Software with Certified Deletion

James Bartusek[1], Vipul Goyal[2,3], Dakshita Khurana[4], Giulio Malavolta[5,6], Justin Raizes[2], and Bhaskar Roberts[1]

[1] UC Berkeley, Berkeley CA, USA
bartusek.james@gmail.com,bhaskarr@eecs.berkeley.edu
[2] CMU, Pittsburgh PA, USA
vipul@cmu.edu,jraizes@andrew.cmu.edu
[3] NTT Research, Sunnyvale CA, USA
[4] UIUC, Champaign IL, USA
dakshita@illinois.edu
[5] Bocconi University, Milan, Italy
giulio.malavolta@hotmail.it
[6] Max Planck Institute for Security and Privacy, Bochum, Germany

Abstract. Is it possible to prove the deletion of a computer program after having executed it? While this task is clearly impossible using classical information alone, the laws of quantum mechanics may admit a solution to this problem. In this work, we propose a new approach to answer this question, using quantum information. In the *interactive* settings, we present the first fully-secure solution for blind delegation with certified deletion, assuming post-quantum hardness of the learning with errors (LWE) problem. In the *non-interactive* settings, we propose a construction of obfuscation with certified deletion, assuming post-quantum iO and one-way functions.

Our main technical contribution is a new deletion theorem for subspace coset states [Vidick and Zhang, EUROCRYPT'21, Coladangelo et al., CRYPTO'21], which enables a generic compiler that adds the certified deletion guarantee to a variety of cryptographic primitives. In addition to our main result, this allows us to obtain a host of new primitives, such as functional encryption with certified deletion and secure software leasing for an interesting class of programs. In fact, we are able for the first time to achieve a *stronger notion of* secure software leasing, where even a *dishonest* evaluator cannot evaluate the program after returning it.

1 Introduction

Consider the following scenario: Alice is a software developer who has written a program that she would like to sell, in order to get financially rewarded for her effort. Bob is interested in using Alice's software, but only for a limited amount of time. Can Alice temporarily lease her software to Bob, without the risk of him pirating the program?

Ideally, we would like to design a protocol where Alice can lease her software to Bob and start charging him for a subscription fee. Once Bob is done using the

software, he can produce a *deletion certificate* which guarantees that he deleted his local copy of the program. At this point, Alice can rest assured that Bob is no longer in possession of the software, and she can stop charging him. In case of a dispute, the deletion certificate will unequivocally determine which of the two parties misbehaved.

If we consider only classical information, then it is easy to see that no protocol can satisfy the security notion sketched above: Whatever information Alice sent to Bob, he can always create a perfect copy of it, thus continuing using the program even after producing the deletion certificate. In fact, nothing prevents Bob from pirating copies of Alice's program. On the other hand, the same argument does not hold if we consider quantum information, since the no-cloning theorem [42] postulates that there does not exist a general algorithm to create perfect copies of quantum states. Indeed, recent works on quantum copy protection [1] and secure software leasing [8,33,18] propose using quantum information to solve similar problems. Unfortunately, all of the existing definitions are subject to strong impossibility results [8,5] and the aforementioned works either attain heuristic constructions in idealized models, or focus on restricted classes of programs. At present, the problem of general-purpose software with certified deletion is wide open (even with quantum information).

To better understand the challenge of constructing software with certified deletion, let us make more concrete the desiderata for such protocol. To cover the full spectrum of software with certified deletion, in this work we consider and formalize two complementary settings.

Interactive settings: Blind delegation with certified deletion. In the interactive settings, Alice is assisting Bob to evaluate the program, i.e., to compute the output of the software on some input, via an exchange of messages. This allows Alice to control exactly how many inputs Bob has queried to the software, so she can charge him accordingly.

It is well-known that *fully-homomorphic encryption* [26] provides the ability to delegate a computation to an untrusted server, while revealing nothing about the computation itself. However, an FHE ciphertext is a classical string that information-theoretically contains Alice's software, and the only thing preventing recovery of this data is the conjectured hardness of a mathematical problem. If this problem becomes easy to solve in the future due to computational or scientific advances, or if Alice's secret key is leaked to Bob, there is no way to prevent Bob from recovering the underlying plaintext.

To mitigate this risk, we want Alice to be able to request Bob to delete their data at the end of the protocol, in such a way that, if Bob produces a valid certificate, then Alice is guaranteed that Bob has deleted the software *information theoretically*. This notion is known as *blind delegation with certified deletion*[7] [17,35,12]. However, existing proposals are actually insecure against

[7] In the standard notion of blind delegation, Alice delegates the computation of a hidden input on a public function to Bob. Alice is then assumed to receive the output. However, this is easily seen to be equivalent to the version where the function is also

a malicious Bob, who may deviate from the description of the protocol in an attempt to learn the Alice's software (more details on this later).

Non-interactive settings: Obfuscation with certified deletion. In the non-interactive settings, Alice ships a copy of the software to Bob, who can evaluate it freely on as many inputs as he wants. After some amount of time, Bob wants to produce a certificate of deletion that convinces Alice that, from that moment on, her software has been deleted information-theoretically. In other words, we want to encode a computer program into a quantum state that preserves its functionality, while enabling an evaluator to information-theoretically delete the underlying program.

We refer to this notion as *obfuscation with certified deletion*. Although a-priori it is not clear that this notion has anything to do with program obfuscation, we argue that the two are in fact intimately connected. After all, if Bob was able to learn Alice's software from its description, then there would really be no way to erase Bob's knowledge after the fact.

Limitations of existing approaches to certified deletion. Despite much recent progress designing cryptosystems with certified deletion [17,28,30,35,12,29], the above natural questions have remained unanswered. Arguably, we can trace the lack of progress back to the fact that we are missing a technique that allows for *repeated access* to partial information about the encoded data, followed by certified deletion of whatever is left. In other words, all works[8] thus far have focused on "all-or-nothing" style primitives: E.g. secret-key encryption, public-key encryption, attribute-based encryption, timed-release encryption, and commitments [38,17,28,30,12]. Even known constructions of *fully-homomorphic encryption* with certified deletion [35,12] are all-or-nothing, in the sense that security becomes compromised (as we show in this work) once we give the evaluator access to some type of decryption oracle.

1.1 Our Results

In this work, we introduce a new paradigm for secure information-theoretic deletion of data. Our main technical ingredient, that enables all of our results, is a new deletion theorem for *subspace coset states* [39,21]. Subspace coset states have previously been used for designing *uncloneable* cryptographic primitives [21,6], and we demonstrate how to use them to obtain information-theoretic deletion. Our proof technique generalizes the work of [12] to states beyond BB84

hidden, using universal circuits. Furthermore, we can let Bob receive the output by adding one more round.

[8] An exception to this claim is a very recent work of [29] which constructs *functional encryption* with certified deletion of ciphertexts. However, in contrast with one of the goals of this work, their scheme is only secure in the setting of bounded collusions, where there is an a-priori upper bound on the number of functional keys an adversary can request.

states. This allows us to achieve *information-theoretic* certified deletion for cryptographic primitives beyond all-or-nothing. Specifically, we obtain the following results:

- **Blind delegation.** We develop the first maliciously secure blind delegation protocol with certified deletion. Our construction is based solely on the quantum hardness of learning with errors (LWE) [36], and carefully combines subspace coset states with compact fully-homomorphic encryption (FHE) [26] and succinct non-interactive arguments (SNARGs) for P.
  We also provide the first construction of *two-message* blind delegation with certified deletion, based on post-quantum sub-exponentially secure indistinguishability obfuscation. In particular, once the client sends their encoding of x, the server can return *both* the evaluated output $f(x)$ *and* a certificate that all other information about x has been deleted without any more interaction with the client.
- **Obfuscation.** Assuming post-quantum indistinguishability obfuscation, we obtain the first construction of differing inputs obfuscation with certified deletion ($\mathsf{di}\mathcal{O}\text{-}\mathsf{CD}$), for a polynomial number of differing inputs. Loosely speaking, $\mathsf{di}\mathcal{O}\text{-}\mathsf{CD}$ satisfies the standard notion of differing inputs obfuscation [10], in addition to the following certified deletion property: Let Π_0 and Π_1 two programs that differ on one input y^* (or a polynomial number of hard to find inputs), then it is hard to distinguish an obfuscation of Π_0 from an obfuscation of Π_1, *even given a differing input* y^*, provided that the distinguisher outputs the deletion certificate first. Intuitively, this formalizes the guarantee that, after deleting a program, one can no longer evaluate it on any input (more discussion on this later).

We can also conceptually abstract the above results as the following (informal) theorem, in an oracle model: For any classical functionality f, one can prepare an oracle that can be queried repeatedly (polynomially many times) before being *permanently deleted*. That is, after deletion, even an unbounded number of queries to the oracle will not reveal any more information about f. This general result may be of independent interest.

