & \mathbf{0}\end{pmatrix}, \begin{pmatrix}\mathbf{y}\\ \mathbf{0}\end{pmatrix}\right),$$

Choosing LWE secrets $\mathbf{r}_0$ and $\mathbf{r}_1$ we compute a ciphertext $\mathsf{ctxt} = (\mathbf{c}, d)$ by

$$\mathbf{c}^\intercal \approx (\mathbf{r}_0^\intercal, \mathbf{r}_1^\intercal) \cdot \begin{pmatrix} \mathbf{A}_0 & \mathbf{A}_1 \\ \mathbf{G} & \mathbf{0} \end{pmatrix} = \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{r}_1^\intercal \cdot (\mathbf{G}\ \mathbf{0}) \bmod q$$
$$d \approx (\mathbf{r}_0^\intercal, \mathbf{r}_1^\intercal) \cdot \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix} + \mathsf{Encode}(\mathsf{msg}) = \mathbf{r}_0^\intercal \cdot \mathbf{y} + \mathsf{Encode}(\mathsf{msg}) \bmod q.$$

If we "decrypt" the ciphertext using $(\mathbf{u}_0 = -\mathbf{G}^{-1}(\mathbf{y}_0), \mathbf{u}_1 = -\mathbf{G}^{-1}(\mathbf{y}_1))$ as the secret key, we obtain

$$\begin{aligned} d - \mathbf{c}^\intercal \cdot \begin{pmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \end{pmatrix} &\approx \mathbf{r}_0^\intercal \cdot \mathbf{y} + \mathsf{Encode}(\mathsf{msg}) - \mathbf{r}_0^\intercal \cdot \underbrace{(\mathbf{A}_0\ \mathbf{A}_1) \cdot \begin{pmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \end{pmatrix}}_{=\mathbf{y}} - \mathbf{r}_1^\intercal \cdot \underbrace{(\mathbf{G}\ \mathbf{0}) \cdot \begin{pmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \end{pmatrix}}_{=-\mathbf{y}_0} \\ &= \mathbf{r}_1^\intercal \cdot \mathbf{y}_0 + \mathsf{Encode}(\mathsf{msg}) \bmod q. \end{aligned}$$

Consequently, this "decryption operation" has produced (part of) a ciphertext encrypted under the public key $\mathbf{y}_0$! Analogously, if we use the public key

$$\mathsf{pk} := \left(\begin{pmatrix} \mathbf{A}_0 & \mathbf{A}_1 \\ \mathbf{0} & \mathbf{G} \end{pmatrix}, \begin{pmatrix} \mathbf{y} \\ \mathbf{0} \end{pmatrix} \right),$$

the above decryption operation would result in a ciphertext component $\mathbf{r}_1^\intercal \cdot \mathbf{y}_1 + \mathsf{Encode}(\mathsf{msg}) \bmod q$. Thus, decryption of such ciphertext with $(\mathbf{u}_0 = -\mathbf{G}^{-1}(\mathbf{y}_0)$, $\mathbf{u}_1 = -\mathbf{G}^{-1}(\mathbf{y}_1))$ is effectively a re-encryption to either public key $\mathbf{y}_0$ or $\mathbf{y}_1$.

To make such a ciphertext decryptable under one of the corresponding secret keys, we add an additional ciphertext component $\mathbf{c}_1^\intercal = \mathbf{r}_1^\intercal \cdot \mathbf{A} + \mathbf{e}_1 \bmod q$ to ctxt. Then, a ciphertext ctxt for $\mathbf{y}_b$ comprises of

$$\begin{aligned} \mathbf{c}^\intercal &\approx \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{r}_1^\intercal \cdot ((1-b) \cdot \mathbf{G}\ b \cdot \mathbf{G}) \bmod q \\ \mathbf{c}_1^\intercal &\approx \mathbf{r}_1^\intercal \cdot \mathbf{A} \bmod q \\ d &\approx \mathbf{r}_0^\intercal \cdot \mathbf{y} + \mathsf{Encode}(\mathsf{msg}) \bmod q. \end{aligned}$$

Finally, observe that it doesn't matter if the $\mathbf{y}_b$ are actually dual-Regev public keys or itself a hash value, the ciphertext structures are identical! Hence, for a larger tree we can apply this mechanism recursively, which results in one additional ciphertext component $\mathbf{c}_i^\intercal \approx \mathbf{r}_i^\intercal \cdot \mathbf{A} + \mathbf{r}_1^\intercal \cdot ((1-b_i) \cdot \mathbf{G}\ b_i \cdot \mathbf{G}) \bmod q$ *per level of the tree*, where the b_i define the path through the tree.

Security of the Construction. We will now focus on establishing the security of this construction with the goal of basing security on the LWE assumption. For this purpose, we need to consider the error terms in our construction explicitly. Let $\mathbf{A} = (\mathbf{A}_0\ \mathbf{A}_1)$. A ciphertext $\mathsf{ctxt} = (\mathbf{c}, \mathbf{c}_1, d)$ for $b = 0$ is computed by

$$\begin{aligned} \mathbf{c}^\intercal &= \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{r}_1^\intercal \cdot (\mathbf{G}\ \mathbf{0}) + \mathbf{e} \bmod q \\ \mathbf{c}_1^\intercal &= \mathbf{r}_1^\intercal \cdot \mathbf{A} + \mathbf{e}_1 \bmod q \\ d &= \mathbf{r}_0^\intercal \cdot \mathbf{y} + e^* + \mathsf{Encode}(\mathsf{msg}) \bmod q, \end{aligned}$$

where $\mathbf{e}$, $\mathbf{e}_1$ and e^* are short error vectors.

On the face of it, this looks almost like a classical LWE encryption. Hence, one might try to reduce security directly to the LWE problem. That is, given LWE samples $(\mathbf{A}, \mathbf{v}^\intercal = \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{e}^\intercal \bmod q)$ and $(\mathbf{y}, v = \mathbf{r}_0^\intercal \cdot \mathbf{y} + e^* \bmod q)$ we can simulate a ciphertext by computing

$$\begin{aligned}\mathbf{c}^\intercal &= \mathbf{v}^\intercal + \mathbf{r}_1^\intercal \cdot (\mathbf{G}\ \mathbf{0}) \bmod q\\ \mathbf{c}_1^\intercal &= \mathbf{r}_1^\intercal \cdot \mathbf{A} + \mathbf{e}_1 \bmod q\\ d &= v + \mathsf{Encode}(\mathsf{msg}) \bmod q.\end{aligned}$$

By replacing $\mathbf{v}^\intercal$ and v by uniformly random values, as per the LWE assumption, the term d now hides msg and security follows.

However, upon closer inspection there is a problem with this approach: The matrix $\mathbf{A}$ and the vector $\mathbf{y}^\intercal$ are *not* independent from the view of an adversary. Specifically, the adversary knows an explicit relation between $\mathbf{A}$ and $\mathbf{y}^\intercal$, namely

$$\mathbf{y}^\intercal = \mathbf{A}_0 \cdot (-\mathbf{G}^{-1}(\mathbf{y}_0)) + \mathbf{A}_0 \cdot (-\mathbf{G}^{-1}(\mathbf{y}_1)) =: \mathbf{A} \cdot \mathbf{z} \bmod q,$$

as $\mathbf{y}_0$ and $\mathbf{y}_1$ are known to the adversary. Here $\mathbf{z} := \begin{pmatrix} -\mathbf{G}^{-1}(\mathbf{y}_0) \\ -\mathbf{G}^{-1}(\mathbf{y}_1) \end{pmatrix}$ is a binary (and thus short) vector (denoted $(\mathbf{u}_0, \mathbf{u}_1)$ above). For this reason, $\mathbf{v}^\intercal = \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{e}^\intercal \bmod q$ and $v = \mathbf{r}_0^\intercal \cdot \mathbf{y} + e^* \bmod q$ are easily distinguishable from uniformly random values: It holds that $v - \mathbf{v}^\intercal \cdot \mathbf{z} = e^* - \mathbf{e}^\intercal \cdot \mathbf{z} \bmod q$ is short, whereas for uniformly random $\mathbf{v}^\intercal$ and v this expression is, with high probability, not short.

Drowning Out Correlations. However, there is a fairly routine solution to this issue using a technique called *drowning*. The idea is, given LWE samples $(\mathbf{A}, \mathbf{v}^\intercal = \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{e} \bmod q)$, to *simulate* v from $\mathbf{v}$ and $\mathbf{z}$ by computing it via

$$v \approx \mathbf{v}^\intercal \cdot \mathbf{z} = (\mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{e}^\intercal) \cdot \mathbf{z} = \mathbf{r}_0^\intercal \cdot \mathbf{A} \cdot \mathbf{z} + \mathbf{e}^\intercal \cdot \mathbf{z} = \mathbf{r}_0^\intercal \cdot \mathbf{y} + \mathbf{e}^\intercal \cdot \mathbf{z} \bmod q.$$

Yet, now the error terms in $\mathbf{v}^\intercal$ and v are obliviously correlated. To get rid of this correlation, we can opt to *drown* it out: If e^* is chosen from a suitable short distribution which produces super-polynomially larger values than $\mathbf{e}^\intercal \cdot \mathbf{z}$, then it holds that $\mathbf{e}^\intercal \cdot \mathbf{z} + e^* \approx_s e^*$, i.e. $\mathbf{e}^\intercal \cdot \mathbf{z} + e^*$ and e^* are statistically close. Hence, we can simulate v by computing $v = \mathbf{v}^\intercal \cdot \mathbf{z} + e^* \bmod q$.

Hence, our security proof now proceeds as follows. Given LWE samples $(\mathbf{A}, \mathbf{v}^\intercal = \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{e}^\intercal \bmod q)$ we can simulate a ciphertext $\mathsf{ctxt} = (\mathbf{c}, \mathbf{c}_1, d)$ by sampling e^* and setting

$$\begin{aligned}\mathbf{c}^\intercal &= \mathbf{v}^\intercal + \mathbf{r}_1^\intercal \cdot (\mathbf{G}\ \mathbf{0}) \bmod q\\ \mathbf{c}_1^\intercal &= \mathbf{r}_1^\intercal \cdot \mathbf{A} + \mathbf{e}_1 \bmod q\\ d &= \mathbf{v}^\intercal \cdot \mathbf{z} + e^* + \mathsf{Encode}(\mathsf{msg}) \bmod q.\end{aligned}$$

If $(\mathbf{A}, \mathbf{v})$ are well-formed LWE samples, then by the above discussion,

$$d = \mathbf{v}^\intercal \cdot \mathbf{z} + e^* + \mathsf{Encode}(\mathsf{msg})$$

$$
\begin{aligned}
&= \mathbf{r}_0^\intercal \cdot \mathbf{y} + \mathbf{e}^\intercal \cdot \mathbf{z} + e^* + \mathsf{Encode}(\mathsf{msg}) \\
&\approx_s \mathbf{r}_0^\intercal \cdot \mathbf{y} + e^* + \mathsf{Encode}(\mathsf{msg}) \bmod q,
\end{aligned}
$$

i.e. such a $\mathsf{ctxt} = (\mathbf{c}, \mathbf{c}_1, d)$ is statistically close to a real ciphertext. Under the LWE assumption, we can now replace $\mathbf{v}$ with a uniformly random $\mathbf{v}'$ and get

$$
\begin{aligned}
\mathbf{c}^\intercal &= \mathbf{v}'^\intercal + \mathbf{r}_1^\intercal \cdot (\mathbf{G}\ \mathbf{0}) \bmod q \\
d &= \mathbf{v}'^\intercal \cdot \mathbf{z} + e^* + \mathsf{Encode}(\mathsf{msg}) \bmod q.
\end{aligned}
$$

