

# Publicly Verifiable Deletion from Minimal Assumptions

Fuyuki Kitagawa<sup>†</sup>, Ryo Nishimaki<sup>†</sup>, Takashi Yamakawa<sup>†</sup>

<sup>†</sup>NTT Social Informatics Laboratories, Tokyo, Japan  
{fuyuki.kitagawa,ryo.nishimaki,takashi.yamakawa}@ntt.com

April 17, 2023

## Abstract

We present a general compiler to add the publicly verifiable deletion property for various cryptographic primitives including public key encryption, attribute-based encryption, and quantum fully homomorphic encryption. Our compiler only uses one-way functions, or more generally hard quantum planted problems for NP, which are implied by one-way functions. It relies on minimal assumptions and enables us to add the publicly verifiable deletion property with no additional assumption for the above primitives. Previously, such a compiler needs additional assumptions such as injective trapdoor one-way functions or pseudorandom group actions [Bartusek-Khurana-Poremba, ePrint:2023/370]. Technically, we upgrade an existing compiler for privately verifiable deletion [Bartusek-Khurana, ePrint:2022/1178] to achieve publicly verifiable deletion by using digital signatures.

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Background . . . . .	3
1.2	Our Results . . . . .	4
1.3	Technical Overview . . . . .	4
1.4	More on Related Works . . . . .	6
<b>2</b>	<b>Preliminaries</b>	<b>7</b>
2.1	Notations . . . . .	7
2.2	Cryptographic Tools . . . . .	7
<b>3</b>	<b>Signature with One-Time Unforgeability for BB84 states</b>	<b>8</b>
<b>4</b>	<b>Certified Everlasting Lemmas</b>	<b>9</b>
<b>5</b>	<b>Instantiations</b>	<b>10</b>
5.1	Public-Key Encryption . . . . .	10
5.2	Other Primitives . . . . .	13
<b>6</b>	<b>Making Assumptions Minimal</b>	<b>13</b>

# 1 Introduction

## 1.1 Background

Quantum mechanics yields new cryptographic primitives that classical cryptography cannot achieve. In particular, the uncertainty principle enables us to certify deletion of information. Broadbent and Islam [BI20] introduced the notion of quantum encryption with certified deletion, where we can generate a classical certificate for the deletion of quantum ciphertext. We need a verification key generated along with a quantum ciphertext to check the validity of a certificate. The root of this concept is revocable quantum time-release encryption by Unruh [Unr15], where a sender can revoke quantum ciphertext if a receiver returns it before a pre-determined time. Encryption with certified deletion is useful because encryption security holds *even if adversaries obtain a secret decryption key after they generate a valid certificate for deletion*. After the work by Broadbent and Islam [BI20], many works presented extended definitions and new constructions of quantum (advanced) encryption with certified deletion [HMNY21, Por23, BK22, HKM<sup>+</sup>23, BGG<sup>+</sup>23, BKP23]. In particular, Bartusek and Khurana [BK22], and Hiroka, Kitagawa, Morimae, Nishimaki, Pal, and Yamakawa [HKM<sup>+</sup>23] considered certified everlasting security, which guarantees that computationally *unbounded* adversaries cannot obtain information about plaintext *after a valid certificate was generated*. Several works [HMNY21, Por23, BGG<sup>+</sup>23, BKP23] considered public verifiability, where we can reveal verification keys without harming certified deletion security. These properties are desirable for encryption with certified deletion in real-world applications.

In this work, we focus on cryptographic primitives with publicly verifiable deletion (PVD) [BKP23], which satisfy certified everlasting security and public verifiability. Known schemes based on BB84 states are privately verifiable [BI20, HMNY21, BK22, HKM<sup>+</sup>23], where we need to keep verification keys secret to ensure encryption security. Some schemes are publicly verifiable [HMNY21, Por23, BGG<sup>+</sup>23, BKP23]. Public verifiability is preferable to private verifiability since we need to keep many verification keys secret when we generate many ciphertexts of encryption with privately verifiable deletion. More secret keys lead to more risks. In addition, anyone can verify deletion in cryptography with PVD.