To demonstrate the usefulness of our newly developed tools, we show how they enable new applications in quantum cryptography, and in some cases they allow us to make progress on important open problems:

- **Secure Software Leasing.** As an immediate corollary of differing inputs obfuscation with certified deletion, we obtain a strong notion of secure software leasing for every differing inputs circuits family. Whereas the standard notion guarantees that the *honest* evaluation procedure fails for pirated copies of software, this strong notion guarantees security against arbitrary evaluation procedures.
- **Functional encryption.** We obtain two flavors of functional encryption with certified deletion: (i) one where *ciphertexts* can be certifiably deleted, and (ii) one where *secret keys* can be certifiably deleted (also known as key revocation or secure key leasing). The former assumes a strong-enough notion

of post-quantum functional encryption (in particular, public-key multi-input FE with arity 2). The latter follows from differing inputs obfuscation with certified deletion, combined with post-quantum public-key encryption and injective one-way functions. Functional encryption with key revocation is our *only* result with a computational certified deletion guarantee. This is inherent in the primitive, as key revocation only emulates the case where a secret key was never received.
- **Public verification.** We develop a generic compiler that results in a variety of primitives with *publicly verifiable* certified deletion, assuming post-quantum indistinguishability obfuscation.

2 Technical Overview

2.1 Warm-Up Example

We illustrate the challenges and the techniques that we introduce in this work via a toy example. Namely, we will start from the, by now standard, notion of encryption with certified deletion and highlight the barriers that one encounters when trying to reveal some partial information about the plaintext. Specifically, we will try to build obfuscation with certified deletion starting from the latter.

Public-key encryption with certified deletion. We recall the basic notion of public-key encryption with certified deletion, and describe a recent construction due to [12] based on Wiesner encodings / BB84 states [41,14]. For describing these states, we use the notation $|x\rangle_\theta$, where $x \in \{0,1\}^n$ is a string of bits, and $\theta \in \{0,1\}^n$ is a string of basis choices. Let Enc be the encryption algorithm for a post-quantum public-key encryption scheme. Then to encrypt a bit b, sample $x, \theta \leftarrow \{0,1\}^n$, and release

$$|x\rangle_\theta \, , \mathsf{Enc}(\theta, b \oplus \bigoplus_{i:\theta_i=0} x_i).$$

To delete, measure $|x\rangle_\theta$ in the Hadamard basis to obtain a string x'. This verifies as a valid deletion certificate if $x'_i = x_i$ for all $i : \theta_i = 1$. [12] show that since Enc is semantically secure and thus hides the choice of θ, any computationally-bounded adversary that produces a valid deletion certificate must have (essentially) measured most of the qubits in the Hadamard basis, erasing enough information about $\{x_i\}_{i:\theta_i=0}$ to claim that b is now statistically hidden.

Obfuscation and malleability. One nice property of the above scheme is that it can be decrypted classically after measuring $|x\rangle_\theta$ in the computational basis. This suggests a natural construction for obfuscation with certified deletion. First, encrypt (with certified deletion) the description of the circuit C. Then, obfuscate the classical program that does the following: given a circuit input, the secret key, and a classical measurement outcome obtained from the ciphertext encrypting C, recover the description of C, and then evaluate it on the input.

Unfortunately, such a construction *does not even satisfy indistinguishability obfuscation* (let alone any certified deletion guarantee). The issue is that the encryption scheme is clearly malleable: An adversary only has to guess a single index i where $\theta_i = 0$ in order to flip the message bit. Let's imagine an adversary that can maul the ciphertext to delete a single gate. It tries to distinguish an obfuscation of C_0 from one of C_1, where C_0 and C_1 are built from the same base circuit except that C_0 appends an identity gate and C_1 appends two consecutive NOT gates. This adversary can attempt to remove the last gate in the circuit. If this flips the output, the gate must have been C_1. This simple mauling capability therefore violates indistinguishability obfuscation (even *before* deletion). Under more sophisticated mauling attacks, it may even be possible to recover the whole circuit from the obfuscation!

Ciphertext validity check. A simple idea to overcome this issue is to enable the classically obfuscated program to check that the ciphertext has not been tampered with. Say the adversary provides y to the obfuscated program as the alleged measurement of $|x\rangle_\theta$. To verify that the ciphertext is intact, the program only needs to verify that $y_i = x_i$ whenever $\theta_i = 0$. This can be done using a hard-coded x and θ. Otherwise, it can output $\perp$.

Unfortunately, the encryption scheme becomes completely insecure in the presence of such a program. An adversary can learn a description of θ one bit at a time, by flipping a bit of its state $|x\rangle_\theta$ and observing whether the program returns a successful evaluation or rejects. Once it learns θ, we cannot hope for any certified deletion guarantees. Moreover, the adversary can make additional queries to learn $\{x_i\}_{i:\theta_i=0}$, and, eventually, the circuit C.

Subspace coset states. Fortunately, there is a way to get around the problem that BB84 states are learnable in this sense. Prior work (for example, in the setting of publicly-verifiable quantum money) has switched to using entangled *subspace* states [2] and the more-general subspace coset states. A subspace coset state is defined by a subspace S of $\mathbb{F}_2^n$ and two vectors $\mathsf{v}, \mathsf{w} \in \mathbb{F}_2^n$, and is written as

$$|S_{\mathsf{v},\mathsf{w}}\rangle := \frac{1}{\sqrt{|S|}} \sum_{\mathsf{z} \in S + \mathsf{v}} |\mathsf{z}\rangle \, (-1)^{\langle \mathsf{z}, \mathsf{w} \rangle}.$$

It is useful to think of BB84 states as a type of subspace coset state in which the subspace is spanned by the standard basis vectors $\{e_i\}_{i:\theta_i=1}$. The coset in the primal space is determined by the bits $\{x_i\}_{i:\theta_i=0}$, which are used to hide the plaintext bit b, and the coset in the dual space is determined by the bits $\{x_i\}_{i:\theta_i=1}$, which determine what constitutes a valid deletion certificate.

Thus, in an attempt to make the obfuscation scheme secure, we replace the use of BB84 states with more-general subspace coset states. To encrypt each bit b of the description of the circuit, consider a ciphertext of the form

$$|S_{\mathsf{v},\mathsf{w}}\rangle \, , \mathsf{Enc}(S, C \oplus \langle \mathsf{v}, \mathbf{1} \rangle),$$

where we set S to be a random $n/2$-dimensional subspace, and a valid deletion certificate is now any vector $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$. The decryption algorithm, on input a

vector z and ciphertext ct, will decrypt ct to obtain (S, b'), compute a canonical coset representative of $S + \mathsf{z}$, and use this resulting vector to unmask b.

Additionally, it is possible to check whether z has been tampered with by verifying that $\mathsf{z} \in S + \mathsf{v}$. It is even possible to publish an oracle for this consistency check, *without* leaking S and v [21]. However, proving that the consistency oracle does not compromise the certified deletion security of the encryption scheme requires new ideas, and is a main technical contribution of this work.

Noisy consistency check. In order to carry out this consistency check, the obfuscated program must have S and v hard-coded. Unfortunately, the obfuscation only hides S and v computationally. After deletion, an unbounded adversary could learn v and $b \oplus \langle \mathsf{v}, \mathbf{1} \rangle$, which reveals b.

To information-theoretically protect b after deletion, we will instead sample a random superspace of S called T and hard-code the coset $T + \mathsf{u}$ that contains $S + \mathsf{v}$. We set $\dim(T) = 3n/4$ as a happy medium, which has two nice properties. First, since T is a negligible fraction of $\mathbb{F}_2^n$, it is hard for an adversary to find a vector in $T + \mathsf{u} \setminus S + \mathsf{v}$, so the consistency check will be essentially as good as using $S + \mathsf{v}$. Second, since S is a negligible fraction of T, $T + \mathsf{u}$ statistically hides enough information about v that $\langle \mathsf{v}, \mathbf{1} \rangle$ is uniformly random, even given $T + \mathsf{u}$. Therefore, the proof of certified deletion works.

It turns out that this "noisy consistency check" will be also be a crucial component in our constructions of both blind delegation and functional encryption with certified deletion.

2.2 General Compiler for Certified Deletion

Now we will present the tool that underlies all of our applications: a compiler that adds a certified deletion guarantee to a variety of cryptographic primitives.

First consider a simple template for certified deletion: to hide a bit b, we give the adversary the following state:

$$|S_{\mathsf{v},\mathsf{w}}\rangle, \mathcal{Z}(S, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle),$$

where $|S_{\mathsf{v},\mathsf{w}}\rangle$ is a random subspace coset state and $\mathcal{Z}$ is some side information, which may be classical. $\mathcal{Z}$ will often represent the primitive to which we are adding a certified deletion guarantee.

Note that given only the side information, b is statistically hidden because it is masked by $\langle \mathsf{v}, \mathbf{1} \rangle$. However, the information needed to remove the mask v is stored in the computational basis of the subspace coset state. To prove deletion, an honest party measures the subspace coset state in the Hadamard basis to get a vector $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$, which destroys essentially all information about v and removes b from their view. We will hope to prove that *any* strategy an (efficient) adversary uses to obtain a $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$ will also statistically remove b from their view.

The recent work of [12] showed how to prove this when $\mathcal{Z}$ satisfies *semantic security* with respect to S.[9] However our applications need a much richer set

[9] Also they only consider the case where the quantum state is a string of BB84 states.

of choices for $\mathcal{Z}$ that do not necessarily hide S semantically. For instance, we want the ability to perform the noisy consistency check. That is: we want $\mathcal{Z}$ to output a randomized (T, u) where $S + \mathsf{v} \subset T + \mathsf{u}$. This is essential for our constructions of blind delegation and obfuscation with certified deletion. But $\mathcal{Z}$ would no longer hide S semantically because T reveals some basis vectors of $S^\perp$. In this case [12]'s proof falls short.