Now, since $\mathbf{v}'$ is uniformly random, we can equivalently choose it by computing $\mathbf{v}'^\intercal = \mathbf{v}''^\intercal - \mathbf{r}_1^\intercal(\mathbf{G}\ \mathbf{0})$, where $\mathbf{v}''$ is also chosen uniformly random. That is, we compute $\mathsf{ctxt} = (\mathbf{c}, \mathbf{c}_1, d)$ by

$$
\begin{aligned}
\mathbf{c}^\intercal &= \mathbf{v}''^\intercal \\
\mathbf{c}_1^\intercal &= \mathbf{r}_1^\intercal \cdot \mathbf{A} + \mathbf{e}_1 \bmod q \\
d &= (\mathbf{v}''^\intercal - \mathbf{r}_1^\intercal \cdot (\mathbf{G}\ \mathbf{0})) \cdot \mathbf{z} + e^* + \mathsf{Encode}(\mathsf{msg}) \\
&= \mathbf{v}''^\intercal \cdot \mathbf{z} - \mathbf{r}_1^\intercal \cdot (\mathbf{G}\ \mathbf{0}) \cdot \mathbf{z} + e^* + \mathsf{Encode}(\mathsf{msg}) \\
&= \mathbf{v}''^\intercal \cdot \mathbf{z} + \mathbf{r}_1^\intercal \cdot \mathbf{y}_0 + e^* + \mathsf{Encode}(\mathsf{msg}) \bmod q,
\end{aligned}
$$

as $(\mathbf{G}\ \mathbf{0}) \cdot \mathbf{z} = -\mathbf{y}_0 \bmod q$. Going a step further, we can compute d by $d = \mathbf{v}''^\intercal \cdot \mathbf{z} + d_1 \bmod q$, where $d_1 = \mathbf{r}_1^\intercal \cdot \mathbf{y}_0 + e^* + \mathsf{Encode}(\mathsf{msg}) \bmod q$ is the payload part of an encryption of msg under the public key $\mathbf{y}_0$. In other words, we are now in a situation where we can simulate a ciphertext $\mathsf{ctxt} = (\mathbf{c}, \mathbf{c}_1, d)$ given and encryption $(\mathbf{c}_1, d_1)$ of msg under the public key $\mathbf{y}_0$! Hence, we can now immediately appeal to the fact that, from the view of the adversary, $\mathbf{y}_0$ looks indeed uniformly random to argue security: Via the LWE assumption, $(\mathbf{A}, \mathbf{r}_1^\intercal \cdot \mathbf{A} + \mathbf{e}_1 \bmod q)$ and $(\mathbf{y}_0, \mathbf{r}_1^\intercal \cdot \mathbf{y}_0 + e_1^* \bmod q)$ are indistinguishable from uniform. Thus, from the adversary's view d_1 looks uniformly random, and therefore $d = \mathbf{v}''^\intercal \cdot \mathbf{z} + d_1 \bmod q$ also looks uniformly random. In fact, from the adversary's view all ciphertext components look uniformly random and independent.

LWE with Error-Leakage. Drowning is, however, a rather heavy-handed approach that, for all intents and purposes, ruins the LWE parameters. Specifically, to use this approach we need to assume the security of LWE with *superpolynomial modulus-to-noise ratio*. This means, in turn, that the underlying worst-to-average case reduction of LWE [41] reduces LWE to worst-case lattice problems with super-polynomial approximation factors. Moreover, it forces us to use a superpolynomially large modulus q.

We will now look a bit closer at the above drowning step. Specifically, given $\mathbf{z}$ and $\mathbf{v}^\intercal = \mathbf{r}_0^\intercal \cdot \mathbf{A} + \mathbf{e} \bmod q$ we computed

$$
v = \mathbf{v}^\intercal \cdot \mathbf{z} + e^* = \mathbf{r}_0^\intercal \cdot \mathbf{A} \cdot \mathbf{z} + \mathbf{e}^\intercal \cdot \mathbf{z} + e^* = \mathbf{r}_0^\intercal \cdot \mathbf{y}_0 + \mathbf{e}^\intercal \cdot \mathbf{z} + e^* \bmod q.
$$

Our main observation is the following: If we were somehow given an *advice* $\mathbf{l} = -\mathbf{e}^\intercal \cdot \mathbf{z} + e^*$ about $\mathbf{e}$ and e^*, we could use $\mathbf{l}$ to *switch* the correlated error term $\mathbf{e}^\intercal \cdot \mathbf{z}$

in $\mathbf{v}^\mathsf{T} \cdot \mathbf{z}$ to a fresh and uncorrelated e^*. Namely by computing $v = \mathbf{v}^\mathsf{T} \cdot \mathbf{z} + \mathbf{l} \bmod q$. Then it holds that

$$v = \mathbf{v}^\mathsf{T} \cdot \mathbf{z} + \mathbf{l} = \mathbf{r}_0^\mathsf{T} \cdot \mathbf{y}_0 + \mathbf{e}^\mathsf{T} \cdot \mathbf{z} - \mathbf{e}^\mathsf{T} \cdot \mathbf{z} + e^* = \mathbf{r}_0^\mathsf{T} \cdot \mathbf{y}_0 + e^* \bmod q.$$

Thus, such an advice $\mathbf{l}$ is sufficient to make the security argument in the last paragraph work. Our hope now is that the advice $\mathbf{l} = -\mathbf{e}^\mathsf{T} \cdot \mathbf{z} + e^*$ *does not fully reveal* $\mathbf{e}$ and e^*, i.e. that $\mathbf{e}$ and e^* mutually conceal one another, even if the parameters of these error terms are way below the drowning regime.

This motivates the definition of *Learning with Errors with Error-Leakage*, elLWE for short. As the name suggests, in this variant of the LWE problem the adversary gets a *leak* or advice about the LWE error term. To make this definition useful for our purposes, we will allow the leak to depend on the LWE matrix $\mathbf{A}$. Consequently, we will define elLWE similarly to the regular LWE assumption, but via an *interactive experiment*. The security experiment of elLWE is given as follows, where we assume that a modulus q, dimensions n, m and error distributions χ, χ^* are parametrized by the security parameter.

The elLWE Security Experiment:

- In the first step, the experiment chooses a uniformly random matrix $\mathbf{A} \leftarrow\!\!\$\ \mathbb{Z}_q^{n \times m}$ and provides $\mathbf{A}$ to the adversary.
- Given the matrix $\mathbf{A}$, the adversary now chooses a *short* vector $\mathbf{z} \in \mathbb{Z}^m$ and provides $\mathbf{z}$ to the experiment.
- The experiment samples $\mathbf{e} \leftarrow\!\!\$\ \chi_1$ and $e^* \leftarrow\!\!\$\ \chi^*$ and sets $\mathbf{l} = \mathbf{e}^\mathsf{T} \cdot \mathbf{z} + e^*$.
- Now the experiment flips a random bit $b \leftarrow\!\!\$\ \{0,1\}$. If $b = 0$ it chooses a uniformly random $\mathbf{r} \leftarrow\!\!\$\ \mathbb{Z}_q^n$ and sets $\mathbf{v}^\mathsf{T} = \mathbf{r}^\mathsf{T} \cdot \mathbf{A} + \mathbf{e}^\mathsf{T} \bmod q$. If $b = 1$ it chooses $\mathbf{v} \leftarrow\!\!\$\ \mathbb{Z}_q^m$ uniformly at random.
- The experiment now provides $(\mathbf{A}, \mathbf{v}, \mathbf{l})$ to the adversary. The adversary then produces a guess $b' \in \{0,1\}$ for the bit b
- If $b' = b$ the adversary wins, and loses otherwise.

As usual, we say that elLWE is secure if no PPT adversary has non-negligible advantage in this experiment. Now, via the above discussion we can routinely reduce the security of our construction to elLWE.

We remark that the elLWE problem generalizes the *extended LWE problem* [38,6]. Specifically, in the extended LWE problem the vector $\mathbf{z}$ is chosen at random from a Gaussian distribution instead of adversarially (as in the case of the elLWE problem).

From LWE to elLWE. As an additional technical contribution of this work, we provide a hardness result for elLWE. Specifically, we show that the security of elLWE can be based on *standard LWE with polynomial modulus-to-noise ratio*. In this paragraph, we will sketch the main ideas underlying this result. In a nutshell, the main idea of our approach is to choose the leakage term $\mathbf{l}$ independent of the LWE error, and then *adjust* the LWE error in such a way that it *conforms* with the leakage. More precisely in the case of Gaussian $\mathbf{e}$ and e^*, we will show the following. There is a (sufficiently wide) Gaussian distribution $\hat{\mathbf{e}}$, such that for

every (short) vector $\mathbf{z}$ there is an *efficiently sampleable pair of correlated random variables* $(\mathbf{f}_\mathbf{z}, f_\mathbf{z})$ (independent of $\hat{\mathbf{e}}$), such that

$$(\mathbf{e}^\mathsf{T}, \mathbf{e}^\mathsf{T} \cdot \mathbf{z} + e^*) \approx_s (\hat{\mathbf{e}}^\mathsf{T} + \mathbf{f}_\mathbf{z}^\mathsf{T}, f_\mathbf{z}).$$

In other words, $f_\mathbf{z}$ simulates the leakage $\mathbf{e}^\mathsf{T} \cdot \mathbf{z} + e^*$, whereas $\mathbf{f}_\mathbf{z}$ can be used to *additively* adjust an independent Gaussian $\hat{\mathbf{e}}$ to have the same distribution as $\mathbf{e}$ given the leakage $\mathbf{e}^\mathsf{T} \cdot \mathbf{z} + e^*$. Equipped with such an efficiently sampleable pair $(\mathbf{f}_\mathbf{z}, f_\mathbf{z})$, reducing elLWE to LWE is almost straightforward: Given an LWE instance $(\mathbf{A}, \mathbf{v}^\mathsf{T})$ we run the elLWE adversary on $\mathbf{A}$, who returns $\mathbf{z}$. The reduction now samples $(\mathbf{f}_\mathbf{z}, f_\mathbf{z})$, provides $(\mathbf{A}, \mathbf{v}^\mathsf{T} + \mathbf{f}_\mathbf{z}^\mathsf{T}, f_\mathbf{z})$ to the adversary, and outputs whatever the adversary outputs.

On one side, if $\mathbf{v}^\mathsf{T}$ is an LWE sample, i.e. $\mathbf{v}^\mathsf{T} = \mathbf{r}^\mathsf{T} \cdot \mathbf{A} + \hat{\mathbf{e}} \bmod q$, then

$$\begin{aligned}(\mathbf{A}, \mathbf{v}^\mathsf{T} + \mathbf{f}_\mathbf{z}^\mathsf{T}, f_\mathbf{z}) &= (\mathbf{A}, \mathbf{r}^\mathsf{T} \cdot \mathbf{A} + \hat{\mathbf{e}} + \mathbf{f}_\mathbf{z}^\mathsf{T} \bmod q, f_\mathbf{z}) \\ &\approx_s (\mathbf{A}, \mathbf{r}^\mathsf{T} \cdot \mathbf{A} + \mathbf{e}^\mathsf{T} \bmod q, \mathbf{e}^\mathsf{T} \cdot \mathbf{z} + e^*),\end{aligned}$$

is statistically close to a correctly formed elLWE sample for $b = 0$.

On the other hand, if $\mathbf{v}$ is chosen uniformly random, then $\mathbf{v}' \coloneqq \mathbf{v} + \mathbf{f}_\mathbf{z} \bmod q$ is also uniformly random. Consequently $(\mathbf{A}, \mathbf{v} + \mathbf{f}_\mathbf{z} \bmod q, f_\mathbf{z}) \approx_s (\mathbf{A}, \mathbf{v}', \mathbf{e}^\mathsf{T}\mathbf{z} + e^*)$, i.e. it is statistically close to an elLWE sample for $b = 1$. The claim follows. Notice that this reduction is *tight*, i.e. it does not (substantially) degrade the adversary's runtime or advantage. Further notice that this reduction is agnostic of the structure of the matrix $\mathbf{A}$ and the secret $\mathbf{r}$. Consequently, it is applicable to any *structured* LWE variant [14].