Hiroka, Morimae, Nishimaki, and Yamakawa [HMNY21] achieved interactive public key encryption with non-everlasting PVD from extractable witness encryption [GKP<sup>+</sup>13], which is a strong knowledge-type assumption, and one-shot signatures which require classical oracles [AGKZ20]. Poremba [Por23] achieved public key encryption (PKE) and fully homomorphic encryption (FHE) with non-everlasting PVD based on lattices. He conjectured Ajtai hash function satisfies a strong Gaussian collapsing property and proved the security of his constructions under the conjecture. Later, Bartusek, Khurana, and Poremba [BKP23] proved the conjecture is true under the LWE assumption.

Bartusek, Garg, Goyal, Khurana, Malavolta, Raizes, and Roberts [BGG<sup>+</sup>23] achieved primitive  $X$  with PVD from  $X$  and post-quantum secure indistinguishability obfuscation (IO) [BGI<sup>+</sup>12] where  $X \in \{\text{SKE, COM, PKE, ABE, FHE, TRE, WE}\}$ .<sup>1</sup> They also achieved functional encryption with PVD and obfuscation with PVD, which rely on post-quantum IO. All their constructions use subspace coset states [CLLZ21]. Bartusek et al. [BKP23] achieved PKE (resp. COM) with PVD from injective trapdoor one-way functions (or superposition-invertible regular trapdoor functions) or pseudorandom group actions [HMY22] (resp. almost-regular one-way functions). They also achieved primitive  $Y$  with PVD from injective trapdoor one-way functions (or superposition-invertible regular trapdoor functions) or pseudorandom group actions, and primitive  $Y$  where  $Y \in \{\text{ABE, QFHE, TRE, WE}\}$ . They obtained these results by considering certified everlasting target-collapsing functions as an intermediate primitive.

As we explained above, known encryption with PVD constructions need strong and non-standard assumptions [HMNY21, BGG<sup>+</sup>23], algebraic assumptions [Por23, BKP23], or additional assumptions [BKP23]. This status is unsatisfactory because Bartusek and Khurana [BK22] prove that we can achieve  $X$  with *privately* verifiable deletion<sup>2</sup> from  $X$  where  $X \in \{\text{SKE, COM, PKE, ABE, FHE, TRE, WE}\}$ <sup>3</sup> *without any additional assumptions*. Thus, our main question in this work is the following.

*Can we achieve encryption with PVD from minimal assumptions?*

---

<sup>1</sup>SKE, COM, ABE, TRE, and WE stand for secret key encryption, commitment, attribute-based encryption, time-release encryption, and witness encryption, respectively. Although Bartusek et al. [BGG<sup>+</sup>23] did not mention, we can apply their transformation to SKE and COM as the results by Bartusek and Khurana [BK22].

<sup>2</sup>We do not abbreviate when we refer to this type to avoid confusion.

<sup>3</sup>Although Bartusek and Khurana [BK22] did not mention, we can apply their transformation to SKE.

## 1.2 Our Results

We affirmatively answer the main question described in the previous section. We present a general transformation from primitive  $Z$  into  $Z$  with PVD where  $Z \in \{\text{SKE, COM, PKE, ABE, QFHE, TRE, WE}\}$ . In the transformation, we only use one-way functions, or more generally, hard quantum planted problems for NP, which we introduce in this work and are implied by one-way functions.

More specifically, we extend the certified everlasting lemma by Bartusek and Khurana [BK22], which enables us to achieve encryption with privately verifiable deletion, to a publicly verifiable certified everlasting lemma. We develop an authentication technique based on (a variant of) the Lamport signature [Lam79] to reduce our publicly verifiable certified everlasting lemma to Bartusek and Khurana’s certified everlasting lemma.

Our new lemma is almost as versatile as Bartusek and Khurana’s lemma. It is easy to apply it to all-or-nothing type encryption<sup>4</sup>, where a secret key (or witness) holder can recover the entire plaintext. One subtle issue is that we need to use QFHE for (Q)FHE with PVD. The reason is that we need to apply an evaluation algorithm to quantum ciphertext. Note that we can use FHE and Bartusek and Khurana’s lemma to achieve FHE with privately verifiable deletion.