We present a compiler that supports a greater variety of choices for $\mathcal{Z}$, including the noisy consistency check. Specifically, we develop techniques to allow any choice of $\mathcal{Z}$ that satisfies a form of subspace-hiding against QPT adversaries. Morally, subspace-hiding means that an adversary cannot tell whether $\mathcal{Z}$ was testing membership in S (and $S^\perp$) or random superspaces $T \geq S$ (and $R \geq S^\perp$). We sketch our notion of subspace-hiding below:

Definition 1 (Subspace-Hiding, Informal). *Given any subspace S of dimension $n/2$ and two cosets $S+\mathsf{v}$ and $S^\perp+\mathsf{w}$, let $T+\mathsf{u}$ and $R+\mathsf{x}$ be random cosets that contain the first two: $S+\mathsf{v} \subset T+\mathsf{u}$ and $S^\perp+\mathsf{w} \subset R+\mathsf{x}$. Then, $\mathcal{Z}(S, T, \mathsf{u}, \mathsf{w}, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle)$ is subspace-hiding if there exists a simulator $\mathcal{S}(R, T, \mathsf{u}, \mathsf{x}, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle)$ such that*

$$\mathcal{Z}(S, T, \mathsf{u}, \mathsf{w}, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle) \approx_c \mathcal{S}(R, T, \mathsf{u}, \mathsf{x}, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle),$$

where $\approx_c$ denotes indistinguishability to a quantum polynomial-time adversary.

Next, we claim that if $\mathcal{Z}$ satisfies subspace-hiding (which is a notion of *computational* security), then after the deletion certificate is accepted, b is *statistically* hidden, even if the inputs to $\mathcal{Z}$ are leaked at a later point. We sketch the security claim below.

Claim (Certified Deletion Security, Informal). Let $\mathsf{EXP}(b)$ be the output of the following experiment:

1. *Challenge:* The challenger samples the following challenge and sends it to the adversary:
$$|S_{\mathsf{v},\mathsf{w}}\rangle, \mathcal{Z}_\lambda(S, T, \mathsf{u}, \mathsf{w}, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle)$$
2. *Response:* The adversary responds with a deletion certificate $\widetilde{\mathsf{z}} \in \mathbb{F}_2^n$ and an auxiliary state ρ.
3. *Outcome:* The challenger checks that
$$\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$$
If so, they output ρ and all the inputs to $\mathcal{Z}$; if not, they output $\perp$.

If $\mathcal{Z}$ is computationally subspace-hiding, then the statistical distance between $\mathsf{EXP}(0)$ and $\mathsf{EXP}(1)$ is negligible.

Proof overview. First, we claim that b is statistically hidden given *only* the side information $\mathcal{Z}(S, T, \mathsf{u}, \mathsf{w}, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle)$. Note that b is masked by $\langle \mathsf{v}, \mathbf{1} \rangle$, and although $\mathcal{Z}$ may give some information about v in the form of (T, u), there is still some randomness left in v. In more detail, we can decompose v into

its deterministic and random components by defining $\mathsf{v}_0 = \mathsf{v} - \mathsf{u}$. Then given $(S, T, \mathsf{u}, \mathsf{w})$, u is deterministic, and v_0 is uniformly random over $\mathsf{co}(S) \cap T$.[10] Because v_0 is uniformly random given $(S, T, \mathsf{u}, \mathsf{w})$, the bit $\langle \mathsf{v}, \mathbf{1} \rangle$ is also uniformly random (with overwhelming probability over the choice of (S, T)).

Next, recall that the adversary's view also includes the quantum state $|S_{\mathsf{v},\mathsf{w}}\rangle = |S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$, which stores v_0 in the computational basis. Now, b is not necessarily statistically hidden given both $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$ and $\mathcal{Z}(S, T, \mathsf{u}, \mathsf{w}, b \oplus \langle \mathsf{v}, \mathbf{1} \rangle)$. However, we will show that to prove deletion, the adversary must essentially measure $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$ in the Hadamard basis, destroying all information that the state had about v_0.

To show this, instead of giving the adversary $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$, we imagine giving them the following state, which stores a random $\widetilde{\mathsf{v}}_0$ in the Hadamard basis:

$$|T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}\rangle \qquad \text{where } \widetilde{\mathsf{v}}_0 \leftarrow \mathsf{co}(T^\perp) \cap S^\perp.$$

$|T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}\rangle$ is in some sense dual with $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$. Both states store u in the computational basis and w in the Hadamard basis. The only difference is that $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$ encodes a random v_0 in the computational basis, and instead $|T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}\rangle$ encodes a random $\widetilde{\mathsf{v}}_0$ in the Hadamard basis. Furthermore, the adversary's behavior will be the same no matter which of the two states we give them. Indeed, we show that for any fixed $S, T, \mathsf{u}, \mathsf{w}$, the following states σ_0 and σ_1 are equivalent:[11]

$$\sigma_0 \propto \sum_{\mathsf{v}_0 \in \mathsf{co}(S) \cap T} |S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle \langle S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}|$$
$$\sigma_1 \propto \sum_{\widetilde{\mathsf{v}}_0 \in \mathsf{co}(T^\perp) \cap S^\perp} |T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}\rangle \langle T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}|$$

This can be seen as a generalization of the fact that

$$\frac{1}{2}\big(|0\rangle\langle 0| + |1\rangle\langle 1| \big) = \frac{1}{2}\big(|+\rangle\langle +| + |-\rangle\langle -| \big).$$

In other words, the state is the same whether it's a maximal mixture of computational basis eigenstates or Hadamard basis eigenstates. Establishing this claim requires new techniques which seem to be generally useful for handling subspace coset states (more detail in Section 2.3).

Now, we want to argue that if the adversary produces a valid deletion certificate $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$, then given their remaining state, v_0 is statistically close to uniform. Imagine the adversary is given $|T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}\rangle$ and they output a valid deletion certificate $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$ with non-negligible probability. Recall that $T^\perp + \widetilde{\mathsf{v}}_0 + \mathsf{w}$ is an affine subspace of $S^\perp + \mathsf{w}$, so one way to do this is to make a measurement of the vector $\widetilde{\mathsf{v}}_0 + \mathsf{w}$ encoded in the phase, producing a vector in $T^\perp + \widetilde{\mathsf{v}}_0 + \mathsf{w}$. In

[10] $\mathsf{co}(S)$ is a group of coset representatives of S. See Section 4.1 for a precise definition of $\mathsf{co}(S)$.

[11] This is implicitly shown in the supplementary material by purifying σ_0 and σ_1 and showing that there exists a unitary acting on the purifying register that maps between the two states.

fact, we show that since $\mathcal{Z}$ is computationally subspace-hiding for S, *any* adversary's strategy must be *statistically close* to making a measurement of $\widetilde{\mathsf{v}}_0 + \mathsf{w}$.[12] Then, if the adversary were instead given $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$, this same measurement of the phase would destroy all information about v_0, completing the proof.

2.3 Discussion

To gain some context, it is useful to zoom out from our main theorem, and compare our proof technique with existing works. As we shall see shortly, our settings require new proof techniques and cannot be framed as a special case of existing theorems.

New techniques for subspace coset states. While the previous section provides intuition, our actual proof is trickier and requires new techniques. Essentially, facts that are obvious for continuous vector spaces are sometimes false or difficult to formalize for discrete vector spaces. We develop new techniques for working with subspace cosets, and the culmination is an algorithm for *delayed preparation of subspace coset states*. Section 4 presents these results.

Our first contribution is to define a coset group $\mathsf{co}(S)$ that is isomorphic to $\mathbb{F}_2^n/S$ and that is a subspace of $\mathbb{F}_2^n$. This improves on prior work, [21], which defined a set of canonical coset representatives that was not necessarily a group. The algebraic structure of $\mathsf{co}(S)$ allows us to prove more-sophisticated claims than what was possible with [21]'s coset representatives.

Our second contribution is to develop a toolkit for proving such claims.

Our third contribution is an algorithm for delayed preparation of subspace coset states. This formalizes the intuition that the adversary's behavior is the same whether they are asked to play a game based on $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$ or a game based on $|T_{\mathsf{u},\tilde{\mathsf{v}}_0+\mathsf{w}}\rangle$. We formalize this by showing that we can purify σ_0 and σ_1 by adding a second register, and then show that there is a unitary that acts on the second register and maps the purification of σ_0 to that of σ_1. The unitary could be applied after the adversary acts on the first register, so the adversary's behavior will be the same in either case.

Why monogamy-of-entanglement techniques fail. Prior works [21,37] that dealt with subspace coset states relied on monogamy-of-entanglement (MoE) theorems, but these theorems fail to achieve the strong guarantees needed in our setting.

First, monogamy of entanglement is an information-theoretic property, and it does not necessarily hold if the adversary receives a computationally-secure encryption of the subspace S. We note that a recent work [7] does use a MoE property to establish a certified deletion property, but crucially only in the information-theoretic one-time-pad encryption setting.

[12] That is, we use *computational* hardness to establish a *statistical* claim, as done in [12] in the context of BB84 states.

Next, MoE claims in prior work do not seem to easily extend to rule out the possibility that Bob outputs the string x, and Charlie simultaneously outputs the parity of x with non-negligible advantage. If this were possible, then even when one player produces a valid deletion certificate, the other player might learn a bit of data with non-negligible advantage, which would violate certified deletion security.