Constructing the Leakage Simulator. We will now briefly discuss how such a pair $(\mathbf{f}_\mathbf{z}, f_\mathbf{z})$ can be constructed. For simplicity, assume that $\mathbf{e}$ and $\mathbf{z}$ are scalars, i.e. $\mathbf{e} = e$ and $\mathbf{z} = z$. To further simplify matters, assume first that e and e^* are continuous Gaussians instead of *discrete* Gaussians. In this perspective, $(e, ez + e^*)$ is a pair of *correlated Gaussians*, i.e. a 2-dimensional Gaussian with (possibly) non-diagonal covariance matrix. If $e \sim D_\sigma$ and $e^* \sim D_{\sigma^*}$[7], then a routine calculation shows that the covariance matrix $\mathbf{C}$ of $(e, ez + e^*)$ is

$$\mathbf{C} = \begin{pmatrix} \sigma^2 & \sigma^2 z \\ \sigma^2 z & \sigma^2 z^2 + {\sigma^*}^2 \end{pmatrix}.$$

Our idea now is, basically speaking, to find an alternative way to represent this distribution. Specifically, we want to alternatively compute $(e, ez + e^*)$ via $(\hat{e} + we^\dagger, e^\dagger)$, where $\hat{e} \sim D_{\hat{\sigma}}$ and $e^\dagger \sim D_{\sigma^\dagger}$ are independent Gaussians and w is fixed (depending on σ, σ^* and z). Again, a routine calculation finds that the covariance matrix $\mathbf{C}'$ of $(\hat{e} + we^\dagger, e^\dagger)$ is

$$\mathbf{C}' = \begin{pmatrix} \hat{\sigma}^2 + {\sigma^\dagger}^2 w^2 & {\sigma^\dagger}^2 w \\ {\sigma^\dagger}^2 w & {\sigma^\dagger}^2 \end{pmatrix}.$$

[7] We denote the continuous Gaussian distribution with *parameter* σ by D_σ, i.e. the probability density function of D_σ is proportional to $e^{-\pi \frac{x^2}{\sigma^2}}$

Now, two centered multivariate Gaussians are identically distributed, if and only if they have the same covariance matrix. Consequently, setting $\mathbf{C} = \mathbf{C}'$ and solving for $\hat{\sigma}^2, {\sigma^\dagger}^2$ and w yields

$$\begin{aligned} {\sigma^\dagger}^2 &= \sigma^2 z^2 + {\sigma^*}^2, \\ w &= \frac{\sigma^2 z}{{\sigma^\dagger}^2} = \frac{\sigma^2 z}{\sigma^2 z^2 + {\sigma^*}^2}, \\ \hat{\sigma}^2 &= \sigma^2 - {\sigma^\dagger}^2 w^2 = \sigma^2 - \frac{\sigma^4 z^2}{\sigma^2 z^2 + {\sigma^*}^2} = \left(1 - \frac{1}{1 + \frac{{\sigma^*}^2}{\sigma^2 z^2}}\right)\sigma^2. \end{aligned} \tag{1}$$

That is, for these parameters of $\hat{\sigma}, \sigma^\dagger$ and w it holds that $(e, ez + e^*) \equiv (\hat{e} + we^\dagger, e^\dagger)$, i.e. the two pairs are identically distributed. Thus, we can define $(\mathbf{f}_z, f_z)$ by $\mathbf{f}_z = we^\dagger$ and $f_z = e^\dagger$.

Now, recall that in our reduction $\hat{e}$ corresponds to the error-term in the underlying LWE-instance. Thus, we should choose σ^* so as to ensure that $\hat{e} \sim D_{\hat{\sigma}}$ is a sufficiently wide Gaussian, while σ^* should not be too large. A reasonable choice for σ^* (which simplifies calculations) is to choose it such that $\sigma^* \geq \sigma \cdot \beta$, where β is an upper bound for $|z|$ (recall that z is adversarially chosen but short). For this choice of σ^*, it holds by (1) that $\hat{\sigma} \geq \sigma/\sqrt{2}$. In other words, for this parameter choice σ^* is only a factor β bigger than σ, whereas $\hat{\sigma}$ is only a factor $1/\sqrt{2}$ smaller than σ. In essence, this means that the reduction roughly preserves the LWE parameters, up to small factors.

The final piece of our reduction is to make this leakage simulator work for discrete Gaussians instead of continuous Gaussians. For this, we will make use of Peikert's randomized rounding approach [39]. That is, a discrete Gaussian can be computed as the randomized rounding of a continuous Gaussian. This, together with Regev's discrete-to-continuous Gaussian smoothing lemma [41], allows us to adapt the simulator for continous Gaussians to discrete Gaussians. While the simplified analysis above only uses simple arithmetic, the actual analysis in the full version, while similar in spirit, relies on more involved concepts from singular value analysis to deal with high-dimensional multivariate Gaussians.

2.2 Applications

Laconic OT. As a warm-up application, it is easy to see that laconic encryption immediately implies laconic OT. Alice can construct a binary tree of keys with the following procedure: For each index pair $(2\mathsf{ind}, 2\mathsf{ind} - 1)$, Alice inserts in the tree a uniformly sampled public key either in the even position if $D_{\mathsf{ind}} = 0$, or in the odd position if $D_{\mathsf{ind}} = 1$. Bob can then simply encrypt msg_0 with respect to the index $2\mathsf{ind}$ and msg_1 with respect to index $2\mathsf{ind}-1$. Since Alice is semi-honest, the security of laconic encryption immediately carries over.

Registration-Based Encryption. Laconic Encryption almost implies RBE: Each user ind generates a key-pair and sends her $\mathsf{pk}_{\mathsf{ind}}$ for registration to an (untrusted) Key Curator, which is added to the database $D \leftarrow D \cup \{\mathsf{pk}_i\}$. Then

the digest d (the root of the tree) and the witnesses wit_j of all users are updated accordingly. Encryption and decryption with respect to ind work exactly as in laconic encryption. The crucial caveat is that in RBE, being highly dynamic, it's unrealistic to consider that the users are receiving an updated wit each time a new user registers. Therefore, there is an additional strict efficiency requirement: No user's witness should change more than $\log N$ times throughout the lifetime of the system (N being the total number of users). This requirement minimizes the interaction between a user and the key curator.

Garg et.al [24] achieve this requirement by providing a direct construction based on Merkle trees. In a nutshell, to accumulate the public keys, there are multiple Merkle trees with an increasing number of leaves. A new public key enters a (degenerate) tree that consists of a single leaf. Then, as soon as the number of its leaves is the same with the next tree, the two trees are merged. This means that a tree (and therefore its corresponding paths-witnesses) is changing only when its leaves are doubled. Overall, this translates to $\log N$ number of trees and thus at most $\log N$ number of updates per user's witness. We generalize this idea and show a generic transformation from any laconic encryption scheme to a registration-based encryption scheme. A more detailed overview and a formal description can be found in the full version.

Laconic PSI. We present a semi-honestly secure laconic PSI from laconic encryption. Here the receiver who owns a large database chooses a message for the sender to encrypt, and then it checks whether the ciphertext can be decrypted correctly with respect to the indices registered on the receiver's side.

We first need to have a hash function $\mathsf{H} : \{0,1\}^* \mapsto \{0,1\}^\ell$ to map elements into the universe of indices. For simplicity, we assume the sender's set is a singleton set $S_\mathsf{S} = \{y\}$. Besides sampling a hash function H, the setup phase is the same as the laconic encryption. Then the receiver constructs a binary tree with the freshly generated public keys with respect to the indices where the elements in S_R are mapped. In the meantime, the receiver generates the witnesses. Then the receiver sends the updated st and a random message msg. Next, the sender encrypts msg with st with respect to the index $\mathsf{H}(y)$, and sends the ciphertext ctxt to the receiver. Finally, upon receiving the ciphertext, the receiver will check for all $x_k \in S_\mathsf{R}$, whether it holds that $\mathsf{Dec}(\mathsf{sk}_k, \mathsf{wit}_k, \mathsf{ctxt}) = \mathsf{msg}$. If it finds such a k, x_k will be output as the intersection of S_S and S_R. The actual protocol will be obtained by running the above for every element in the senders set. Correctness and security of this protocol follows from the guarantees of the laconic encryption scheme. For more details, we refer the reader to the full version.

Identity-Based Encryption. We also show that our laconic encryption scheme can be modified to construct an IBE. The basic idea is simple: Instead of constructing a tree of public keys iteratively, the key authority *implicitly* defines an exponentially large tree by sampling the root of the tree at random. The key difference is that now the authority must choose the matrices in the public parameters with a trapdoor. This way, when the user ind wants to register to the system, the authority can provide it with the appropriate root-to-leaf path

(which will function as the secret key) by sampling pre-images, starting from the root and all the way down to the corresponding leaf.

Compared with other LWE-based constructions [16,1,13,21,18], our IBE supports an *unbounded* identity space, and has a tight security reduction of *full (adaptive)* security in the standard model. This is achieved with a new simulation strategy that relies on two alternating pairs of matrices $(\mathbf{B}_{0,\mathsf{even}}, \mathbf{B}_{1,\mathsf{even}})$ and $(\mathbf{B}_{0,\mathsf{odd}}, \mathbf{B}_{1,\mathsf{odd}})$, for left and right children and for even and odd layers, respectively. In the security proof, the simulator can "forget" the trapdoor of any one of the four matrices, and it can still issue decryption keys using the remaining trapdoors. This way, one can substitute ciphertext components one-by-one with uniformly sampled vectors. Proceeding until the last layer completes the security proof. A more detailed overview can be found in the full version.

Pre-Processing and Other Extensions. To increase the efficiency of our laconic encryption even further, we also construct a pre-processing variant of our scheme. Informally, the encryption algorithm Enc is split into an offline part ($\mathsf{OfflineEnc}$), which is input-independent, and an online part ($\mathsf{OnlineEnc}$). Crucially, the online algorithm is much more efficient and does not perform any public-key operation. The main observation is that each element of the ciphertext $\mathbf{c}_i$ depends only on *a single bit* of the corresponding index/identity. Thus, we can let the $\mathsf{OfflineEnc}$ algorithm computing both possible ciphertexts for each bit of the index (making sure to use the randomness consistently), and output two commitments. The $\mathsf{OnlineEnc}$ algorithm is on the other hand given the index ind, so it can complete the encryption by simply revealing the openings of the commitments corresponding to $(\mathsf{ind}_1, \ldots, \mathsf{ind}_\ell)$. As for the message, the $\mathsf{OfflineEnc}$ algorithm can simply encrypt a random bit r, and when the message msg is given to the $\mathsf{OnlineEnc}$ algorithm, it can simply output $\mathsf{msg} \oplus r$. This way, the $\mathsf{OnlineEnc}$ is entirely combinatorial, and all the public-key operation happen in an offline and input-independent phase.

We also explore a number of other extensions of laconic encryption: We describe how we can make the encryption algorithm compatible with other encryption schemes (possibly not even lattice-based), and we present an alternative laconic encryption construction that offers different efficiency trade-offs. We refer the reader to the full version for more details.

3 Preliminaries

Let $(n, p, q) = (n, p, q)(\lambda)$ with $p < q$. Let $m := n \cdot \lceil \log_p q \rceil$. Define the (p, q)-ary gadget matrix

$$\mathbf{G} := \mathbf{I}_n \otimes \left(1\ p \ldots p^{\lfloor \log_p q \rfloor}\right)$$

and denote the (balanced) p-ary decomposition by $\mathbf{G}^{-1}(\cdot)$. For a bit $b \in \{0, 1\}$, denote $\bar{b} := 1 - b$.

3.1 Lattices

Let $\mathcal{K} = \mathbb{Q}(\zeta)$ be a cyclotomic field and $\mathcal{R} = \mathbb{Z}[\zeta]$ its ring of integers, where $\zeta \in \mathbb{C}$ is a root of unity. Write $d_{\mathcal{R}}$ for the degree of (the cyclotomic polynomial defining $\mathcal{K}$ and) $\mathcal{R}$. The (infinity) norm $\|\cdot\|$ of an element $a = \sum_{i=0}^{d_{\mathcal{R}}-1} a_i \zeta^i \in \mathcal{R}$ is defined as the norm of its coefficient vector $(a_0, \ldots, a_{d_{\mathcal{R}}-1}) \in \mathbb{Z}^{d_{\mathcal{R}}}$, i.e. $\|a\| = \max_{i=0}^{d_{\mathcal{R}}-1} |a_i|$. For a vector $\mathbf{x} = (x_0, \ldots, x_{m-1}) \in \mathcal{R}^m$, its norm is defined as $\|\mathbf{x}\| \coloneqq \max_{i=0}^{m-1} \|x_i\|$. For $q \in \mathbb{N}$, write $\mathcal{R}_q \coloneqq \mathcal{R}/q\mathcal{R}$. Let χ be a distribution over $\mathcal{R}$.