The advantages of our techniques are as follows:

- For  $Z' \in \{\text{SKE, COM, PKE, ABE, QFHE, TRE}\}$ , we can convert plain  $Z'$  into  $Z'$  with PVD with no additional assumption. For WE, we can convert it into WE with PVD additionally assuming one-way functions (or hard quantum planted problems for NP).<sup>5</sup> Bartusek et al. [BKP23] require injective (or almost-regular) trapdoor one-way functions, pseudorandom group actions, or almost-regular one-way functions for their constructions.
- Our transformation is applicable even if the base scheme has quantum encryption and decryption (or committing) algorithms.<sup>6</sup> This means that our assumptions are minimal since  $Z$  with PVD implies both plain  $Z$  with quantum encryption and decryption (or committing) algorithms and hard quantum planted problems for NP for  $Z \in \{\text{SKE, COM, PKE, ABE, QFHE, TRE, WE}\}$ .
- Our approach is simple. Bartusek et al. [BKP23] introduced an elegant intermediate notion, certified everlasting target-collapsing, to achieve cryptography with PVD. However, they use a few more intermediate notions (such as balanced binary-measurement target-collision-resistance) for their approach.

## 1.3 Technical Overview

As explained in Section 1.1, Bartusek and Khurana [BK22] gave a generic compiler to add the *privately verifiable* deletion property for various types of encryption. Our finding is that there is a surprisingly simple way to upgrade their compiler to achieve *publicly verifiable* deletion by additionally using digital signatures.

**Notations.** For a bit string  $x$ , we write  $x_j$  to mean  $j$ -th bit of  $x$ . For bit strings  $x, \theta \in \{0, 1\}^\ell$ , we write  $|x\rangle_\theta$  to mean the BB84 state  $\bigotimes_{j \in [\ell]} H^{\theta_j} |x_j\rangle$  where  $H$  is the Hadamard operator.

**Certified everlasting lemma of [BK22].** The compiler of [BK22] is based on the *certified everlasting lemma* described below.<sup>7</sup> We describe it in the dual version, where the roles of computational and Hadamard bases are swapped for convenience. We stress that the dual version is equivalent to the original one because they coincide under an appropriate basis change.

Consider a family of distributions  $\{\mathcal{Z}(m)\}_{m \in \{0,1\}^{\lambda+1}}$  over classical strings, such that for any  $m \in \{0,1\}^{\lambda+1}$ , the distribution  $\mathcal{Z}(m)$  is computationally indistinguishable from the distribution  $\mathcal{Z}(0^{\lambda+1})$ . In other words, each distribution  $\mathcal{Z}(m)$  can be thought of as an “encryption” of the input  $m$ . Let  $\tilde{\mathcal{Z}}(b)$  be an experiment between an adversary and challenger defined as follows for  $b \in \{0, 1\}$ :

<sup>4</sup>SKE, PKE, ABE, (Q)FHE, TRE, and WE fall into this category.

<sup>5</sup>WE does not seem to imply one-way functions.

<sup>6</sup>The compilers of [BK22, BKP23] are also applicable to schemes that have quantum encryption and decryption (or committing) algorithms though they do not explicitly mention it.

<sup>7</sup>For simplicity, we state a simplified version of the lemma that is sufficient for the conversion for PKE, FHE, TRE, and WE, but not for ABE. See Lemma 4.1 for the general version.

- The challenger samples  $x, \theta \leftarrow \{0, 1\}^\lambda$  and sends  $|x\rangle_\theta$  and  $\mathcal{Z}(\theta, b \oplus \bigoplus_{j:\theta_j=1} x_j)$  to the adversary.
- The adversary sends a classical string  $x' \in \{0, 1\}^\lambda$  and a quantum state  $\rho$  to the challenger.
- The challenger outputs  $\rho$  if  $x'_j = x_j$  for all  $j$  such that  $\theta_j = 0$ , and otherwise outputs a special symbol  $\perp$  as the output of the experiment.

The certified everlasting lemma states that for any QPT adversary, the trace distance between  $\tilde{\mathcal{Z}}(0)$  and  $\tilde{\mathcal{Z}}(1)$  is negligible in  $\lambda$ .