Why non-committing encryption techniques fail. Recall that we would like to eventually prove information-theoretic deletion of a secret that is initially information-theoretically determined by the adversary's view. Prior works (e.g., [4]) used receiver non-committing encryption schemes which have an "equivocality" property, allowing one to sample the fake keys after S is revealed. These were inherently limited to proving weaker forms of security; e.g., restricted to (computational) security against key-leakage attacks. Furthermore, equivocality is hard to achieve [31] for applications such as blind delegation, which involves FHE. Another setting where an equivocality-based approach fails is differing-inputs obfuscation. The choice of whether to behave as C_0 or C_1 is "hidden" under the differing inputs. Thus, the differing inputs act as a key to decrypt S, which reveals this choice bit. However, any differing input (i.e. key) y^* is easy to check by simply evaluating the two programs C_0 and C_1 on y^*, allowing fake keys to be immediately recognized. While [12] developed methods to overcome the equivocality issue for certified deletion, they only apply their techniques to settings where the subspace S is semantically hidden.

2.4 Blind Delegation with Certified Deletion

In this section, we discuss our construction of maliciously-secure blind delegation with certified deletion.

Insecurity of prior protocols against malicious adversaries. We first discuss why both prior protocols [35,12], while secure against semi-honest adversaries, are *insecure* against malicious adversaries.

Both of these protocols consist of four messages. First, the client encrypts their input m and sends a quantum ciphertext $|\mathsf{Enc}(m)\rangle$ to the server. Next, the server evaluates a function f to obtain a register holding a superposition over output ciphertexts $|\mathsf{Enc}(f(m))\rangle$, which is sent to the client. The client then coherently applies FHE decryption using their secret key, which allows them to recover $f(m)$ without disturbing the state, and then reverse their computation and send the undisturbed register back to the server. Finally, the server can uncompute f and recover the original ciphertext $|\mathsf{Enc}(m)\rangle$. Then, if they want, they can measure the ciphertext in a particular way to recover a certificate of deletion, which is sent to the client.

Now, consider the following attack. Suppose that the server wants to learn the first bit m_1 of m. They can easily prepare a state of the form

$$\frac{1}{\sqrt{2}}|\mathsf{Enc}(m_1)\rangle^{\mathsf{C}}|0\rangle^{\mathsf{S}} + \frac{1}{\sqrt{2}}|\mathsf{Enc}(0)\rangle^{\mathsf{C}}|1\rangle^{\mathsf{S}},$$

where $\mathsf{Enc}(0)$ is a freshly prepared encryption of 0. Then, suppose they send register C to the client in place of the second message of the protocol described above. In the case that $m_1 = 0$, the client's computation will not disturb the state, and the server will receive back the C register unharmed. But in the case when $m_1 = 1$, the client's measurement of the output *will* collapse the state. Thus, if the server unentangles register C and S, measures S in the Hadamard basis, and observes outcome $|-\rangle$, they will learn *for sure* that $m_1 = 1$, breaking privacy of the protocol.[13]

Our solution. The attack above relies on the fact that the client always immediately applies an operation that depends on their FHE secret key sk. To prevent this attack, we must introduce a way for the client to *check* that the server is honestly following the protocol, *before* using its secret key to operate on the state.

Suppose the client's input is a single bit b, and consider our basic encryption scheme

$$|S_{\mathsf{v},\mathsf{w}}\rangle\,, \mathsf{Enc}(S, b \oplus \langle \mathsf{v}, \mathbf{1}\rangle),$$

but where Enc is now instantiated as a fully-homomorphic encryption (FHE) scheme. We will have the server perform a classical FHE evaluation for circuit f in superposition over the vectors in $S + \mathsf{v}$, resulting in a superposition over $\mathsf{Enc}(f(b))$. Now, the client will need to perform two checks to make sure the server was behaving honestly:

- The client needs to check that the FHE evaluation in superposition was performed honestly. This can by accomplished by requesting that the server use a succinct non-interactive argument (SNARG) for P (polynomial-time computation) in superposition, and having the client verify this proof before decrypting. Moreover, SNARGs for P are known just from the LWE assumption [20], and it is straightforward to see that this SNARG remains post-quantum secure assuming the post-quantum hardness of LWE.
- The client *also* needs to check that the *input* to the server's computation is honest. This input is supposed to include any vector in $S + \mathsf{v}$. Thus, one solution is to have the client remember a description of $S + \mathsf{v}$, and also perform this check on the input before decrypting the output. However, this

[13] This attack does not contradict any claims made in [35] or [12] because neither paper claims that their protocol is maliciously-secure. In [35], the correctness and security properties are defined entirely separately. That is, it is argued that correctness of the four-message protocol holds assuming parties are honest, but *security* (and certified deletion security) is only argued *assuming that the server does not interact with the client*. Thus, the claim is essentially that *either* correctness of delegation holds, *or* privacy against a malicious server holds. But it is never claimed that both can hold simultaneously. In [12], the authors do jointly consider correctness and security of the four-message protocol. However, they only claim security against servers that are *semi-honest* during the protocol execution (and potentially malicious after, while producing the deletion certificate). Thus, the above attack is explicitly disallowed by the semi-honest assumption.

solution requires that the client keep the vector v around in memory, which if leaked would completely compromise certified deletion security. Instead, we use the same "noisy consistency check" as explained earlier, and have the client sample a random affine superspace $T + \mathsf{u}$ of $S + \mathsf{v}$, and only keep this around in memory.

This is the basic idea of our blind delegation scheme.

Remarks on the security definition.

- We prove simulation-based security in the "protocols with certified deletion" framework of [12]. That is, we show that our protocol securely realizes an ideal functionality that takes input (f, x) from the client, delivers *only* f to the server, and delivers the output $f(x)$ to the client. Thus, in the simulated world, the server obtain *no* information about the client's private input x. We show that conditioned on the server producing a valid deletion certificate, their final view in the ideal protocol and in the ideal world are *statistically* indistinguishable, indicating that they have information-theoretically lost all information about x.
- We show that our protocol in fact realizes a *reusable* ideal functionality, where the client can request that the server compute for them multiple functions $f_1(x), f_2(x), \ldots$ on their original encrypted data x, and security still holds.
- In the case that deletion is accepted, we also explicitly leak all of the client's secret parameters to the adversary, and still require that statistical security holds. We capture this by defining a "long-term secrets" tape sec, where after each message, the honest client is supposed to write all of the information it needs to interact in the remainder of the protocol on this tape. Conditioned on deletion being accepted, we give the adversary access to this tape.

Finally, we remark that prior to our work, maliciously-secure blind delegation with certified deletion was not known *even without reusability*, and *even without requiring information-theoretic security after deletion.*

2.5 Obfuscation with Certified Deletion

A certifiably deletable program is an encoding of a classical circuit C into a quantum state $|\widetilde{C}\rangle$ that allows for evaluation of $C(x)$ on any input x. To realize certified deletion, there should also be a procedure for measuring $|\widetilde{C}\rangle$ in a particular way that certifiably destroys information about C.

While the natural notion of virtual black-box obfuscation is impossible to achieve in general [10], several relaxed notions are plausibly achievable. We focus on the case of differing inputs obfuscation for a polynomial number of differing inputs [10], which is implied by the related notion of indistinguishability obfuscation [15]. This notion requires that for any two circuits C_0 and C_1 which differ on a polynomial number of hard-to-find inputs, an obfuscation of C_0 is indistinguishable from an obfuscation of C_1.

Our goal is thus to achieve differing inputs obfuscation with certified deletion for a polynomial number of differing inputs.[14] This is described by the following (simplified) game.

1. The challenger samples two programs C_0 and C_1 from a given distribution, where it holds that $C_0(y) = C_1(y)$, except for some y^*, that we refer to as the *differing input*. Importantly, even given the description of C_0 and C_1, it is computationally hard to find y^*.
2. The challenger flips a coin $b \leftarrow \{0,1\}$ and sends an obfuscation of C_b to the attacker.
3. At some point of the experiment, the attacker outputs the deletion certificate, which is verified by the challenger.
4. If the certificate correctly verifies, then the challenger sends y^* to the attacker.
5. The attacker outputs a guess for the bit b.

We say that an obfuscation scheme is secure if the success probability of the attacker is negligibly close to 1/2. We note that if the attacker becomes unbounded after outputting a valid deletion certificate, then they could compute y^* themselves. Thus, we remove Step 4 in our definition for information-theoretic certified deletion.

To justify this definition, we argue that it captures the intuitive guarantees that one would expect from software with certified deletion. First, there is a sense in which software with certified deletion implies some notion of obfuscation: if one could learn the program by just looking at its code, then there would be no point in issuing a deletion certificate. Second, we want to model the fact that once the adversary has deleted the program, they can no longer evaluate it on any input. However, it is not clear how to model the information which the adversary learned before deletion, i.e., when they had a *functional* copy of the program. We have no way of "looking inside the adversary's head" to learn which inputs they evaluate, and, even worse, the attacker may not even execute the program properly. Our definition sidesteps this issue, by requiring security for inputs that are hidden even given the plain description of the program (i.e., the differing inputs). In this sense, our definition can be interpreted as saying that the deletion prevents the adversary from learning any information that was not obviously leaked from having a running copy of the program.[15]

Construction. As outlined previously, the general structure of the construction is to encrypt the circuit C under a random coset v of the subspace S. Suppose

[14] Our definition also generalizes to the case of an arbitrary number of differing inputs. However, achieving this would imply the existence of general differing inputs obfuscation, contrary to current evidence [13,24].