Definition 1 ($\mathsf{LWE}_{\mathcal{R},n,q,\chi}$ Assumption). *Let $\mathcal{R}, n, m, q, \chi$ be parametrised by λ. The (decision) $\mathsf{LWE}_{\mathcal{R},n,m,q,\chi}$ assumption states that for any PPT adversary $\mathcal{A}$*

$$\left| \Pr\left[\mathcal{A}(\mathbf{A}, \mathbf{b}) = 1 \;\middle|\; \begin{array}{l} \mathbf{A} \leftarrow\!\$\; \mathcal{R}_q^{n \times m} \\ \mathbf{s} \leftarrow\!\$\; \mathcal{R}_q^n \\ \mathbf{e} \leftarrow\!\$\; \chi^m \\ \mathbf{b}^\mathsf{T} = \mathbf{s}^\mathsf{T} \cdot \mathbf{A} + \mathbf{e}^\mathsf{T} \bmod q \end{array} \right] - \Pr\left[\mathcal{A}(\mathbf{A}, \mathbf{b}) = 1 \;\middle|\; \begin{array}{l} \mathbf{A} \leftarrow\!\$\; \mathcal{R}_q^{n \times m} \\ \mathbf{b} \leftarrow\!\$\; \mathcal{R}_q^m \end{array} \right] \right| \leq \mathsf{negl}(\lambda).$$

The $\mathsf{LWE}_{\mathcal{R},n,q,\chi}$ assumption is said to hold if the $\mathsf{LWE}_{\mathcal{R},n,m,q,\chi}$ assumption holds for all $m = \mathsf{poly}(\lambda)$.

Definition 2 (Discrete Gaussian Distributions). *Let $m \in \mathbb{N}$ and $s > 0$. The discrete Gaussian function over $\mathbb{R}$ with parameter s is defined as $\rho_s(x) \coloneqq \exp\left(-\pi \frac{|x|^2}{s^2}\right)$ with support $\mathbb{R}$. The discrete Gaussian distribution over $\mathbb{Z}$ with parameter s is defined as $\mathcal{D}_{\mathbb{Z},s}(x) \coloneqq \frac{\rho_s(x)}{\sum_{x' \in \mathbb{Z}} \rho_s(x')}$ with support $\mathbb{Z}$. The discrete Gaussian distribution over $\mathcal{R}$ with parameter s, denoted by $\mathcal{D}_{\mathcal{R},s}$ is induced by sampling $d_{\mathcal{R}}$ independent samples $x_i \leftarrow\!\$\; \mathcal{D}_{\mathbb{Z},s}$ and outputing $x = \sum_{i=0}^{d_{\mathcal{R}}-1} x_i \cdot \zeta^i$.*

We recall a version of the leftover hash lemma over cyclotomic rings.

Lemma 1 (Adapted from [11, Lemma 7]). *Let $n = \mathsf{poly}(\lambda)$, $p, q \in \mathbb{N}$, and $m \geq n \cdot \log_p q + \omega(\log \lambda)$. The following distributions are statistically close in λ:*

$$\left\{ (\mathbf{B}, \mathbf{y}) : \begin{array}{c} \mathbf{B} \leftarrow\!\$\; \mathcal{R}_q^{n \times m} \\ \mathbf{x} \leftarrow\!\$\; \mathcal{R}_p^m \\ \mathbf{y} \coloneqq \mathbf{B} \cdot \mathbf{x} \bmod q \end{array} \right\} \quad \textit{and} \quad \left\{ (\mathbf{B}, \mathbf{y}) : \begin{array}{c} \mathbf{B} \leftarrow\!\$\; \mathcal{R}_q^{n \times m} \\ \mathbf{y} \leftarrow\!\$\; \mathcal{R}_p^n \end{array} \right\}.$$

Lemma 2 (Derived from [37, Section 2.4]). *For any $k > 0$,*

$$\Pr[\|u\| > k \cdot s \mid u \leftarrow\!\$\; \mathcal{D}_{\mathcal{R},s}] < 2 \cdot d_{\mathcal{R}} \cdot \exp(-\pi \cdot k^2).$$

Definition 3 (Ring Expansion Factor). *The expansion factor of $\mathcal{R}$, denoted by $\gamma_{\mathcal{R}}$, is $\gamma_{\mathcal{R}} \coloneqq \max_{a,b \in \mathcal{R} \setminus \{0\}} \frac{\|a \cdot b\|}{\|a\| \cdot \|b\|}$.*

Proposition 1 ([4]). *If $\mathcal{R}$ is a prime-power cyclotomic ring, then $\gamma_{\mathcal{R}} \leq 2 \deg_{\mathcal{R}}$. If $\mathcal{R}$ is a power-of-2 cyclotomic ring, then $\gamma_{\mathcal{R}} \leq \deg_{\mathcal{R}}$.*

4 Laconic Encryption

4.1 Definition

Definition 4 (Laconic Encryption). *A laconic encryption scheme for message space $\mathcal{M}$ consists of a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{KGen}, \mathsf{Upd}, \mathsf{Enc}, \mathsf{WGen}, \mathsf{Dec})$ *with the following syntax:*

- $(\mathsf{pp}, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$*: The setup algorithm is a randomized algorithm which takes as input the security parameter 1^λ and a length parameter 1^ℓ. It generates the public parameters* pp*, a state* st*, and some auxiliary information* aux*.*
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$*: The key generation algorithm takes as input the public parameters* pp *and outputs a pair of public and secret keys* $(\mathsf{pk}, \mathsf{sk})$*.*
- $\mathsf{st}' \leftarrow \mathsf{Upd}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{pk})$*: The membership update algorithm, with (read-and-write-)random access to the auxiliary information* aux*, takes as input the public parameters* pp*, the state* st*, an index* $\mathsf{ind} \in \{0,1\}^\ell$*, and a public key* pk *(or $\perp$). It outputs updated state* st'*.*
- $\mathsf{ctxt} \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{msg})$*: The encryption algorithm is a randomized algorithm which takes as input the public parameters* pp*, the state* st*, an index* $\mathsf{ind} \in \{0,1\}^\ell$*, and a message* $\mathsf{msg} \in \mathcal{M}$*. It outputs a ciphertext* ctxt*.*
- $\mathsf{wit} \leftarrow \mathsf{WGen}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{pk})$*: The witness generation algorithm, with (read-)random access to the auxiliary information* aux*, takes as input the public parameters* pp*, the state* st*, an index* $\mathsf{ind} \in \{0,1\}^\ell$*, and a public key* pk*. It outputs a (non)-membership witness* wit*.*
- $\mathsf{msg} \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{wit}, \mathsf{ctxt})$*: The decryption algorithm takes as input a secret key* sk*, a membership witness* wit*, and a ciphertext* ctxt*. It outputs a message* msg*.*

Furthermore, there exists $t \in \mathsf{poly}(\lambda, \ell)$ such that all above algorithms run in time at most $t(\lambda, \ell)$.

Our correctness definition considers a scenario where the public parameters have underdone an arbitrary sequence of updates such that in the latest version a tuple $(\mathsf{ind}, \mathsf{pk})$ is registered. In this case, if a message is encrypted with respect to $(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{pk})$, then decrypting the ciphertext with the secret key sk corresponding to pk recovers the message with overwhelming probability.

Definition 5 (Correctness). *A laconic encryption scheme Π is said to be statistically correct if for any (unbounded) algorithm $\mathcal{A}$, any $\ell = \mathsf{poly}(\lambda)$, it holds that*

$$\Pr\left[\mathsf{Correctness}_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell) = 1\right] \geq 1 - \mathsf{negl}(\lambda)$$

where the experiment $\mathsf{Correctness}_{\Pi,\mathcal{A}}$ is defined in Fig. 1.

Our security definition combines both index and message-hiding. It requires that if each of two adversarially chosen indices $\mathsf{ind}_0, \mathsf{ind}_1$ is either registered by an honest party (so that the secret key is unknown to the adversary) or not

$\mathsf{Correctness}_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell)$

$\mathsf{Honest} \coloneqq$ Empty dictionary
$(\mathsf{pp}, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$
$(\mathsf{ind}^*, \mathsf{msg}^*) \leftarrow \mathcal{A}^{\mathsf{KGen}\mathcal{O}, \mathsf{Upd}\mathcal{O}}(\mathsf{pp}, \mathsf{st}, \mathsf{aux})$
if $\mathsf{ind}^* \notin \mathsf{Honest}$ **then return** 1
$\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}^*, \mathsf{msg}^*)$
$(\mathsf{pk}^*, \mathsf{sk}^*) \leftarrow \mathsf{Honest}[\mathsf{ind}^*]$
$\mathsf{wit}^* \leftarrow \mathsf{WGen}(\mathsf{pp}, \mathsf{st}, \mathsf{aux}, \mathsf{ind}^*, \mathsf{pk}^*)$
$\mathsf{msg} = \mathsf{Dec}(\mathsf{sk}^*, \mathsf{wit}^*, \mathsf{ctxt}^*)$
return $(\mathsf{msg} = \mathsf{msg}^*)$

$\mathsf{KGen}\mathcal{O}(\mathsf{ind})$

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(\mathsf{pp})$
$\mathsf{st} \leftarrow \mathsf{Upd}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{pk})$
$\mathsf{Honest}[\mathsf{ind}] \coloneqq (\mathsf{pk}, \mathsf{sk})$
return $(\mathsf{st}, \mathsf{aux}, \mathsf{pk})$

$\mathsf{Upd}\mathcal{O}(\mathsf{ind}, \mathsf{pk})$

$\mathsf{pp} \leftarrow \mathsf{Upd}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{pk})$
if $\mathsf{pk} = \bot$ **then** $\mathsf{Honest}[\mathsf{ind}] \coloneqq \bot$
else $\mathsf{Malicious} \coloneqq \mathsf{Malicious} \cup \{ \mathsf{ind} \}$
return $(\mathsf{pp}, \mathsf{aux})$

$\mathsf{Security}^b_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell)$

$\mathsf{Malicious} \coloneqq \emptyset$
$(\mathsf{pp}, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$
$(M_0, M_1) \leftarrow \mathcal{A}^{\mathsf{KGen}\mathcal{O}, \mathsf{Upd}\mathcal{O}}(\mathsf{pp}, \mathsf{st}, \mathsf{aux})$
$M_0 = (\mathsf{ind}_0, \mathsf{msg}_0), M_1 = (\mathsf{ind}_1, \mathsf{msg}_1)$
if $\{ \mathsf{ind}_0, \mathsf{ind}_1 \} \cap \mathsf{Malicious} \neq \emptyset$ **then**
  return 0
$\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}_b, \mathsf{msg}_b)$
$b' \leftarrow \mathcal{A}^{\mathsf{KGen}\mathcal{O}, \mathsf{Upd}\mathcal{O}}(\mathsf{ctxt}^*)$
return b'

$\mathsf{Pseudorandomness}^b_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell)$

$\mathsf{Malicious} \coloneqq \emptyset$
$(\mathsf{pp}, \mathsf{st}, \mathsf{aux}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$
$(\mathsf{ind}^*, \mathsf{msg}^*) \leftarrow \mathcal{A}^{\mathsf{KGen}\mathcal{O}, \mathsf{Upd}\mathcal{O}}(\mathsf{pp}, \mathsf{st}, \mathsf{aux})$
if $\mathsf{ind}^* \in \mathsf{Malicious}$ **then return** 0
if $b = 0$ **then**
  $\mathsf{ctxt}^* \leftarrow \mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}^*, \mathsf{msg}^*)$
else $\mathsf{ctxt}^* \leftarrow\!\!\$\ \mathcal{C}$
$b' \leftarrow \mathcal{A}^{\mathsf{KGen}\mathcal{O}, \mathsf{Upd}\mathcal{O}}(\mathsf{ctxt}^*)$
return b'

Fig. 1. Correctness, security, pseudorandomness and update privacy experiments for laconic encryption.

registered, then for any adversarially chosen messages $\mathsf{msg}_0, \mathsf{msg}_1$ the adversary should not be able to distinguish a ciphertext encrypting msg_0 with respect to ind_0 from that encrypting msg_1 with respect to ind_1.

Definition 6 (Security). *A laconic encryption scheme Π is said to be secure if for any PPT (stateful) adversary $\mathcal{A}$, any $\ell = \mathsf{poly}(\lambda)$, it holds that*

$$\left|\Pr\left[\mathsf{Security}^0_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell) = 1\right] - \Pr\left[\mathsf{Security}^1_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell) = 1\right]\right| \leq \mathsf{negl}(\lambda)$$

where the experiment $\mathsf{Security}^b_{\Pi,\mathcal{A}}$ is defined in Fig. 1.

We will further define a slightly stronger security notion called *pseudorandom ciphertexts*. In essence, this property guarantees that if an index ind^* has not been registered, then a ciphertext with respect to ind^* looks pseudorandom.