The above lemma immediately gives a generic compiler to add the privately verifiable deletion property. To encrypt a message  $b \in \{0, 1\}$ , we set the ciphertext to be  $(|x\rangle_\theta, \text{Enc}(\theta, b \oplus \bigoplus_{j:\theta_j=1} x_j))$ , where  $x$  and  $\theta$  are randomly chosen from  $\{0, 1\}^\lambda$ , and  $\text{Enc}$  is the encryption algorithm of the underlying scheme.<sup>8</sup> The decryptor first decrypts the second component to get  $(\theta, b \oplus \bigoplus_{j:\theta_j=1} x_j)$ , recovers  $\bigoplus_{j:\theta_j=1} x_j$  from  $|x\rangle_\theta$  and  $\theta$ , and then XORs it with  $b \oplus \bigoplus_{j:\theta_j=1} x_j$  to recover  $b$ . To delete the ciphertext and obtain a certificate  $x'$ , we measure  $|x\rangle_\theta$  in the standard basis. To verify the certificate, we check if  $x'_j = x_j$  for all  $j$  such that  $\theta_j = 0$ . By utilizing the above lemma, we can see that an adversary's internal state will not contain any information about  $b$  given the verification algorithm's acceptance. Therefore, the scheme offers certified everlasting security. However, this scheme has only privately verifiable deletion because the verification algorithm needs to know  $x$  and  $\theta$ , which are part of the encryption randomness that has to be hidden from the adversary.

**Making verification public via digital signatures.** We show a publicly verifiable variant of the certified everlasting lemma by using digital signatures. Roughly speaking, our idea is to generate a signature for the BB84 state  $|x\rangle_\theta$  by coherently running the signing algorithm so that the verification of deletion can be done by the verification of the signature, which can be done publicly. Note that the signature does *not* certify  $|x\rangle_\theta$  as a quantum state. It rather certifies its computational basis part (i.e.,  $x_j$  for  $j$  such that  $\theta_j = 0$ ). This is sufficient for our purpose because the verification of deletion just checks the computational basis part.

With the above idea in mind, we modify the experiment  $\tilde{\mathcal{Z}}(b)$  as follows:

- The challenger generates a key pair  $(\text{vk}, \text{sigk})$  of a digital signature scheme and samples  $x, \theta \leftarrow \{0, 1\}^\lambda$ . Let  $U_{\text{sign}}$  be a unitary that works as follows:

$$|m\rangle |0 \dots 0\rangle \mapsto |m\rangle |\text{Sign}(\text{sigk}, m)\rangle.$$

where  $\text{Sign}(\text{sigk}, \cdot)$  is a deterministic signing algorithm with a signing key  $\text{sigk}$ . The challenger sends  $\text{vk}$ ,  $U_{\text{sign}} |x\rangle_\theta |0 \dots 0\rangle$ , and  $\mathcal{Z}(\text{sigk}, \theta, b \oplus \bigoplus_{j:\theta_j=1} x_j)$  to the adversary.

- The adversary sends a classical string  $x' \in \{0, 1\}^\lambda$ , a signature  $\sigma$ , and a quantum state  $\rho$  to the challenger.
- The challenger outputs  $\rho$  if  $\sigma$  is a valid signature for  $x'$ , and otherwise outputs a special symbol  $\perp$  as the output of the experiment.

We show that for any QPT adversary, the trace distance between  $\tilde{\mathcal{Z}}(0)$  and  $\tilde{\mathcal{Z}}(1)$  is negligible in  $\lambda$  if we instantiate the digital signatures with an appropriate scheme as explained later. The crucial difference from the original lemma is that the challenger does not need to check if  $x'_j = x_j$  for all  $j$  such that  $\theta_j = 0$  and only needs to run the verification algorithm of the digital signature scheme.