[15] Our definition does not prevent an adversary from evaluating the program on *easy-to-find* inputs after deletion. This is because we cannot rule out the possibility of them having already evaluated those inputs before deletion.

for a moment that we can simultaneously hide all bits of C with a single vector v. To use the noisy consistency check, we will sample a uniform superspace $T+\mathsf{u}$ of $S+\mathsf{v}$. Then, we will hard-code S, T and u into a classical program $P[S,T,\mathsf{u},C+\mathsf{v}]$ to be obfuscated. $P[S,T,\mathsf{u},C+\mathsf{v}]$ takes as input a vector z (which should be in $S+\mathsf{v}$) and a string x to be evaluated. It checks that $z \in T+\mathsf{u}$, then computes the coset v of S that z belongs to, unmasks C using v, and finally computes and outputs $C(x)$. If $z \notin T+\mathsf{u}$, it aborts. Then, the construction is

$$|S_{\mathsf{v},\mathsf{w}}\rangle\,, \mathsf{Obf}(P[S,T,\mathsf{u},C+\mathsf{v}])$$

To argue security, we would like to switch an obfuscation of C_0 to an obfuscation of C_1, and argue that this switch is *statistically* indistinguishable to an adversary that produces a successful deletion certificate. Our main theorem provides a way to obtain such statistical guarantees, but it only handles statistically hiding a single bit. Thus, we must perform a hybrid argument over the bits of the descriptions of the circuits. We cannot do this naively, since descriptions of circuits "in between" C_0 and C_1 are not guaranteed to be functionally equivalent to C_0 and C_1. Instead, we make use of the *two-slot technique* [34], and we defer details of this to the technical sections.

The above describes the main intuition and techniques that allow us to hide functionality, while still allowing for certified deletion. In the body of the paper, we also derive the following related results and applications.

Application: Strong Secure Software Leasing. Secure software leasing is defined with respect to a family of programs [8]. The adversary is given a leased program randomly chosen from this family and outputs two programs. If one of the programs is authenticated, then the other cannot be evaluated using the honest evaluation procedure.

We observe that any differing inputs program family can be securely leased by obfuscating it with certified deletion. A differing inputs program family contains pairs of programs (C_0, C_1) such that given a random pair, it is hard to find an input y^* where $C_0(y^*) \neq C_1(y^*)$. If an obfuscation of C_0 is returned to the lessor who then generates a valid deletion certificate, then the residual state cannot be used to distinguish whether the program was C_0 or C_1, even given a differing input y^*. In particular, the adversary that returned the program cannot later evaluate a pirated copy of it on y^* - otherwise they could check which program matched the output. Therefore, a leased program can be validated by attempting to delete it and checking the deletion certificate.

We emphasize that this guarantee is *stronger* than the original notion of secure software leasing, which permits the adversary to evaluate a pirated (i.e. unauthenticated) program as long as they do not use the honest evaluation procedure. In our definition, security is guaranteed even if the adversary uses an *arbitrary* evaluation procedure after returning a valid copy of the program.

Since we construct obfuscation with certified deletion for a polynomial number of differing inputs, we immediately obtain (strong) secure software leasing for

differing inputs program families with a polynomial number of differing inputs. Existing impossibility results for secure software leasing [8,5] rule out secure software leasing for families containing programs which cannot be learned with black-box query access, but *can* be learned using non-black-box access to any functionally equivalent program. In contrast, a differing inputs program family contains programs which cannot be learned, even with non-black-box access to an obfuscation of the program.

Application: Functional Encryption with Key Revocation. To substantiate the usefulness of our definition, we show that our obfuscation scheme allows a simple and intuitive construction of public-key encryption, and even *functional encryption*, with key revocation. Moreover, our key revocation guarantee is *publicly-verifiable*. In key revocation, one or more secret keys are temporarily distributed to users. Later on, if the users comply with the revocation process, these keys are deleted and cannot be used to decrypt freshly generated ciphertexts [3,9].[16]

Our construction is essentially the same as the transformation from obfuscation to functional encryption given in [23], but our obfuscation scheme supports certified deletion. We describe a simplified version of their construction here. The secret key for a function f will be an obfuscated circuit that first decrypts a classical ciphertext to recover the message m, and then computes and returns $f(m)$. The encryption of m will use a standard public-key encryption scheme.

The above construction already guarantees that a key sk_f *only* reveals information about $f(m)$, by virtue of being a functional encryption scheme. The certified deletion property *additionally* ensures that, if the adversary has a key for f, but deletes it before receiving the challenge ciphertext, then he learns nothing. In fact, a straightforward reduction to the certified deletion security of the obfuscation scheme ensures that this is the case even if the adversary has access to other secret keys (security against *unbounded* collusion). We note that a similar technique allows adding publicly-verifiable key revocation to other encryption schemes, assuming iO.

3 Related Work

3.1 Prior Work

We first discuss prior works that build cryptographic schemes from subspace coset states. These states were first used by [39] in the context of proofs of quantum knowledge and by [21] to construct signature tokens (among other unclonable primitives) in the plain model. These were also used to build semi-quantum tokenized signatures [37]. Most recently, [6] used subspace coset states to construct unclonable encryption satisfying the notion of unclonable indistinguishability. We remark that, while there are clearly similarities between the

[16] This property has also been referred to as secure key leasing.

notions of unclonable encryption and encryption with certified deletion, our security definitions and proofs are quite different than those in [6]. For example, [6] crucially rely on *random oracles*, while our results are all in the plain model. Moreover, we achieve security definitions that promise *everlasting security* against unbounded adversaries after deletion, while [21,6,37] focuses on proving that computationally-bounded (or query-bounded) adversaries cannot perform a certain task, e.g., generating additional signatures.

Next, we mention two prior works that have considered functional encryption with certified deletion. [32] achieves a *private*-key version of functional encryption with certified deletion of secret keys. [29] achieves functional encryption where the ciphertext can be deleted and it is certified everlasting (i.e. information theoretic certified deletion). Their construction is secure against bounded collusions, and either relies on the QROM or requires quantum certificates of deletion, and assumes public-key encryption. On the other hand, our functional encryption schemes support *public*-key encryption and are secure (in the sense of certified deletion) against an unbounded number of colluding users. We assume iO, which (up to subexponential hardness factors) is necessary, since it is implied by unbounded-collusion FE.

Finally, we remark that the blind delegation protocol of [35] is also shown to be publicly-verifiable, under the strong Gaussian-collapsing conjecture, and thus our results on blind delegation with public verification are technically incomparable. However, [35]'s conjecture involves an interactive game that has a baked in certified deletion component, wherein the adversary receives some trapdoor information conditioned on them successfully returning a pre-image of the hash function. On the other hand, indistinguishability obfuscation a priori has nothing to do with certified deletion. While post-quantum indistinguishability obfuscation is also not known from standard assumptions, this is a very active area of research with many candidates proposed over the last few years [11,19,16,25,40,22].

3.2 Concurrent and Independent Work

We will discuss three recent works [27,3,9] that construct revocable/deletable cryptographic primitives, a few of which overlap with ours. We compare the results in more detail below. At a high level, our constructions produce *publicly-verifiable, classical* deletion certificates in the plain model. The deletion certificates of [27,3,9] are all *privately* verifiable, and some of their constructions require *quantum* deletion certificates. Furthermore, our work unifies techniques using a general compiler for certified deletion based on subspace coset states. Whereas [27] and [3] developed different techniques for constructing, respectively, FE with certified deletion for ciphertexts and FE with key revocation, we construct both primitives from a single technique.

[27] construct functional encryption with certified deletion for *ciphertexts*. One of their main techniques is a way to verify BB84 states by individually signing them with a one-time signature. This serves a similar purpose as our

use of subspace coset states. One difference is that their one-time signature approach results in privately-verifiable certificates of deletion, while our subspace coset state approach also supports public verification when combined with iO. They also construct several primitives with certified deletion that are not considered in our work: compute-and-compare obfuscation and predicate encryption. Likewise, we construct blind delegation, CCA encryption, and (differing-inputs) obfuscation, which they do not consider.

Next, [3] construct functional encryption that supports revocation of *secret keys*. They call this notion *secure key leasing*, which is the same as key revocation, except that their "certificate of deletion" is a privately-verifiable quantum state (the key itself) whereas ours is a classical string obtained by performing a destructive measurement on the quantum key. They also show how to add secure key leasing to various encryption schemes (public-key, identity-based, attribute-based, and functional) while requiring no additional assumptions. We only consider functional encryption with secure key leasing and assume iO, which is implied by subexponentially-secure functional encryption. Our technique is generic and could also be used to add publicly-verifiable secure key leasing to other encryption schemes, at the cost of assuming iO.

Finally, [9] also study revocable cryptography, which is the same notion of secure key leasing as studied by [3]. They obtain various primitives (including psuedorandom functions, PKE, and fully-homomorphic encryption) with key revocation, from the hardness of LWE. The comparison with our work is similar to the previous paragraph: [9] achieve constructions from standard assumptions, but only support privately-verifiable quantum certificates of revocation.

4 Delayed Preparation of Coset States

Here we develop tools for working with subspace coset states that will help us prove Theorem 2, which is our main theorem. In particular, we show how to prepare a random subspace coset state but delay the choice of subspace until after the register has been given out. Similar techniques exist for BB84/Wiesner states, but it is non-trivial to extend them to subspace coset states. Along the way, we develop a framework for representing the cosets of two subspaces $S \leq T$ that maintains the algebraic structure of the quotient groups $\mathbb{F}_2^n/S$ and $\mathbb{F}_2^n/T$. We believe the techniques in this section are interesting independently of their applications to certified deletion.