Definition 7 (Pseudorandom Ciphertexts). *A laconic encryption scheme Π with ciphertext space $\mathcal{C}$ is said to be have* pseudorandom ciphertexts *if for any*

PPT (stateful) adversary $\mathcal{A}$, any $\ell = \mathsf{poly}(\lambda)$, it holds that

$$\left|\Pr\left[\mathsf{Pseudorandomness}^0_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell) = 1\right] - \Pr\left[\mathsf{Pseudorandomness}^1_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell) = 1\right]\right| \leq \mathsf{negl}(\lambda)$$

where the experiment $\mathsf{Pseudorandomness}^b_{\Pi,\mathcal{A}}$ is defined in Fig. 1.

We also define a security notion called *update privacy*. It requires that the state st hides the indices where the public keys have been registered.

Definition 8 (Update Privacy). *A laconic encryption scheme Π is said to be updated private if the distribution of the state st is (statistically close to) independent of the update operations $\mathsf{Upd}^{\mathsf{aux}}$.*

4.2 Our Construction

We construct a laconic encryption scheme for the message space $\mathcal{M} = \mathcal{R}_2$ in Fig. 2.

Theorem 1. *Let $\mathcal{R}, \ell, m, p, q, s, t$ be such that $s < t$, $\chi = \mathcal{D}_{\mathcal{R},s}$, $\bar{\chi} = \mathcal{D}_{\mathcal{R},t}$, and $q > ((2\ell+1) \cdot m \cdot \gamma_{\mathcal{R}} \cdot p + 4) \cdot \sqrt{\lambda} \cdot t + 1$. The construction in Fig. 2 is correct with overwhelming probability in λ.*

Proof. Observe that decryption is correct whenever $\left| e - \mathbf{e}^{\mathsf{T}} \cdot \begin{pmatrix} \mathbf{u}_{[\mathsf{ind}]} \\ \mathbf{x}_{\mathsf{ind}} \end{pmatrix} \right| < (q-1)/4$. By Lemma 2, with overwhelming probability in λ, we have $\|e\| \leq \frac{\sqrt{\lambda}}{2} \cdot t$ and $\|\mathbf{e}\| \leq \frac{\sqrt{\lambda}}{2} \cdot s < \frac{\sqrt{\lambda}}{2} \cdot t$. Since $\begin{pmatrix} \mathbf{u}_{[\mathsf{ind}]} \\ \mathbf{x}_{\mathsf{ind}} \end{pmatrix} \in \mathcal{R}_p^{(2\ell+1)m}$, we have $\left\| \begin{pmatrix} \mathbf{u}_{[\mathsf{ind}]} \\ \mathbf{x}_{\mathsf{ind}} \end{pmatrix} \right\| \leq p/2$. Combining these facts yields

$$\left\| e - \mathbf{e}^{\mathsf{T}} \cdot \begin{pmatrix} \mathbf{u}_{[\mathsf{ind}]} \\ \mathbf{x}_{\mathsf{ind}} \end{pmatrix} \right\| \leq (2\ell+1) \cdot m \cdot \gamma_{\mathcal{R}} \cdot \frac{\sqrt{\lambda}}{2} \cdot t \cdot \frac{p}{2} + \sqrt{\lambda} \cdot t < (q-1)/4$$

with overwhelming probability in λ. □

Theorem 2. *If $d_{\mathcal{R}} \geq \lambda$, $m \geq n \cdot \log_p q + \omega(\log \lambda)$, and the $\mathsf{LWE}_{\mathcal{R},n,q,\chi}$ assumption holds, the laconic encryption in Fig. 2 is secure. More specifically, for every PPT adversary $\mathcal{A}$ against the pseudorandom ciphertext security of the construction in Fig. 2, there exist PPT adversaries $\mathcal{A}_1$ against $\mathsf{elLWE}_{\mathcal{R},n,m,1,q,\chi,\bar{\chi},p/2}$, $\mathcal{A}_2$ against $\mathsf{LWE}_{\mathcal{R},n,2m,q,\chi}$ and $\mathcal{A}_3$ against $\mathsf{LWE}_{\mathcal{R},n,m+1,q,\chi}$ such that*

$$\mathsf{adv}(\mathcal{A}) \geq \ell \cdot \mathsf{adv}(\mathcal{A}_1) + \mathsf{adv}(\mathcal{A}_2) + \ell \cdot \mathsf{adv}(\mathcal{A}_3) + \mathsf{lhl}(\lambda)$$

where lhl is the statistical distance defined by Lemma 1.

Proof. Denote the construction by Π and write $\mathcal{C} := \mathcal{R}_q^{(2\ell+1)m+1}$ for the ciphertext space. Before we discuss the hybrids, we will briefly analyze the structure of the challenge ciphertext. In the following, let $\mathsf{ind}^* = (\mathsf{ind}_1^*, \ldots, \mathsf{ind}_\ell^*)$, and let k

$\mathsf{Setup}(1^\lambda)$

$\mathbf{A}_0, \mathbf{A}_1, \mathbf{B} \leftarrow\!\$\ \mathcal{R}_q^{n\times m}$
$\mathbf{y}_\epsilon := \mathbf{y}^* \leftarrow\!\$\ \mathcal{R}_q^n$
$\mathcal{T} := \{\,\epsilon\,\}$
$\mathsf{pp} := (\mathbf{A}_0, \mathbf{A}_1, \mathbf{B}, \mathbf{y}^*)$
$\mathsf{st} := \mathbf{y}_\epsilon$
$\mathsf{aux} := (\mathcal{T}, \{\,\mathbf{y}_v\,\}_{v\in\mathcal{T}})$
return $(\mathsf{pp}, \mathsf{st}, \mathsf{aux})$

$\mathsf{KGen}(\mathsf{pp})$

$\mathbf{x} \leftarrow\!\$\ \mathcal{R}_p^m$
$\mathbf{y} := \mathbf{B}\cdot\mathbf{x} \bmod q$
return $(\mathsf{pk}, \mathsf{sk}) := (\mathbf{y}, \mathbf{x})$

$\mathsf{Upd}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{pk})$

if $\mathsf{pk} = \perp$ **then**
  $\mathcal{T} := \mathcal{T} \setminus \{\,\mathsf{ind}\,\}$
else
  $\mathcal{T} := \mathcal{T} \cup \{\,\mathsf{ind}\,\}$
  $\mathbf{y}_{\mathsf{ind}} := \mathsf{pk}$
$\mathsf{st}' \leftarrow \mathsf{TreeUpdate}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind})$
return st'

$\mathsf{Enc}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{msg})$

$\mathbf{r}_j \leftarrow\!\$\ \mathcal{R}_q^n,\ \forall j \in \{\,0,\ldots,\ell\,\}$
for $j = 0,\ldots,\ell-1$ **do**
  $\mathbf{e}_j \leftarrow\!\$\ \chi^{2m}$
  $\mathbf{B}_j := \begin{pmatrix} \mathbf{A}_0 & \mathbf{A}_1 \\ \bar{\mathsf{ind}}_{j+1}\cdot\mathbf{G} & \mathsf{ind}_{j+1}\cdot\mathbf{G}\end{pmatrix}$
  $\mathbf{c}_j^\intercal := (\mathbf{r}_j^\intercal, \mathbf{r}_{j+1}^\intercal)\cdot\mathbf{B}_j + \mathbf{e}_j^\intercal \bmod q$
$\mathbf{e}_\ell \leftarrow\!\$\ \chi^m,\ e \leftarrow\!\$\ \bar{\chi}$
$\mathbf{c}_\ell^\intercal := \mathbf{r}_\ell^\intercal\cdot\mathbf{B} + \mathbf{e}_\ell^\intercal \bmod q$
$d := \mathbf{r}_0^\intercal\cdot\mathbf{y}_\epsilon + e + \left\lfloor\frac{q}{2}\right\rfloor\cdot\mathsf{msg} \bmod q$
return $\mathsf{ctxt} := (\mathbf{c}_0,\ldots,\mathbf{c}_\ell, d)$

$\mathsf{WGen}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind}, \mathsf{pk})$

for $j = \ell-1,\ldots,0$ **do**
  $\mathbf{u}_{\mathsf{ind}_{1:j}\|0} := -\mathbf{G}^{-1}(\mathbf{y}_{\mathsf{ind}_{1:j}\|0})$
  $\mathbf{u}_{\mathsf{ind}_{1:j}\|1} := -\mathbf{G}^{-1}(\mathbf{y}_{\mathsf{ind}_{1:j}\|1})$
return $\mathsf{wit} := (\mathbf{u}_{\mathsf{ind}_{1:j}\|0}, \mathbf{u}_{\mathsf{ind}_{1:j}\|1})_{j=0}^{\ell-1}$

$\mathsf{Dec}(\mathsf{sk}, \mathsf{wit}, \mathsf{ctxt})$

parse sk **as** $\mathbf{x}_{\mathsf{ind}}$
$\bar{\mu} := d - \sum_{j=0}^{\ell-1}\mathbf{c}_j^\intercal\cdot\begin{pmatrix}\mathbf{u}_{\mathsf{ind}_{1:j}\|0}\\ \mathbf{u}_{\mathsf{ind}_{1:j}\|1}\end{pmatrix} - \mathbf{c}_\ell^\intercal\cdot\mathbf{x}_{\mathsf{ind}} \bmod q$
if $|\bar{\mu}| < q/4$ **then return** 0
else return 1

$\mathsf{TreeUpdate}^{\mathsf{aux}}(\mathsf{pp}, \mathsf{st}, \mathsf{ind})$

for $j = \ell-1,\ldots,0$ **do**
  if $(\mathsf{ind}_{1:j}\|0) \notin \mathcal{T} \wedge (\mathsf{ind}_{1:j}\|1) \notin \mathcal{T}$ **then** // Both children of $\mathsf{ind}_{1:j}$ are unassigned.
    $\mathcal{T} := \mathcal{T} \setminus \{\,\mathsf{ind}_{1:j}\,\}$
  else
    if $(\mathsf{ind}_{1:j}\|\bar{\mathsf{ind}}_{j+1}) \notin \mathcal{T}$ **then** // $\mathsf{ind}_{1:j+1}$ is assigned but its sibling not.
      $\mathbf{y}_{\mathsf{ind}_{1:j}\|\bar{\mathsf{ind}}_{j+1}} := \mathbf{y}^*$
    $\mathbf{u}_{\mathsf{ind}_{1:j}\|0} := -\mathbf{G}^{-1}(\mathbf{y}_{\mathsf{ind}_{1:j}\|0}),\ \mathbf{u}_{\mathsf{ind}_{1:j}\|1} := -\mathbf{G}^{-1}(\mathbf{y}_{\mathsf{ind}_{1:j}\|1})$
    $\mathcal{T} := \mathcal{T} \cup \{\,\mathsf{ind}_{1:j}\,\}$ // Assign $\mathsf{ind}_{1:j}$ if any of its children is assigned.
    $\mathbf{y}_{\mathsf{ind}_{1:j}} := \mathbf{A}_0\cdot\mathbf{u}_{\mathsf{ind}_{1:j}\|0} + \mathbf{A}_1\cdot\mathbf{u}_{\mathsf{ind}_{1:j}\|1} \bmod q$
return st

Fig. 2. Construction of laconic encryption.

be such that $\mathsf{ind}^*_{1:k} = (\mathsf{ind}^*_1, \ldots, \mathsf{ind}^*_k)$ is a *leaf node* in the tree for which the adversary *does not* have a corresponding preimage. Furthermore, we denote $\mathbf{y}^*_i = \mathbf{y}_{\mathsf{ind}^*_{1:i}}$ at the nodes $\mathsf{ind}^*_1, \ldots, \mathsf{ind}^*_{1:k}$. In the following let $\mathsf{ctxt}^* = (\mathbf{c}_0, \ldots, \mathbf{c}_\ell, d)$ be the challenge ciphertext. Consider the following hybrids.