By using the above variant similarly to the original privately verifiable construction, we obtain a generic compiler that adds publicly verifiable deletion property. For clarity, we describe the construction below. To encrypt a message  $b \in \{0, 1\}$ , the encryption algorithm generates a key pair  $(\text{vk}, \text{sigk})$  of the digital signature scheme, chooses  $x, \theta \leftarrow \{0, 1\}^\lambda$ , and outputs a ciphertext  $(U_{\text{sign}} |x\rangle_\theta, \text{Enc}(\text{sigk}, \theta, b \oplus \bigoplus_{j:\theta_j=1} x_j))$  and a public verification key  $\text{vk}$ . The decryptor first decrypts the second component to get  $(\text{sigk}, \theta, b \oplus \bigoplus_{j:\theta_j=1} x_j)$ , uncompute the signature register of the first component by using  $\text{sigk}$  to get  $|x\rangle_\theta$ , recovers  $\bigoplus_{j:\theta_j=1} x_j$  from  $|x\rangle_\theta$  and  $\theta$ , and then XORs it with  $b \oplus \bigoplus_{j:\theta_j=1} x_j$

---

<sup>8</sup>We write  $\text{Enc}(\theta, b \oplus \bigoplus_{j:\theta_j=1} x_j)$  to mean an encryption of the message  $(\theta, b \oplus \bigoplus_{j:\theta_j=1} x_j)$  where we omit the encryption key.

to recover  $b$ . To delete the ciphertext and obtain a certificate  $(x', \sigma)$ , we measure  $U_{\text{sign}} |x\rangle_\theta$  in the standard basis to get  $(x', \sigma)$ . To verify the certificate, we check if  $\sigma$  is a valid signature for  $x'$ . By utilizing the above lemma, we can see that the above scheme achieves certified everlasting security with public verification.

**Proof idea and instantiation of digital signatures.** We prove the above publicly verifiable version by reducing it to the original one in [BK22]. Noting that  $\mathcal{Z}(\text{sigk}, \theta, b \oplus \bigoplus_{j:\theta_j=1} x_j)$  computationally hides sigk by the assumption, a straightforward reduction works if the digital signature scheme satisfies the following security notion which we call one-time unforgeability for BB84 states.

**Definition 1.1 (One-time unforgeability for BB84 states (informal)).** *Given  $vk$  and  $U_{\text{sign}} |x\rangle_\theta |0 \dots 0\rangle$  for uniformly random  $x, \theta \leftarrow \{0, 1\}^\lambda$ , no QPT adversary can output  $x' \in \{0, 1\}^\lambda$  and a signature  $\sigma$  such that  $\sigma$  is a valid signature for  $x'$  and  $x'_j \neq x_j$  for some  $j$  such that  $\theta_j = 0$  with a non-negligible probability.*

It is easy to show that the Lamport signature satisfies the above property. This can be seen as follows. Recall that a verification key of the Lamport signature consists of  $(v_{j,b})_{j \in [\lambda], b \in \{0,1\}}$  where  $v_{j,b} := f(u_{j,b})$  for a one-way function  $f$  and uniformly random inputs  $(u_{j,b})_{j \in [\lambda], b \in \{0,1\}}$ , and a signature for a message  $m \in \{0, 1\}^\lambda$  is  $\sigma := (u_{j,m_j})_{j \in [\lambda]}$ . Suppose that there is an adversary that breaks the above property. Then, there must exist  $j \in [\lambda]$  such that  $x'_j \neq x_j$  and  $\theta_j = 0$ , in which case the input state  $U_{\text{sign}} |x\rangle_\theta |0 \dots 0\rangle$  does not have any information of  $u_{j,1-x_j}$ . On the other hand, to generate a valid signature for  $x'$ , the adversary has to find a preimage of  $v_{j,x'_j} = v_{j,1-x_j}$ . This is impossible by the one-wayness of  $f$ . Thus, a digital signature scheme that satisfies one-time unforgeability for BB84 states exists assuming the existence of one-way functions.