4.1 Coset Representatives

Given a subspace $S \leq \mathbb{F}_2^n$, let $\mathsf{co}(S)$ be a subspace of $\mathbb{F}_2^n$ that contains exactly one vector from every coset of S.[17] Note, $\mathsf{co}(S)$ is analogous to the quotient group $\mathbb{F}_2^n/S$, whose elements are the actual cosets of S and which has the same

[17] [21] used a different set of coset representatives, called Can_S, which is not necessarily a vector space.

algebraic structure as $\mathsf{co}(S)$. $\mathsf{co}(S)$ is useful for decomposing vectors since every $\mathsf{z} \in \mathbb{F}_2^n$ has a *unique* decomposition as $\mathsf{z} = \mathsf{u} + \mathsf{v}$, for some $(\mathsf{u}, \mathsf{v}) \in S \times \mathsf{co}(S)$ (Lemma 1).

It is useful intuition to imagine choosing $\mathsf{co}(S) = S^\perp$, but this is not always allowed. Because S is a subspace of $\mathbb{F}_2^n$ and not $\mathbb{R}^n$, it's possible that some coset of S contains multiple vectors from $S^\perp$ and another coset contains none.

For any S, there exists a valid $\mathsf{co}(S)$ (see Definition 2 and Lemma 2). Usually, there are many valid choices of $\mathsf{co}(S)$, so we pick one of them to be canonical. To avoid ambiguity, let the description of S include a basis for the following subspaces: $[S, S^\perp, \mathsf{co}(S), \mathsf{co}(S^\perp)]$. This defines the canonical choices for $\mathsf{co}(S)$ and $\mathsf{co}(S^\perp)$.

Lemma 1 shows that the definition of $\mathsf{co}(S)$ is equivalent to some useful properties. In this lemma, we refer to $\mathsf{co}(S)$ as C.

Lemma 1. *For subspaces $S, C \leq \mathbb{F}_2^n$, the following are equivalent:*

1. *C contains exactly one element from each coset of S.*
2. *For any $\mathsf{z} \in \mathbb{F}_2^n$, there is a unique pair $(\mathsf{u}, \mathsf{v}) \in S \times C$ such that $\mathsf{z} = \mathsf{u} + \mathsf{v}$.*
3. *$\dim(C) = n - \dim(S)$ and $\mathsf{span}(S, C) = \mathbb{F}_2^n$.*

4.2 Sampling Procedure

In this section, we give a procedure for choosing $\mathsf{co}(S)$. We actually consider a more-general problem: given two subspaces $S \leq T$, we will choose $[\mathsf{co}(S), \mathsf{co}(S^\perp), \mathsf{co}(T), \mathsf{co}(T^\perp)]$ that satisfy $\mathsf{co}(T) \leq \mathsf{co}(S)$ along with some other useful properties. In later sections, whenever we need to sample two subspaces, $S \leq T$, we will implicitly use Definition 2 to sample the associated coset representatives.

Definition 2 (Procedure to Sample Coset Representatives). *Given two subspaces $S \leq T \leq \mathbb{F}_2^n$:*

1. *Choose n linearly independent vectors $\{\mathsf{z}_1, \ldots, \mathsf{z}_n\}$ uniformly at random such that*

$$S = \mathsf{span}(\mathsf{z}_1, \ldots, \mathsf{z}_{\dim(S)})$$
$$T = \mathsf{span}(\mathsf{z}_1, \ldots, \mathsf{z}_{\dim(T)})$$

2. *Then let*

$$\mathsf{co}(S) = \mathsf{span}(\mathsf{z}_{\dim(S)+1}, \ldots, \mathsf{z}_n)$$
$$\mathsf{co}(T) = \mathsf{span}(\mathsf{z}_{\dim(T)+1}, \ldots, \mathsf{z}_n)$$
$$\mathsf{co}(S^\perp) = \mathsf{co}(S)^\perp$$
$$\mathsf{co}(T^\perp) = \mathsf{co}(T)^\perp$$

3. *Choose a fresh random basis for each subspace in $[\mathsf{co}(S), \mathsf{co}(S^\perp), \mathsf{co}(T), \mathsf{co}(T^\perp)]$, and output these bases. Note that the subspaces do not change in this step – just the bases used to represent them.*

The reason we choose fresh random bases for each subspace is so that someone with a description of S and $\mathsf{co}(S)$ but not T does not learn anything about T other than the fact that $S \leq T$. The original basis we chose for $\mathsf{co}(S)$ was built from the basis for T, which might leak information about T.

Lemma 2 analyzes the procedure in Definition 2 and proves that it satisfies some useful properties.

Lemma 2. *Given two subspaces $S \leq T \leq \mathbb{F}_2^n$, the procedure in Definition 2 chooses the subspaces $[\mathsf{co}(S), \mathsf{co}(S^\perp), \mathsf{co}(T), \mathsf{co}(T^\perp)]$ such that:*

1. *$\mathsf{co}(S)$ is valid: it contains exactly one element from each coset of S. The analogous statement holds for $\mathsf{co}(S^\perp), \mathsf{co}(T), \mathsf{co}(T^\perp)$.*
2. *$\mathsf{co}(T) \leq \mathsf{co}(S)$ and $\mathsf{co}(S^\perp) \leq \mathsf{co}(T^\perp)$.*
3. *$\mathsf{co}(S^\perp) = \mathsf{co}(S)^\perp$ and $\mathsf{co}(T^\perp) = \mathsf{co}(T)^\perp$.*

4.3 Delayed Preparation of Coset States

Our goal in this section is for Alice to prepare a random subspace coset state for Bob, but delay choosing the underlying subspace until after she sends the register to Bob. This technique is used in the proof of the main theorem, Theorem 2, and it uses the formalism for coset representatives that we developed above.

Let Alice be given two subspaces $S \leq T \leq \mathbb{F}_2^n$, and let the corresponding coset representatives $[\mathsf{co}(S), \mathsf{co}(S^\perp), \mathsf{co}(T), \mathsf{co}(T^\perp)]$ be sampled from the procedure in Definition 2. Next, let Alice be given cosets $\mathsf{u} \in \mathsf{co}(T)$ and $\mathsf{w} \in \mathsf{co}(S^\perp)$, which partially determine the subspace coset state that Alice will sample.

Alice will sample the subspace coset state from one of the following distributions:

- Distribution 0: Sample $|S_{\mathsf{v},\mathsf{w}}\rangle$ such that $S + \mathsf{v} \subseteq T + \mathsf{u}$, uniformly at random.
- Distribution 1: Sample $|T_{\mathsf{u},\widetilde{\mathsf{v}}}\rangle$ such that $T^\perp + \widetilde{\mathsf{v}} \subseteq S^\perp + \mathsf{w}$, uniformly at random.

Bob's register will eventually contain the sampled state. But there's a twist: Alice will decide which distribution to sample from *after* she sends Bob her register.

Here is one way for Alice to sample from distribution 0 or 1:

1. To sample from distribution 0, Alice prepares the following state on two n-qubit registers:
$$|\psi\rangle_0 := \frac{1}{\sqrt{|A|}} \sum_{\mathsf{v}_0 \in A} |S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle\, |\mathsf{v}_0\rangle$$
where $A = \mathsf{co}(S) \cap T$. She sends the first register to Bob, and measures the second register in the computational basis.
2. To sample from distribution 1, she prepares the following state:
$$|\psi\rangle_1 := \frac{1}{\sqrt{|B|}} \sum_{\widetilde{\mathsf{v}}_0 \in B} |T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}\rangle\, |\widetilde{\mathsf{v}}_0\rangle$$
where $B = \mathsf{co}(T^\perp) \cap S^\perp$. She sends the first register to Bob, and measures the second register in the computational basis.

Claim. The procedure above correctly samples from distribution 0 or distribution 1.

Proof. In this procedure, we have decomposed v into its deterministic and random components, u and v_0 respectively. Every $\mathsf{v} \in \mathsf{co}(S)$ has a unique decomposition as $\mathsf{v} = \mathsf{u} + \mathsf{v}_0$ for some $\mathsf{u} \in \mathsf{co}(T)$ and $\mathsf{v}_0 \in A$ (see supplementary material). Since v_0 is sampled uniformly at random from A, v is sampled uniformly at random such that $\mathsf{v} \in \mathsf{co}(S)$ and $S + \mathsf{v} \subseteq T + \mathsf{u}$. Therefore, the procedure correctly samples from distribution 0.