- $\mathcal{H}_0$: Identical to $\mathsf{Pseudorandomness}^0_{\Pi,\mathcal{A}}(1^\lambda, 1^\ell)$. Note that in this hybrid

$$\mathbf{c}_j^\intercal = \mathbf{r}_j^\intercal \cdot (\mathbf{A}_0 \ \mathbf{A}_1) + \mathbf{r}_{i+1}^\intercal(\bar{\mathsf{ind}}_{i+1}\mathbf{G} \ \mathsf{ind}_{i+1}\mathbf{G}) + \mathbf{e}_j^\intercal \bmod q \ \forall j \in \{0, \ldots, \ell-1\},$$
$$\mathbf{c}_\ell^\intercal = \mathbf{r}_\ell^\intercal \cdot \mathbf{B} + \mathbf{e}_\ell^\intercal \bmod q, \text{ and}$$
$$d = \mathbf{r}_0^\intercal \cdot \mathbf{y}_0^* + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q.$$

- $\mathcal{H}_1$: Compute ctxt^* as follows. Choose $\mathbf{c}_0, \ldots, \mathbf{c}_{k-1} \leftarrow\!\!\$\ \mathcal{R}_q^{2m}$ uniformly at random, choose $\mathbf{e}_0, \ldots, \mathbf{e}_{k-1} \leftarrow\!\!\$\ \chi^{2m}$, and set

$$d = \sum_{j=0}^{k-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j + \mathbf{r}_k^\intercal \cdot \mathbf{y}_k^* + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q,$$

  where $\mathbf{z}_j = \begin{pmatrix} \mathbf{u}_{\mathsf{ind}^*_{1:j}\|0} \\ \mathbf{u}_{\mathsf{ind}^*_{1:j}\|1} \end{pmatrix} \in \mathcal{R}_p^{2m}$. Furthermore, compute $\mathbf{c}_k, \ldots, \mathbf{c}_\ell$ as in $\mathcal{H}_0$.
- $\mathcal{H}_2$: In this hybrid we choose $\mathbf{c}_k \leftarrow\!\!\$\ \mathcal{R}_q^{2m}$ and $d \leftarrow\!\!\$\ \mathcal{R}_q$ uniformly at random.
- $\mathcal{H}_3$: In this hybrid we choose $\mathbf{c}_i \leftarrow\!\!\$\ \mathcal{R}_q^{2m}$ uniformly at random for $i = k+1, \ldots, \ell-1$ and $\mathbf{c}_\ell \leftarrow\!\!\$\ \mathcal{R}_q^m$ uniformly at random.

Note that in $\mathcal{H}_3$ all ciphertext components are chosen uniformly random. Hence the claim of the theorem follows. We will establish the indistinguishability of successive hybrids via a sequence of lemmata. □

Lemma 3. *For any PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{A}_1$ against $\mathsf{elLWE}_{\mathcal{R},n,m,1,q,\chi,\bar{\chi},p/2}$ such that*

$$|\mathsf{Pr}[\mathcal{H}_0(\mathcal{A}) = 1] - \mathsf{Pr}[\mathcal{H}_1(\mathcal{A}) = 1]| \leq \ell \cdot \mathsf{adv}(\mathcal{A}_1).$$

Proof. To show that $\mathcal{H}_0$ and $\mathcal{H}_1$ are computationally indistinguishable, we define the following sub-hybrids $\mathcal{H}'_0, \ldots, \mathcal{H}'_\ell$ and $\mathcal{H}''_0, \ldots, \mathcal{H}''_\ell$.

- $\mathcal{H}'_i$ (for $i = 0, \ldots, \ell$): $\mathcal{H}'_0$ is identical to $\mathcal{H}_0$ and hybrids $\mathcal{H}'_{>k}$ are identical to $\mathcal{H}_1$. For the middle cases, i.e. $1 \leq i \leq k$, we define hybrid $\mathcal{H}'_i$ so that $\mathbf{c}_0, \ldots, \mathbf{c}_{i-1}$ and d are computed as in $\mathcal{H}_1$, and $\mathbf{c}_i, \ldots, \mathbf{c}_\ell$ are computed as in $\mathcal{H}_0$. Specifically, different from $\mathcal{H}_0$, we choose $\mathbf{c}_0, \ldots, \mathbf{c}_{i-1} \leftarrow\!\!\$\ \mathcal{R}_q^{2m}$ uniformly at random, choose $\mathbf{e}_0, \ldots, \mathbf{e}_{i-1} \leftarrow\!\!\$\ \chi^{2m}$ and set

$$d = \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j + \mathbf{r}_i^\intercal \cdot \mathbf{y}_i^* + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q.$$

– $\mathcal{H}''_i$ (for $i = 1, \ldots, \ell$): If $i > k$, then this hybrid is identical to $\mathcal{H}'_i$. Else ($1 \leq i \leq k$), $\mathbf{c}_j$ for all $j \in [0 : \ell] \setminus \{ i-1 \}$ are computed as in $\mathcal{H}'_i$, and $\mathbf{c}_{i-1}$ is computed as follows. Choose $\hat{\mathbf{c}}_{i-1}$ uniformly at random and set

$$\mathbf{c}_{i-1}^{\intercal} = \hat{\mathbf{c}}_{i-1}^{\intercal} + \mathbf{r}_i^{\intercal} \cdot (\bar{\mathsf{ind}}_i \cdot \mathbf{G}\ \mathsf{ind}_i \cdot \mathbf{G}) \bmod q.$$

Furthermore, we set

$$d = \sum_{j=0}^{i-2} (\mathbf{c}_j - \mathbf{e}_j)^{\intercal} \cdot \mathbf{z}_j + (\hat{\mathbf{c}}_{i-1}^{\intercal} - \mathbf{e}_{i-1}^{\intercal}) \cdot \mathbf{z}_{i-1} + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q.$$

First, observe that $\mathcal{H}'_i$ and $\mathcal{H}''_i$ are in fact identically distributed: In $\mathcal{H}''_i$, since $\hat{\mathbf{c}}_{i-1}$ is uniformly and independently distributed, we can equivalently compute it as

$$\hat{\mathbf{c}}_{i-1}^{\intercal} = \bar{\mathbf{c}}_{i-1}^{\intercal} - \mathbf{r}_i^{\intercal} \cdot (\bar{\mathsf{ind}}_i \cdot \mathbf{G}\ \mathsf{ind}_i \cdot \mathbf{G})$$

for a uniformly random and independent $\bar{\mathbf{c}}_{i-1}$. This makes $\mathbf{c}_{i-1} = \bar{\mathbf{c}}_{i-1}$ uniformly random, as in $\mathcal{H}'_i$. Substituting the new $\hat{\mathbf{c}}_i$ to the expression of d in $\mathcal{H}''_i$, we have

$$\begin{aligned}
d &= \sum_{j=0}^{i-2} (\mathbf{c}_j - \mathbf{e}_j)^{\intercal} \cdot \mathbf{z}_j + (\hat{\mathbf{c}}_{i-1}^{\intercal} - \mathbf{e}_{i-1}^{\intercal}) \cdot \mathbf{z}_{i-1} + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q \\
&= \sum_{j=0}^{i-2} (\mathbf{c}_j - \mathbf{e}_j)^{\intercal} \mathbf{z}_j + (\mathbf{c}_{i-1}^{\intercal} - \mathbf{r}_i^{\intercal} (\bar{\mathsf{ind}}_i \mathbf{G}\ \mathsf{ind}_i \mathbf{G}) - \mathbf{e}_{i-1}^{\intercal}) \mathbf{z}_{i-1} + e + \left\lfloor \frac{q}{2} \right\rfloor \mathsf{msg} \bmod q \\
&= \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^{\intercal} \cdot \mathbf{z}_j - \mathbf{r}_i^{\intercal} \cdot (\bar{\mathsf{ind}}_i \cdot \mathbf{G}\ \mathsf{ind}_i \cdot \mathbf{G}) \cdot \mathbf{z}_{i-1} + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q \\
&= \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^{\intercal} \cdot \mathbf{z}_j + \mathbf{r}_i^{\intercal} \cdot \mathbf{y}_i^* + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q,
\end{aligned}$$

as in $\mathcal{H}'_i$, where the last equality was due to $(\bar{\mathsf{ind}}_i \cdot \mathbf{G}\ \mathsf{ind}_i \cdot \mathbf{G}) \cdot \mathbf{z}_{i-1} = -\mathbf{y}_i^*$.

The main technical part of this proof lies in establishing indistinguishability between hybrids $\mathcal{H}'_i$ and $\mathcal{H}''_{i+1}$ for $i \in \{ 0, \ldots, \ell - 1 \}$. Note that the case $k < i \leq \ell - 1$ is trivial since $\mathcal{H}''_i = \mathcal{H}'_i = \mathcal{H}_1$ for $i > k$. In the following, we focus on the remaining case $0 \leq i \leq k$. We will show that these two hybrids are indistinguishable under elLWE. Assume towards contradiction that

$$\mathsf{Pr}\left[\mathcal{H}'_i(\mathcal{A}) = 1\right] - \mathsf{Pr}\left[\mathcal{H}''_{i+1}(\mathcal{A}) = 1\right] \geq \epsilon.$$

We will show that this implies a PPT adversary $\mathcal{A}'_1$ against elLWE with advantage ϵ.

The adversary $\mathcal{A}'_1$ is specified as follows. As input it receives a matrix $\mathbf{A} \in \mathcal{R}_q^{n \times 2m}$, and it parses $\mathbf{A}$ as $\mathbf{A} = (\mathbf{A}_0\ \mathbf{A}_1)$ where $\mathbf{A}_0, \mathbf{A}_1 \in \mathcal{R}_q^{n \times m}$. Now $\mathcal{A}'_1$ simulates $\mathcal{H}'_i(\mathcal{A})$ with the matrices $\mathbf{A}_0, \mathbf{A}_1$ thus obtained, until the adversary $\mathcal{A}$ queries the challenge ciphertext. Now it chooses $\mathbf{z}^* = -\mathbf{z}_i$ and sends $\mathbf{z}^*$ to its

challenger. Note that $\mathbf{z}^*$ is a legit query as $\|\mathbf{z}^*\| \leq p/2$. Now $\mathcal{A}_1'$ obtaining a leak l and $\mathbf{y}$. Next, it computes the challenge ciphertext as in $\mathcal{H}_i'(\mathcal{A})$, except that it sets

$$\mathbf{c}_i = \mathbf{y} + \mathbf{r}_{i+1} \cdot (\bar{\mathsf{ind}}_{i+1} \cdot \mathbf{G} \ \mathsf{ind}_{i+1} \cdot \mathbf{G}) \bmod q$$

and

$$d = \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \mathbf{z}_j - \mathbf{y}^\intercal \cdot \mathbf{z}^* + l + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q.$$

Note that the remaining ciphertext components are the same as in $\mathcal{H}_i'(\mathcal{A})$ and $\mathcal{H}_{i+1}''(\mathcal{A})$. From there on, $\mathcal{A}_1'$ continues the simulation of $\mathcal{H}_i'(\mathcal{A})$ and outputs whatever $\mathcal{H}_i'(\mathcal{A})$ outputs.

Now let $b \in \{0,1\}$ be the challenge bit of the elLWE experiment. We claim that if $b = 0$, then $\mathcal{A}_1'$ faithfully simulates $\mathcal{H}_i'(\mathcal{A})$. On the other hand, we claim that for $b = 1$ the $\mathcal{A}_1'$ faithfully simulates $\mathcal{H}_{i+1}''(\mathcal{A})$. From these two claims it follows that $\mathcal{A}_1'$ has advantage ϵ.