**Achieving minimal assumptions.** In the above, we explain that one-way functions are sufficient for instantiating the digital signature scheme needed for our compiler. On the other hand, encryption schemes with publicly verifiable deletion does not seem to imply the existence of one-way functions because ciphertexts can be quantum. Thus, one-way functions may not be a necessary assumption for them. To weaken the assumption, we observe that we can use hard quantum planted problems for NP instead of one-way functions in the Lamport signature. Here, hard quantum planted problems for a NP language  $L$  is specified by a quantum polynomial-time sampler that samples an instance-witness pair  $(x, w)$  for the language  $L$  in such a way that no QPT adversary can find a witness for  $x$  with non-negligible probability. Given such a sampler, it is easy to see that we can instantiate the digital signature scheme similar to the above except that  $(v_{j,b}, u_{j,b})$  is now replaced with an instance-witness pair sampled by the sampler.

We observe that  $Z$  with PVD implies the existence of hard quantum planted problems for NP where  $Z \in \{\text{SKE}, \text{COM}, \text{PKE}, \text{ABE}, \text{QFHE}, \text{TRE}, \text{WE}\}$ .<sup>9</sup> This can be seen by considering the verification key as an instance and certificate as a witness for an NP language. Our construction relies on hard quantum planted problems for NP and plain  $Z$  with quantum encryption and decryption (or committing) algorithms, both of which are implied by  $Z$  with PVD, and thus it is based on the minimal assumptions.

## 1.4 More on Related Works

**Certified deletion for ciphertext with private verifiability.** Broadbent and Islam [BI20] achieved one-time SKE with privately verifiable deletion without any cryptographic assumptions. Hiroka et al. [HMNY21] achieved PKE and ABE with non-everlasting privately verifiable deletion. They also achieve interactive PKE with non-everlasting privately verifiable deletion and classical communication in the quantum random oracle model (QROM) from the LWE assumption. However, none of their constructions satisfy certified everlasting security. Hiroka, Morimae, Nishimaki, and Yamakawa [HMNY22] defined certified everlasting commitment and zero-knowledge (for QMA) by extending everlasting security [MU10]. They achieved these notions by using plain commitment and QROM. Bartusek and Khurana [BK22] defined certified everlasting security for various all-or-nothing type encryption primitives and presented a general framework to achieve certified everlasting secure all-or-nothing type encryption and commitment

<sup>9</sup>We assume that the verification algorithm of  $Z$  with PVD is a classical deterministic algorithm. If we allow it to be a quantum algorithm, we have to consider hard quantum planted problems for QCMA, which are also sufficient to instantiate our compiler.

with privately verifiable deletion using BB84 states. They also studied secure computation with everlasting security transfer. Hiroka et al. [HKM<sup>+</sup>23] also defined and achieved certified everlasting security for various cryptographic primitives. In particular, they defined and achieved certified everlasting (collusion-resistant) functional encryption, garbled circuits, and compute-and-compare obfuscation, which are outside of all-or-nothing type encryption. Their constructions are also based on BB84 states and are privately verifiable. They also use a signature-based authentication technique for BB84 states. However, its role is not achieving public verifiability but functional encryption security. See the paper by Hiroka et al. [HKM<sup>+</sup>23] for the differences between their results and Bartusek and Khurana’s results [BK22].

**Certified deletion for keys or secure key leasing.** Kitagawa and Nishimaki [KN22] defined secret key functional encryption with secure key leasing, where we can generate a classical certificate for the deletion of functional keys. They achieved such a primitive with bounded collusion-resistance from one-way functions. Agrawal, Kitagawa, Nishimaki, Yamada, and Yamakawa [AKN<sup>+</sup>23] defined PKE with secure key leasing, where we lose decryption capability after we return a quantum decryption key. They achieved it from standard PKE. They also extended the notion to ABE with secure key leasing and public key functional encryption with secure key leasing. They achieved them from standard ABE and public key functional encryption, respectively. Garg et al. [BGG<sup>+</sup>23] presented a public key functional encryption with secure key leasing scheme based on IO and injective one-way functions. Ananth, Poremba, and Vaikuntanathan [APV23] also defined the same notion as PKE with secure key leasing (they call key-revocable PKE). However, their definition differs slightly from that of Agrawal et al. [AKN<sup>+</sup>23]. They also studied key-revocable FHE and PRF. They achieved them from the LWE assumption.