A similar argument works for distribution 1. We have decomposed $\widetilde{\mathsf{v}}$ into its deterministic and random components, w and $\widetilde{\mathsf{v}}_0$ respectively. Every $\widetilde{\mathsf{v}} \in \mathsf{co}(T^\perp)$ has a unique decomposition as $\widetilde{\mathsf{v}} = \widetilde{\mathsf{v}}_0 + \mathsf{w}$ for some $\widetilde{\mathsf{v}}_0 \in B$ and $\mathsf{w} \in \mathsf{co}(S^\perp)$ (see supplementary material). Since $\widetilde{\mathsf{v}}_0$ is sampled uniformly at random from B, $\widetilde{\mathsf{v}}$ is sampled uniformly at random such that $\widetilde{\mathsf{v}} \in \mathsf{co}(T^\perp)$ and $T^\perp + \widetilde{\mathsf{v}} \subseteq S^\perp + \mathsf{w}$. Therefore, the procedure correctly samples from distribution 1. □

Next, Alice can map between $|\psi\rangle_0$ and $|\psi\rangle_1$ by applying local operations to the second register. We will define a unitary U that acts on the second register and maps superpositions over to A to superpositions over B. For any $\mathsf{v}_0 \in A$, let

$$U\,|\mathsf{v}_0\rangle = \frac{1}{\sqrt{c}} \sum_{\widetilde{\mathsf{v}}_0 \in B} |\widetilde{\mathsf{v}}_0\rangle \cdot (-1)^{\langle \mathsf{v}_0, \widetilde{\mathsf{v}}_0 \rangle}$$

and for any $\widetilde{\mathsf{v}}_0 \in B$, let

$$U^\dagger\,|\widetilde{\mathsf{v}}_0\rangle = \frac{1}{\sqrt{c}} \sum_{\mathsf{v}_0 \in A} |\mathsf{v}_0\rangle \cdot (-1)^{\langle \mathsf{v}_0, \widetilde{\mathsf{v}}_0 \rangle}$$

where c is a normalization constant. Technically, U acts on any superposition over $\mathbb{F}_2^n$, and we define it fully in the supplementary materials. We show in the supplementary material that when U is applied to the second register of $|\psi\rangle_0$, it maps the state to $|\psi\rangle_1$.

Now we have the tools to do delayed preparation of the subspace coset state:

Delayed Preparation of a Subspace Coset State

1. Alice prepares $|\psi\rangle_0$ on two n-qubit registers. She sends the first register to Bob.
2. (a) To sample from distribution 0: Alice measures the second register in the computational basis to get a random $\mathsf{v}_0 \leftarrow A$. The state on Bob's register collapses to $|S_{\mathsf{u}+\mathsf{v}_0,\mathsf{w}}\rangle$.
   (b) Instead, to sample from distribution 1: Alice applies U to the second register, mapping $|\psi\rangle_0$ to $|\psi\rangle_1$. Then she measures the second register in the computational basis to get a random $\widetilde{\mathsf{v}}_0 \leftarrow B$. The state on Bob's register collapses to $|T_{\mathsf{u},\widetilde{\mathsf{v}}_0+\mathsf{w}}\rangle$.

5 General Compiler for Certified Deletion

In this section, we present a general technique for proving certified deletion that works well with existing cryptographic primitives and enables the constructions in subsequent sections.

Consider the following simple construction. To hide a bit b, we give the adversary:

$$|S_{\mathsf{v},\mathsf{w}}\rangle\,, b \oplus \langle \mathsf{v}, \mathbf{1}\rangle$$

where $|S_{\mathsf{v},\mathsf{w}}\rangle$ is a subspace coset state, sampled uniformly at random such that $\dim(S) = n/2$, $\mathsf{v} \in \mathsf{co}(S), \mathsf{w} \in \mathsf{co}(S^\perp)$. The bit b is masked by $\langle \mathsf{v}, \mathbf{1}\rangle$, and the information needed to remove the mask is stored in the subspace coset state.

To prove deletion, the adversary measures the subspace coset state in the Hadamard basis to get a vector $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$. We'll show that if they prove deletion, then all but negligible information about v is lost. That is, even if S is leaked at a later point, b remains statistically hidden because $\langle \mathsf{v}, \mathbf{1}\rangle$ is statistically close to uniformly random.

Definition 3 below describes this scenario. We say that security holds if the output of EXP^* is statistically close between the cases where $b = 0$ and $b = 1$.

Definition 3 (Certified Deletion Game, Abstract Form). *Let b be a bit, let $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$ be a QPT adversary, and let $\{\mathcal{Z}_\lambda\}_{\lambda\in\mathbb{N}}$ be a quantum or classical operation. Then let $\mathsf{EXP}^*(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda, b)$ be the output of the following experiment:*

1. *Challenge: Let $n = 4\lambda$. The challenger samples a subspace S of dimension $n/2$ along with vectors $(\mathsf{v}, \mathsf{w}) \leftarrow \mathsf{co}(S) \times \mathsf{co}(S^\perp)$, uniformly at random.*
   Next, they sample a subspace T uniformly at random such that $S \leq T$ and $\dim(T) = 3n/4$, using the procedure in Definition 2. Let $\mathsf{u} \in \mathsf{co}(T)$ be the unique coset such that $S + \mathsf{v} \subset T + \mathsf{u}$.
   Finally, the challenger sends the adversary the following challenge:

   $$|S_{\mathsf{v},\mathsf{w}}\rangle\,, \mathcal{Z}_\lambda(S, T, \mathsf{u}, \mathsf{w}, b \oplus \langle \mathsf{v}, \mathbf{1}\rangle)$$

2. *Response: The adversary, running $\mathcal{A}_\lambda$, responds with a deletion certificate $\widetilde{\mathsf{z}} \in \mathbb{F}_2^n$ and an auxiliary state ρ.*
3. *Outcome: The challenger checks that*

   $$\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$$

   If so, they output $(\rho, S, T, \mathsf{u}, \mathsf{w}, b \oplus \langle \mathsf{v}, \mathbf{1}\rangle)$; if not, they output $\perp$.

In Definition 3, $\mathcal{Z}$ represents the side information given to the adversary. In the simplest case, $\mathcal{Z} = \perp$, and it's simple to prove that security holds. Note that we cannot give S, v, or w in the clear because then the adversary could learn v while also outputting a $\widetilde{\mathsf{z}} \in S^\perp + \mathsf{w}$. In all of our applications, we need to give the adversary some information about $(S, \mathsf{v}, \mathsf{w})$, but we're careful to hide it. For instance, when $\mathcal{Z} = \mathsf{Enc}(1^\lambda, S)$ for some semantically-secure encryption scheme Enc, or $\mathcal{Z} = [\mathsf{Enc}(1^\lambda, S), \mathsf{i}\mathcal{O}(P_{S^\perp + \mathsf{w}})]$, we can prove that security holds. The latter case allows the deletion certificate to be *publicly verifiable*, assuming post-quantum $\mathsf{i}\mathcal{O}$.

Theorem 1 (Encryption with Publicly-Verifiable Deletion). *Let* Enc *be a semantically secure encryption scheme, and assume post-quantum indistinguishability obfuscation* ($\mathsf{i}\mathcal{O}$)*. Next, for any* $\lambda \in \mathbb{N}$*, let*

$$\mathcal{Z}_\lambda(S, T, \mathsf{u}, \mathsf{w}, b') = \left[\mathsf{Enc}(1^\lambda, S), \mathsf{i}\mathcal{O}(P_{S^\perp + \mathsf{w}})\right].$$

Then for any QPT adversary $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}}$*,*

$$\mathsf{TD}\left(\mathsf{EXP}^*(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda, 0), \mathsf{EXP}^*(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda, 1)\right) = \mathrm{negl}(\lambda).$$

It is natural to wonder whether we can include $\mathsf{i}\mathcal{O}(P_{S+\mathsf{v}})$ in $\mathcal{Z}$, just like we included $\mathsf{i}\mathcal{O}(P_{S^\perp+\mathsf{w}})$. In fact, this is not allowed because $\mathsf{i}\mathcal{O}$ only hides v computationally. If $\langle \mathsf{v}, \mathbf{1}\rangle$ is not statistically hidden, then neither is b. However, $\mathcal{Z}$ can include (T, u), which satisfy $S + \mathsf{v} \subset T + \mathsf{u}$, and for our applications that is good enough. The bit b remains statistically hidden because even given $(S, T, \mathsf{u}, \mathsf{w})$, $\langle \mathsf{v}, \mathbf{1}\rangle$ is uniformly random (with overwhelming probability over the choice of (S, T)).

5.1 General Theorem

The previous theorems are special cases of Theorem 2 below. We will use it in subsequent sections to prove certified deletion. Theorem 2 says that security holds in EXP^* if any information that $\mathcal{Z}$ gives the adversary about $S^\perp + \mathsf{w}$ could also be computed from a larger random coset $R + \mathsf{x}$ that contains $S^\perp + \mathsf{w}$. We call this property *subspace hiding*, and it is analogous to [43]'s notion of subspace-hiding obfuscation . Below, we will precisely define the property of $\mathcal{Z}$ we need.

Definition 4 (Subspace Hiding). *Let* $\mathscr{A}$ *be a class of adversaries*[18]*. We say that a quantum operation* $\{\mathcal{Z}_\lambda\}_{\lambda\in\mathbb{N}}$ *is* ***subspace-hiding*** *for* $\mathscr{A}$ *if there exists a simulator* $\{\mathcal{S}_\lambda\}_{\lambda\in\mathbb{N}}$ *such that for any adversary in* $\mathscr{A}$*, their advantage in the following game is negligible in* λ*:*

1. *Let* $n = 4\lambda$*. The adversary chooses subspaces* $S, T \leq \mathbb{F}_2^n$ *such that* $S \leq T, \dim(S) = n/2$*, and* $\dim(T) = 3n/4$*, they choose vectors* $\mathsf{u} \in \mathsf{co}(T)$ *and* $\mathsf{w} \in \mathsf{co}(S^\perp)$*, and they choose a bit* b'*. Then they send these variables to the challenger.*
2. *The challenger samples* R*, a uniformly random superspace of* $S^\perp$ *of dimension* $3n/4$*. Let* $\mathsf{x} \in \mathsf{co}(R)$ *be the unique coset such that* $S^\perp + \mathsf{w} \subset R + \mathsf{x}$*. Next, the challenger samples a bit* $c \leftarrow \{0, 1\}$*. If* $c = 0$*, they compute* $\mathcal{Z}_\lambda(S, T, \mathsf{u}, \mathsf{w}, b')$ *and send the output to the challenger. If* $c = 1$*, they compute* $\mathcal{S}_\lambda(R, T, \mathsf{u}, \mathsf{x}, b')$ *and send the output to the challenger.*
3. *The adversary outputs a guess* $c' \in \{0, 1\}$ *for* c*.*

[18] $\mathscr{A}$ should be closed under constant-factor increases in space and time. That is say: for any adversary $\{\mathcal{A}\}_{\lambda\in\mathbb{N}} \in \mathscr{A}$ and any quantum adversary $\{\mathcal{B}\}_{\lambda\in\mathbb{N}}$, if the time and space complexity of $\{\mathcal{B}\}_{\lambda\in\mathbb{N}}$ is more than that of $\{\mathcal{A}\}_{\lambda\in\mathbb{N}}$ by a constant factor, then $\{\mathcal{B}\}_{\lambda\in\mathbb{N}}$ is also in $\mathscr{A}$.