- For $b = 0$, it holds that $\mathbf{y}^\intercal = \mathbf{r}^\intercal \cdot \mathbf{A} + \mathbf{e}^\intercal = \mathbf{r}^\intercal \cdot (\mathbf{A}_0 \ \mathbf{A}_1) + \mathbf{e}^\intercal \bmod q$ and $l = \mathbf{e}^\intercal \cdot \mathbf{z}^* + e = -\mathbf{e}^\intercal \cdot \mathbf{z}_i + e \bmod q$. Renaming $\mathbf{r}$ to $\mathbf{r}_i$ and $\mathbf{e}$ to $\mathbf{e}_i$, it holds that

$$\begin{aligned} \mathbf{c}_i^\intercal &= \mathbf{y}^\intercal + \mathbf{r}_{i+1}^\intercal \cdot (\bar{\mathsf{ind}}_{i+1} \cdot \mathbf{G} \ \mathsf{ind}_{i+1} \cdot \mathbf{G}) \bmod q \\ &= \mathbf{r}_i^\intercal \cdot (\mathbf{A}_0 \ \mathbf{A}_1) + \mathbf{r}_{i+1}^\intercal \cdot (\bar{\mathsf{ind}}_{i+1} \cdot \mathbf{G} \ \mathsf{ind}_{i+1} \cdot \mathbf{G}) + \mathbf{e}_i^\intercal \bmod q \end{aligned}$$

  and

$$\begin{aligned} d &= \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j - \mathbf{y}^\intercal \cdot \mathbf{z}^* + l + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q && (2) \\ &= \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j + (\mathbf{r}_i^\intercal \cdot \mathbf{A} + \mathbf{e}_i^\intercal) \cdot \mathbf{z}_i - \mathbf{e}_i^\intercal \mathbf{z}_i + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q && (3) \\ &= \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j + \mathbf{r}_i^\intercal \cdot \mathbf{y}_i^* + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q, && (4) \end{aligned}$$

  where the last equality holds as $\mathbf{y}_i^* = \mathbf{A}\mathbf{z}_i$. We can conclude that in this case the simulation of $\mathcal{A}_1'$ and $\mathcal{H}_i'(\mathcal{A})$ are identically distributed.
- For $b = 1$, it holds that $\mathbf{y} = \hat{\mathbf{c}}_i$ for a uniformly random $\hat{\mathbf{c}}_i \leftarrow\!\!\$\ \mathcal{R}_q^{2m}$ and $l = \mathbf{e}^\intercal \cdot \mathbf{z}^* + e = -\mathbf{e}^\intercal \mathbf{z}_i + e$. It therefore holds that

$$\begin{aligned} \mathbf{c}_i^\intercal &= \mathbf{y}^\intercal + \mathbf{r}_{i+1}^\intercal \cdot (\bar{\mathsf{ind}}_{i+1} \cdot \mathbf{G} \ \mathsf{ind}_{i+1} \cdot \mathbf{G}) \bmod q \\ &= \hat{\mathbf{c}}_i^\intercal + \mathbf{r}_{i+1}^\intercal \cdot (\bar{\mathsf{ind}}_{i+1} \cdot \mathbf{G} \ \mathsf{ind}_{i+1} \cdot \mathbf{G}) \bmod q \end{aligned}$$

  and

$$d = \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j - \mathbf{y}^\intercal \cdot \mathbf{z}^* + l + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q$$

$$= \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j + \hat{\mathbf{c}}_i^\intercal \cdot \mathbf{z}_i - \mathbf{e}_i^\intercal \mathbf{z}_i + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q$$
$$= \sum_{j=0}^{i-1} (\mathbf{c}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j + (\hat{\mathbf{c}}_i^\intercal - \mathbf{e}_i^\intercal) \cdot \mathbf{z}_i + e + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q.$$

I.e. it holds that in this case the simulation of $\mathcal{A}_1'$ and $\mathcal{H}_{i+1}''(\mathcal{A})$ are identically distributed. □

Lemma 4. *For any PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{A}_2$ against $\mathsf{LWE}_{\mathcal{R},n,2m,q,\chi}$ such that*

$$|\mathsf{Pr}\left[\mathcal{H}_1(\mathcal{A}) = 1\right] - \mathsf{Pr}\left[\mathcal{H}_2(\mathcal{A}) = 1\right]| \leq \mathsf{adv}(\mathcal{A}_2) + \mathsf{lhl}(\lambda).$$

Proof. In the following we describe the adversary $\mathcal{A}_2$ for the case where $k \neq \ell$, i.e., the challenge identity is not registered. For the case $k = \ell$, the argument is the same, except that we first invoke Lemma 1 to switch the matrix $\mathbf{B}$ to uniformly sampled, which introduces an additive (statistical) term $\mathsf{lhl}(\lambda)$ in the distance between the two hybrids.

$\mathcal{A}_2$ first queries $2m$ LWE samples from its oracle and arranges them in matrix form as $(\mathbf{A}, \mathbf{v})$, this $\mathbf{A}$ is then parsed as $\mathbf{A} = (\mathbf{A}_0\ \mathbf{A}_1)$ and uses $\mathbf{A}_0$ and $\mathbf{A}_1$ in pp, whereas the vector $\mathbf{v}$ is stored. Next, $\mathcal{A}_2$ queries m LWE samples from its oracle and arranges them in matrix form as $(\mathbf{B}, \mathbf{v}')$, this $\mathbf{B}$ is then used as part of pp. Now $\mathcal{A}_2$ simulates $\mathcal{H}_1$, but whenever a new honest key pk_i^* is generated, $\mathcal{A}_2$ queries its LWE oracle and obtains $(\hat{\mathbf{y}}_i, \hat{v}_i)$, sets $\mathsf{pk}_i = \hat{\mathbf{y}}_i$ and stores $\hat{v}_i$. The challenge ciphertext is generated as follows: Assume the challenge identity ind^* terminates in a public key pk_{i^*}. The challenge ciphertext is computed as in $\mathcal{H}_2(\mathcal{A})$, except that we set

$$d = \sum_{j=0}^{k-1} (\mathbf{u}_j - \mathbf{e}_j)^\intercal \cdot \mathbf{z}_j + \hat{v}_{i^*} + \left\lfloor \frac{q}{2} \right\rfloor \cdot \mathsf{msg} \bmod q$$

and $\mathbf{c}_k^\intercal = \mathbf{v}$ if $k < \ell$ and $\mathbf{c}_k^\intercal = \mathbf{v}'$ if $k = \ell$. $\mathcal{A}_2$ then continues simulation of $\mathcal{H}_2(\mathcal{A})$ and outputs whatever $\mathcal{H}_2(\mathcal{A})$ outputs.

First observe that if $(\mathbf{A}, \mathbf{v})$, $(\mathbf{B}, \mathbf{v}')$ and $\{(\hat{\mathbf{y}}_{i^*}, \hat{v}_{i^*})\}$ are LWE samples, i.e. $\mathbf{v} = \mathbf{s}^\intercal \cdot \mathbf{A} + \mathbf{e}^\intercal \bmod q$, $(\mathbf{v}')^\intercal = \mathbf{s}^\intercal \cdot \mathbf{B} + \hat{\mathbf{e}}^\intercal \bmod q$ and $\hat{v}_{i^*} = \mathbf{s}^\intercal \cdot \hat{\mathbf{y}}_{i^*} + \hat{e}_{i^*} \bmod q$, then the simulation of $\mathcal{A}_2$ is identically distributed to $\mathcal{H}_2(\mathcal{A})$. On the other hand, if $\mathbf{c}$, $\mathbf{v}'$ and the $\hat{v}_{i^*}$ are uniformly random and independent, then the simulation of $\mathcal{A}_2$ is identically distributed to $\mathcal{H}_3(\mathcal{A})$. The claim of the lemma follows. □

Lemma 5. *For any PPT adversary $\mathcal{A}$ there exists a PPT adversary $\mathcal{A}_3$ against $\mathsf{LWE}_{\mathcal{R},n,m+1,q,\chi}$ such that*

$$|\mathsf{Pr}\left[\mathcal{H}_2(\mathcal{A}) = 1\right] - \mathsf{Pr}\left[\mathcal{H}_3(\mathcal{A}) = 1\right]| \leq \ell \cdot \mathsf{adv}(\mathcal{A}_3).$$

Proof. Consider the following hybrids $\mathcal{H}_0''', \ldots, \mathcal{H}_\ell'''$, where $\mathcal{H}_0'''$ is identically distributed to $\mathcal{H}_2$, and $\mathcal{H}_\ell'''$ is identically distributed to $\mathcal{H}_3$.

– $\mathcal{H}'''_{i+1}$ (For $i = 0, \ldots, \ell - 1$): Identically distributed to $\mathcal{H}_i$, except that, for $i > k$, $\mathbf{c}_i$ is chosen uniformly at random.

Assume towards contradiction that

$$\mathsf{Pr}\left[\mathcal{H}'''_{i+1}(\mathcal{A}) = 1\right] - \mathsf{Pr}\left[\mathcal{H}'''_{i}(\mathcal{A}) = 1\right] \geq \epsilon$$

for some $i \in \{0, \ldots, \ell\}$. We will show that this implies a PPT adversary $\mathcal{A}_3$ against $\mathsf{LWE}_{\mathcal{R},n,m,q,\chi}$ with advantage ϵ. The adversary $\mathcal{A}_3$ receives an input $(\mathbf{A}, \mathbf{v})$ and proceeds as follows. $\mathcal{A}_3$ simulates $\mathcal{H}'''_i(\mathcal{A})$, except for the following modifications. First, it uses the matrix $\mathbf{A} = (\mathbf{A}_0\ \mathbf{A}_1)$ in the public parameters. Next, when the challenge ciphertext is generated, if $i > k$ it sets $\mathbf{c}_i = \mathbf{v}$. $\mathcal{A}_3$ then continues the simulation and outputs whatever its simulation of $\mathcal{H}'''_{i-1}(\mathcal{A})$ outputs.

Now, it follows routinely that if $(\mathbf{A}, \mathbf{v})$ is an LWE sample, i.e. $\mathbf{v}^\intercal = \mathbf{s}^\intercal \cdot \mathbf{A} + \mathbf{e}^\intercal \bmod q$, then the simulation of $\mathcal{A}_3$ is distributed identically to $\mathcal{H}'''_i(\mathcal{A})$. On the other hand, if $\mathbf{v}$ is uniformly random, then the simulation of $\mathcal{A}_3$ is distributed identically to $\mathcal{H}'''_{i+1}(\mathcal{A})$. We can conclude that $\mathcal{A}_3$ has advantage ϵ. □

Update Privacy. There is a simple modification to construction of laconic encryption in Fig. 2 which yields update-privacy. The idea is to make the hash-function $f_{\mathbf{A}_0,\mathbf{A}_1}(\mathbf{y}_0, \mathbf{y}_1) = \mathbf{A}_0 \cdot (-\mathbf{G}^{-1}(\mathbf{y}_0)) + \mathbf{A}_1 \cdot (-\mathbf{G}^{-1}(\mathbf{y}_1))$ randomized such that $f_{\mathbf{A}_0,\mathbf{A}_1}(\mathbf{y}_0, \mathbf{y}_1)$ *statistically hides* $\mathbf{y}_0$ and $\mathbf{y}_1$. This can be achieved by slightly modifying the gadget matrix $\mathbf{G}$ into $\mathbf{G}'$ and making $\mathbf{G}'^{-1}$ randomized [8], and replacing the parameter m with a slightly larger $m' = m + n\log(q)$. Specifically, we set $\mathbf{G}' = (\mathbf{G}\ \mathbf{0}) \in \mathbb{Z}^{n \times m'}$, i.e. we obtain $\mathbf{G}'$ by appending $n\log(q)$ all-zero columns to $\mathbf{G}$. Furthermore, we define $\mathbf{G}'^{-1}(\mathbf{x}) = \begin{pmatrix} \mathbf{G}^{-1}(\mathbf{x}) \\ \mathbf{r} \end{pmatrix}$, where $\mathbf{r} \leftarrow\!\!\$\ \mathcal{R}_2^{2n\log(q)}$ is chosen uniformly at random. Note that it still holds that $\mathbf{G}'\mathbf{G}'^{-1}(\mathbf{x}) = \mathbf{x}$ for all $\mathbf{x} \in \mathcal{R}_q^n$. The modified hash function is now

$$f'(\mathbf{y}_0, \mathbf{y}_1) = \mathbf{A}_0 \cdot (-\mathbf{G}'^{-1}(\mathbf{y}_0)) + \mathbf{A}_1 \cdot (-\mathbf{G}'^{-1}(\mathbf{y}_1)).$$

Now, decomposing $\mathbf{A}_0 = (\mathbf{A}_{0,1}\ \mathbf{A}_{0,2})$ and $\mathbf{A}_1 = (\mathbf{A}_{1,1}\ \mathbf{A}_{1,2})$, where $\mathbf{A}_{0,1}, \mathbf{A}_{1,1} \in \mathcal{R}_q^{n \times m}$ and $\mathbf{A}_{0,2}, \mathbf{A}_{1,2} \in \mathcal{R}_q^{n \times n\log(q)}$, it holds that

$$\begin{aligned} f'(\mathbf{y}_0, \mathbf{y}_1) &= \mathbf{A}_0 \cdot (-\mathbf{G}'^{-1}(\mathbf{y}_0)) + \mathbf{A}_1 \cdot (-\mathbf{G}'^{-1}(\mathbf{y}_1)) \\ &= (\mathbf{A}_{0,1}\ \mathbf{A}_{0,2}) \cdot \left(-\begin{pmatrix} \mathbf{G}^{-1}(\mathbf{y_0}) \\ \mathbf{r}_0 \end{pmatrix}\right) + (\mathbf{A}_{1,1}\ \mathbf{A}_{1,2}) \cdot \left(-\begin{pmatrix} \mathbf{G}^{-1}(\mathbf{y}_1) \\ \mathbf{r}_1 \end{pmatrix}\right) \\ &= (\mathbf{A}_{0,1}\ \mathbf{A}_{1,1}) \begin{pmatrix} -\mathbf{G}^{-1}(\mathbf{y_0}) \\ -\mathbf{G}^{-1}(\mathbf{y_1}) \end{pmatrix} - \underbrace{(\mathbf{A}_{0,2}\ \mathbf{A}_{1,2}) \begin{pmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \end{pmatrix}}_{=:\mathbf{v}}. \end{aligned}$$

[8] [12] defined a similar notion of *randomized* $\mathbf{G}^{-1}$*, which however samples a discrete gaussian preimage*

Since the matrix $(\mathbf{A}_{0,2}\ \mathbf{A}_{1,2}) \in \mathcal{R}_q^{n\times 2n\log(q)}$ is chosen uniformly random and $(\mathbf{r}_0, \mathbf{r}_1)$ is uniformly random in $\mathcal{R}_2^{2n\log(q)}$, it holds by the leftover hash lemma (Lemma 1) that $\mathbf{v}$ is 2^{-n}-close to uniform. Consequently, the hash-value $f'(\mathbf{y}_0, \mathbf{y}_1)$ is statistically close to uniform. Hence, update privacy of the modified construction follows. We can conclude the following lemma.