**Secure software leasing.** Ananth and La Placa [AL21] defined secure software leasing, where we lose software functionality after we return a quantum software. They achieved secure software leasing for a subclass of evasive functions from public key quantum money and the LWE assumptions. After that, several works proposed extensions, variants, and improved constructions of secure software leasing [CMP20, BJJ<sup>+</sup>21, KNY21, ALL<sup>+</sup>21].

## 2 Preliminaries

### 2.1 Notations

Here we introduce basic notations we will use in this paper.

In this paper, standard math or sans serif font stands for classical algorithms (e.g.,  $C$  or  $\text{Gen}$ ) and classical variables (e.g.,  $x$  or  $\text{pk}$ ). Calligraphic font stands for quantum algorithms (e.g.,  $\mathcal{G}$ ) and calligraphic font and/or the bracket notation for (mixed) quantum states (e.g.,  $q$  or  $|\psi\rangle$ ).

Let  $x \leftarrow X$  denote selecting an element  $x$  from a finite set  $X$  uniformly at random, and  $y \leftarrow A(x)$  denote assigning to  $y$  the output of a quantum or probabilistic or deterministic algorithm  $A$  on an input  $x$ . When  $D$  is a distribution,  $x \leftarrow D$  denotes sampling an element  $x$  from  $D$ .  $y := z$  denotes that  $y$  is set, defined, or substituted by  $z$ . Let  $[n] := \{1, \dots, n\}$ . Let  $\lambda$  be a security parameter. For a bit string  $s \in \{0, 1\}^n$ ,  $s_i$  denotes the  $i$ -th bit of  $s$ . QPT stands for quantum polynomial time. PPT stands for (classical) probabilistic polynomial time. We say that a quantum (resp. probabilistic classical) algorithm is efficient if it runs in QPT (resp. PPT). A function  $f : \mathbb{N} \rightarrow \mathbb{R}$  is a negligible function if for any constant  $c$ , there exists  $\lambda_0 \in \mathbb{N}$  such that for any  $\lambda > \lambda_0$ ,  $f(\lambda) < \lambda^{-c}$ . We write  $f(\lambda) \leq \text{negl}(\lambda)$  to denote  $f(\lambda)$  being a negligible function. The trace distance between two quantum states  $\rho$  and  $\sigma$  denoted as  $\text{TD}(\rho, \sigma)$  is given by  $\frac{1}{2} \|\rho - \sigma\|_{\text{tr}}$ , where  $\|A\|_{\text{tr}} := \text{tr} \sqrt{A^\dagger A}$  is the trace norm.

### 2.2 Cryptographic Tools

In this section, we review the cryptographic tools used in this paper.

**Definition 2.1 (Signature).** Let  $\mathcal{M}$  be a message space. A signature scheme for  $\mathcal{M}$  is a tuple of efficient algorithms  $(\text{Gen}, \text{Sign}, \text{Vrfy})$  where:

$\text{Gen}(1^\lambda) \rightarrow (\text{vk}, \text{sigk})$ : The key generation algorithm takes as input the security parameter  $1^\lambda$  and outputs a verification key  $\text{vk}$  and a signing key  $\text{sigk}$ .

$\text{Sign}(\text{sigk}, m) \rightarrow \sigma$ : The signing algorithm takes as input a signing key  $\text{sigk}$  and a message  $m \in \mathcal{M}$  and outputs a signature  $\sigma$ .

$\text{Vrfy}(\text{vk}, m, \sigma) \rightarrow \top$  or  $\perp$ : The verification algorithm is a deterministic algorithm that takes as input a verification key  $\text{vk}$ , a message  $m$  and a signature  $\sigma$  and outputs  $\top$  to indicate acceptance of the signature and  $\perp$  otherwise.

**Correctness:** For all  $m \in \mathcal{M}$ ,  $(\text{vk}, \text{sigk})$  in the range of  $\text{Gen}(1^\lambda)$ , and  $\sigma \in \text{Sign}(\text{sigk}, m)$ , we have  $\text{Vrfy}(\text{vk}, m, \sigma) = \top$ .

**Definition 2.2 (Deterministic Signature).** We say that a signature scheme  $\text{SIG} = (\text{Gen}, \text{Sign}, \text{Vrfy})$  is a deterministic signature if  $\text{Sign}(\text{sigk}, \cdot)$  is a deterministic function.