The adversary's advantage is $|\Pr(c' = c) - 1/2|$.

Finally, the theorem below says that b is statistically hidden in EXP^* if $\mathcal{Z}$ is subspace-hiding.

Theorem 2 (General theorem). *Let* $\{\mathcal{Z}_\lambda\}_{\lambda\in\mathbb{N}}$ *be defined as it was in* EXP^**, and let* $\mathscr{A}$ *be a class of adversaries. Next, if* $\{\mathcal{Z}_\lambda\}_{\lambda\in\mathbb{N}}$ *is subspace-hiding for* $\mathscr{A}$*, then for any adversary* $\{\mathcal{A}_\lambda\}_{\lambda\in\mathbb{N}} \in \mathscr{A}$*,*

$$\mathsf{TD}\left(\mathsf{EXP}^*(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda, 0), \mathsf{EXP}^*(1^\lambda, \mathcal{A}_\lambda, \mathcal{Z}_\lambda, 1)\right) = \mathrm{negl}(\lambda)$$

6 Acknowledgments

We thank Sanjam Garg for collaboration and valuable contributions during the earlier stages of this research.

D.K. was supported by DARPA SIEVE, AFOSR, NSF CAREER CNS-2238718 and NSF CNS-2247727. This material is based upon work supported by the Defense Advanced Research Projects Agency through Award HR00112020024, and by the Air Force Office of Scientific Research under award number FA9550-23-1-0543.

G.M. was supported by the European Research Council through an ERC Starting Grant (Grant agreement No. 101077455, ObfusQation).

J.R was supported in part by the NSF award 1916939, DARPA SIEVE program, a gift from Ripple, a DoE NETL award, a JP Morgan Faculty Fellowship, a PNC center for financial services innovation award, and a Cylab seed funding award.

References

1. Scott Aaronson. Quantum copy-protection and quantum money. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 229–242. IEEE Computer Society, 2009.
2. Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '12, page 41–60, New York, NY, USA, 2012. Association for Computing Machinery.
3. Shweta Agrawal, Fuyuki Kitagawa, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Public key encryption with secure key leasing. Cryptology ePrint Archive, Paper 2023/264, 2023. https://eprint.iacr.org/2023/264.
4. Prabhanjan Ananth and Fatih Kaleoglu. Unclonable encryption, revisited. LNCS, pages 299–329. Springer, Heidelberg, 2021.
5. Prabhanjan Ananth and Fatih Kaleoglu. A note on copy-protection from random oracles. Cryptology ePrint Archive, Paper 2022/1109, 2022. https://eprint.iacr.org/2022/1109.
6. Prabhanjan Ananth, Fatih Kaleoglu, Xingjian Li, Qipeng Liu, and Mark Zhandry. On the feasibility of unclonable encryption, and more. CRYPTO, 2022.

7. Prabhanjan Ananth, Fatih Kaleoglu, and Qipeng Liu. Cloning games: A general framework for unclonable primitives. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 66–98, Cham, 2023. Springer Nature Switzerland.
8. Prabhanjan Ananth and Rolando L. La Placa. Secure software leasing. LNCS, pages 501–530. Springer, Heidelberg, 2021.
9. Prabhanjan Ananth, Alexander Poremba, and Vinod Vaikuntanathan. Revocable cryptography from learning with errors. Cryptology ePrint Archive, Paper 2023/325, 2023. https://eprint.iacr.org/2023/325.
10. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012.
11. James Bartusek, Jiaxin Guan, Fermi Ma, and Mark Zhandry. Return of GGH15: Provable security against zeroizing attacks. In Amos Beimel and Stefan Dziembowski, editors, *TCC 2018, Part II*, volume 11240 of *LNCS*, pages 544–574. Springer, Heidelberg, November 2018.
12. James Bartusek and Dakshita Khurana. Cryptography with certified deletion. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023*, pages 192–223, Cham, 2023. Springer Nature Switzerland.
13. Mihir Bellare, Igors Stepanovs, and Brent Waters. New negative results on differing-inputs obfuscation. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 792–821. Springer, Heidelberg, May 2016.
14. Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of the IEEE International Conference on Computers, Systems, and Signal Processing*, pages 175–179, 1984.
15. Elette Boyle, Kai-Min Chung, and Rafael Pass. On extractability obfuscation. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 52–73. Springer, Heidelberg, February 2014.
16. Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and Pairings Are Not Necessary for IO: Circular-Secure LWE Suffices. In Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022)*, volume 229 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:20, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.
17. Anne Broadbent and Rabib Islam. Quantum encryption with certified deletion. LNCS, pages 92–122. Springer, Heidelberg, March 2020.
18. Anne Broadbent, Stacey Jeffery, Sébastien Lord, Supartha Podder, and Aarthi Sundaram. Secure software leasing without assumptions. LNCS, pages 90–120. Springer, Heidelberg, 2021.
19. Yilei Chen, Vinod Vaikuntanathan, and Hoeteck Wee. GGH15 beyond permutation branching programs: Proofs, attacks, and candidates. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992 of *LNCS*, pages 577–607. Springer, Heidelberg, August 2018.
20. Arka Rai Choudhuri, Abhishek Jain, and Zhengzhong Jin. Snargs for $\mathcal{P}$ from LWE. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 68–79. IEEE, 2021.
21. Andrea Coladangelo, Jiahui Liu, Qipeng Liu, and Mark Zhandry. Hidden cosets and applications to unclonable cryptography. LNCS, pages 556–584. Springer, Heidelberg, 2021.

22. Lalita Devadas, Willy Quach, Vinod Vaikuntanathan, Hoeteck Wee, and Daniel Wichs. Succinct LWE sampling, random polynomials, and obfuscation. LNCS, pages 256–287. Springer, Heidelberg, 2021.
23. Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013.
24. Sanjam Garg, Craig Gentry, Shai Halevi, and Daniel Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 518–535. Springer, Heidelberg, August 2014.
25. Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 736–749, New York, NY, USA, 2021. Association for Computing Machinery.
26. Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 169–178. ACM Press, May / June 2009.
27. Taiga Hiroka, Fuyuki Kitagawa, Tomoyuki Morimae, Ryo Nishimaki, Tapas Pal, and Takashi Yamakawa. Certified everlasting secure collusion-resistant functional encryption, and more. Cryptology ePrint Archive, Paper 2023/236, 2023. https://eprint.iacr.org/2023/236.
28. Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Quantum encryption with certified deletion, revisited: Public key, attribute-based, and classical communication. LNCS, pages 606–636. Springer, Heidelberg, 2021.
29. Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Certified everlasting functional encryption. *CoRR*, abs/2207.13878, 2022.
30. Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Certified everlasting zero-knowledge proof for QMA, 2022. Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, Santa Barbara, CA, USA.
31. Jonathan Katz, Aishwarya Thiruvengadam, and Hong-Sheng Zhou. Feasibility and infeasibility of adaptively secure fully homomorphic encryption. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 14–31. Springer, Heidelberg, February / March 2013.
32. Fuyuki Kitagawa and Ryo Nishimaki. Functional encryption with secure key leasing. In *Advances in Cryptology - ASIACRYPT 2022*. Springer Cham, 2022.
33. Fuyuki Kitagawa, Ryo Nishimaki, and Takashi Yamakawa. Secure software leasing from standard assumptions. LNCS, pages 31–61. Springer, Heidelberg, 2021.
34. Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. *ACM Symposium on Theory of Computing*, 03 2001.
35. Alexander Poremba. Quantum proofs of deletion for learning with errors. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 90:1–90:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.
36. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

37. Omri Shmueli. Semi-quantum tokenized signatures. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part I*, volume 13507 of *Lecture Notes in Computer Science*, pages 296–319. Springer, 2022.
38. Dominique Unruh. Revocable quantum timed-release encryption. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 129–146. Springer, Heidelberg, May 2014.
39. Thomas Vidick and Tina Zhang. Classical proofs of quantum knowledge. LNCS, pages 630–660. Springer, Heidelberg, 2021.
40. Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious LWE sampling. LNCS, pages 127–156. Springer, Heidelberg, 2021.
41. Stephen Wiesner. Conjugate coding. *SIGACT News*, 15:78–88, 1983.
42. W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, Oct 1982.
43. Mark Zhandry. Quantum lightning never strikes the same state twice. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 408–438. Springer, Heidelberg, May 2019.