Lemma 6. *The modified construction of Fig. 2 using* $\mathbf{G}'$ *and the randomized* $\mathbf{G}'^{-1}$ *is update private.*

Properties and Efficiency. We remark about some properties and efficiency of our construction. The public parameters (initially) consists of three uniformly random matrices $\mathbf{A}_0, \mathbf{A}_1, \mathbf{B} \in \mathcal{R}_q^{n\times m}$ which can be sampled with public coin. Subsequent updates to the public parameters, more specifically to $\mathbf{y}_\epsilon \in \mathcal{R}_q^n$, are deterministic. Suppose we pick q to be linear in ℓ, then the Setup and KGen algorithms run in time logarithmic in ℓ, while the Upd, Enc, WGen, and Dec algorithms run in time quasi-linear in ℓ.

Acknowledgments

Nico Döttling and Chuanwei Lin: Funded by the European Union (ERC, LACONIC, 101041207). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them. Giulio Malavolta and Ahmadreza Rahimi: This work was partially funded by the German Federal Ministry of Education and Research (BMBF) in the course of the 6GEM research hub under grant number 16KISK038 and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2092 CASA 390781972. Dimitris Kolonelos: Received funding from projects from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under project PICOCRYPT (grant agreement No. 101001283), from the Spanish Government under project PRODIGY (TED2021-132464B-I00), and from the Madrid Regional Government under project BLOQUES (S2018/TCS-4339). The last two projects are co-funded by European Union EIE, and NextGenerationEU/PRTR funds.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (Aug 2010). 10.1007/978-3-642-14623-7_6
2. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). In: 28th ACM STOC. pp. 99–108. ACM Press (May 1996). 10.1145/237814.237838
3. Alamati, N., Branco, P., Döttling, N., Garg, S., Hajiabadi, M., Pu, S.: Laconic private set intersection and applications. In: Nissim, K., Waters, B. (eds.) TCC 2021, Part III. LNCS, vol. 13044, pp. 94–125. Springer, Heidelberg (Nov 2021). 10.1007/978-3-030-90456-2_4

4. Albrecht, M.R., Lai, R.W.F.: Subtractive sets over cyclotomic rings - limits of Schnorr-like arguments over lattices. In: Malkin, T., Peikert, C. (eds.) CRYPTO 2021, Part II. LNCS, vol. 12826, pp. 519–548. Springer, Heidelberg, Virtual Event (Aug 2021). 10.1007/978-3-030-84245-1_18
5. Alekhnovich, M.: More on average case vs approximation complexity. In: 44th FOCS. pp. 298–307. IEEE Computer Society Press (Oct 2003). 10.1109/SFCS.2003.1238204
6. Alperin-Sheriff, J., Peikert, C.: Circular and KDM security for identity-based encryption. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 334–352. Springer, Heidelberg (May 2012). 10.1007/978-3-642-30057-8_20
7. Aranha, D., Lin, C., Orlandi, C., Simkin, M.: Laconic private set-intersection from pairings. Cryptology ePrint Archive, Report 2022/529 (2022), https://eprint.iacr.org/2022/529
8. Benhamouda, F., Lin, H.: k-round multiparty computation from k-round oblivious transfer via garbled interactive circuits. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 500–532. Springer, Heidelberg (Apr / May 2018). 10.1007/978-3-319-78375-8_17
9. Boneh, D., Franklin, M.K.: Identity-based encryption from the Weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (Aug 2001). 10.1007/3-540-44647-8_13
10. Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J.M., Schwabe, P., Seiler, G., Stehlé, D.: Crystals – kyber: a cca-secure module-lattice-based kem. Cryptology ePrint Archive, Paper 2017/634 (2017). 10.1109/EuroSP.2018.00032, https://eprint.iacr.org/2017/634, https://eprint.iacr.org/2017/634
11. Boudgoust, K., Jeudy, C., Roux-Langlois, A., Wen, W.: Towards classical hardness of module-LWE: The linear rank case. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part II. LNCS, vol. 12492, pp. 289–317. Springer, Heidelberg (Dec 2020). 10.1007/978-3-030-64834-3_10
12. Bourse, F., del Pino, R., Minelli, M., Wee, H.: FHE circuit privacy almost for free. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part II. LNCS, vol. 9815, pp. 62–89. Springer, Heidelberg (Aug 2016). 10.1007/978-3-662-53008-5_3
13. Boyen, X., Li, Q.: Towards tightly secure lattice short signature and id-based encryption. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 404–434. Springer, Heidelberg (Dec 2016). 10.1007/978-3-662-53890-6_14
14. Brakerski, Z., Döttling, N.: Lossiness and entropic hardness for ring-LWE. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 1–27. Springer, Heidelberg (Nov 2020). 10.1007/978-3-030-64375-1_1
15. Brakerski, Z., Lombardi, A., Segev, G., Vaikuntanathan, V.: Anonymous IBE, leakage resilience and circular security from new assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part I. LNCS, vol. 10820, pp. 535–564. Springer, Heidelberg (Apr / May 2018). 10.1007/978-3-319-78381-9_20
16. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (May / Jun 2010). 10.1007/978-3-642-13190-5_27
17. Cho, C., Döttling, N., Garg, S., Gupta, D., Miao, P., Polychroniadou, A.: Laconic oblivious transfer and its applications. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 33–65. Springer, Heidelberg (Aug 2017). 10.1007/978-3-319-63715-0_2

18. Döttling, N., Garg, S.: From selective IBE to full IBE and selective HIBE. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part I. LNCS, vol. 10677, pp. 372–408. Springer, Heidelberg (Nov 2017). 10.1007/978-3-319-70500-2_13
19. Döttling, N., Garg, S.: Identity-based encryption from the Diffie-Hellman assumption. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 537–569. Springer, Heidelberg (Aug 2017). 10.1007/978-3-319-63688-7_18
20. Döttling, N., Garg, S., Goyal, V., Malavolta, G.: Laconic conditional disclosure of secrets and applications. In: Zuckerman, D. (ed.) 60th FOCS. pp. 661–685. IEEE Computer Society Press (Nov 2019). 10.1109/FOCS.2019.00046
21. Döttling, N., Garg, S., Hajiabadi, M., Masny, D.: New constructions of identity-based and key-dependent message secure encryption schemes. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part I. LNCS, vol. 10769, pp. 3–31. Springer, Heidelberg (Mar 2018). 10.1007/978-3-319-76578-5_1
22. Döttling, N., Garg, S., Ishai, Y., Malavolta, G., Mour, T., Ostrovsky, R.: Trapdoor hash functions and their applications. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 3–32. Springer, Heidelberg (Aug 2019). 10.1007/978-3-030-26954-8_1
23. Garg, S., Hajiabadi, M.: Trapdoor functions from the computational Diffie-Hellman assumption. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part II. LNCS, vol. 10992, pp. 362–391. Springer, Heidelberg (Aug 2018). 10.1007/978-3-319-96881-0_13
24. Garg, S., Hajiabadi, M., Mahmoody, M., Rahimi, A.: Registration-based encryption: Removing private-key generator from IBE. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part I. LNCS, vol. 11239, pp. 689–718. Springer, Heidelberg (Nov 2018). 10.1007/978-3-030-03807-6_25
25. Garg, S., Hajiabadi, M., Mahmoody, M., Rahimi, A., Sekar, S.: Registration-based encryption from standard assumptions. In: Lin, D., Sako, K. (eds.) PKC 2019, Part II. LNCS, vol. 11443, pp. 63–93. Springer, Heidelberg (Apr 2019). 10.1007/978-3-030-17259-6_3
26. Garg, S., Srinivasan, A.: Garbled protocols and two-round MPC from bilinear maps. In: Umans, C. (ed.) 58th FOCS. pp. 588–599. IEEE Computer Society Press (Oct 2017). 10.1109/FOCS.2017.60
27. Garg, S., Srinivasan, A.: Adaptively secure garbling with near optimal online complexity. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 535–565. Springer, Heidelberg (Apr / May 2018). 10.1007/978-3-319-78375-8_18
28. Garg, S., Srinivasan, A.: Two-round multiparty secure computation from minimal assumptions. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 468–499. Springer, Heidelberg (Apr / May 2018). 10.1007/978-3-319-78375-8_16
29. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Mitzenmacher, M. (ed.) 41st ACM STOC. pp. 169–178. ACM Press (May / Jun 2009). 10.1145/1536414.1536440
30. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press (May 2008). 10.1145/1374376.1374407
31. Glaeser, N., Kolonelos, D., Malavolta, G., Rahimi, A.: Efficient registration-based encryption. Cryptology ePrint Archive, Paper 2022/1505 (2022), https://eprint.iacr.org/2022/1505, https://eprint.iacr.org/2022/1505

32. Goldreich, O., Goldwasser, S., Halevi, S.: Collision-free hashing from lattice problems. Cryptology ePrint Archive, Report 1996/009 (1996), https://eprint.iacr.org/1996/009
33. Goyal, R., Vusirikala, S.: Verifiable registration-based encryption. In: Micciancio, D., Ristenpart, T. (eds.) CRYPTO 2020, Part I. LNCS, vol. 12170, pp. 621–651. Springer, Heidelberg (Aug 2020). 10.1007/978-3-030-56784-2_21
34. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM J. Comput. **28**(4), 1364–1396 (1999)
35. Hohenberger, S., Lu, G., Waters, B., Wu, D.J.: Registered attribute-based encryption. Cryptology ePrint Archive, Paper 2022/1500 (2022), https://eprint.iacr.org/2022/1500, https://eprint.iacr.org/2022/1500
36. Libert, B., Ling, S., Nguyen, K., Wang, H.: Zero-knowledge arguments for lattice-based accumulators: Logarithmic-size ring signatures and group signatures without trapdoors. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 1–31. Springer, Heidelberg (May 2016). 10.1007/978-3-662-49896-5_1
37. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (Apr 2012). 10.1007/978-3-642-29011-4_41
38. O'Neill, A., Peikert, C., Waters, B.: Bi-deniable public-key encryption. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 525–542. Springer, Heidelberg (Aug 2011). 10.1007/978-3-642-22792-9_30
39. Peikert, C.: An efficient and parallel Gaussian sampler for lattices. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 80–97. Springer, Heidelberg (Aug 2010). 10.1007/978-3-642-14623-7_5
40. Quach, W., Wee, H., Wichs, D.: Laconic function evaluation and applications. In: Thorup, M. (ed.) 59th FOCS. pp. 859–870. IEEE Computer Society Press (Oct 2018). 10.1109/FOCS.2018.00086
41. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC. pp. 84–93. ACM Press (May 2005). 10.1145/1060590.1060603
42. Yao, A.C.C.: How to generate and exchange secrets (extended abstract). In: 27th FOCS. pp. 162–167. IEEE Computer Society Press (Oct 1986). 10.1109/SFCS.1986.25