**Definition 2.3 (Public Key Encryption).** A PKE scheme is a tuple of efficient algorithms  $\text{PKE} = (\text{Gen}, \text{Enc}, \text{Dec})$ .

$\text{Gen}(1^\lambda) \rightarrow (\text{pk}, \text{sk})$ : The key generation algorithm takes the security parameter  $1^\lambda$  as input and outputs a public key  $\text{pk}$  and a secret key  $\text{sk}$ .

$\text{Enc}(\text{pk}, m) \rightarrow \text{ct}$ : The encryption algorithm takes  $\text{pk}$  and a plaintext  $m \in \{0, 1\}$  as input, and outputs a ciphertext  $\text{ct}$ .

$\text{Dec}(\text{sk}, \text{ct}) \rightarrow m'$ : The decryption algorithm takes  $\text{sk}$  and  $\text{ct}$  as input, and outputs a plaintext  $m' \in \mathcal{M}$  or  $\perp$ .

**Correctness:** For any  $m \in \mathcal{M}$ , we have

$$\Pr \left[ m' \neq m \mid \begin{array}{l} (\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda) \\ \text{ct} \leftarrow \text{Enc}(\text{pk}, m) \\ m' \leftarrow \text{Dec}(\text{sk}, \text{ct}) \end{array} \right] \leq \text{negl}(\lambda).$$

**Semantic Security:** For any QPT  $\mathcal{A}$ , we have

$$\left| \Pr \left[ \mathcal{A}(\text{pk}, \text{ct}) = 1 \mid \begin{array}{l} (\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda) \\ \text{ct} \leftarrow \text{Enc}(\text{pk}, 0) \end{array} \right] - \Pr \left[ \mathcal{A}(\text{pk}, \text{ct}) = 1 \mid \begin{array}{l} (\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda) \\ \text{ct} \leftarrow \text{Enc}(\text{pk}, 1) \end{array} \right] \right| = \text{negl}(\lambda).$$

### 3 Signature with One-Time Unforgeability for BB84 states

**Definition.** We first provide the definition of one-time unforgeability for BB84 states.

**Definition 3.1 (One-Time Unforgeability for BB84 states).** Let  $\text{SIG} = (\text{Gen}, \text{Sign}, \text{Vrfy})$  be a signature scheme. We define the experiment  $\text{Exp}_{\text{SIG}, \mathcal{A}}^{\text{otu-bb84}}(1^\lambda)$  between an adversary  $\mathcal{A}$  and challenger as follows.

1. The challenger runs  $(\text{vk}, \text{sigk}) \leftarrow \text{Gen}(1^\lambda)$  and generates  $x, \theta \leftarrow \{0, 1\}^\lambda$ . The challenger generates a quantum state  $|\psi\rangle$  by applying the map  $|m\rangle |0 \dots 0\rangle \rightarrow |m\rangle |\text{Sign}(\text{sigk}, m)\rangle$  to  $|x\rangle_\theta \otimes |0 \dots 0\rangle$ . The challenger gives  $\text{vk}$  and  $|\psi\rangle$  to  $\mathcal{A}$ .
2.  $\mathcal{A}$  outputs a pair of message and signature  $(x', \sigma')$  as the challenge message-signature pair to the challenger.
3. The experiment outputs 1 if the followings are satisfied.
  - $x_i \neq x'_i$  holds for some  $i$  such that  $\theta_i = 0$ .
  - $\text{Vrfy}(\text{vk}, x', \sigma') = 1$  holds.

We say  $\text{SIG}$  is one-time unforgeable for BB84 states if, for any QPT adversary  $\mathcal{A}$ , it holds that

$$\text{Adv}_{\text{SIG}, \mathcal{A}}^{\text{otu-bb84}}(\lambda) := \Pr \left[ \text{Exp}_{\text{SIG}, \mathcal{A}}^{\text{otu-bb84}}(1^\lambda) = 1 \right] = \text{negl}(\lambda).$$

**Construction.** We construct a classical deterministic signature scheme  $\text{SIG} = (\text{Gen}, \text{Sign}, \text{Vrfy})$  satisfying one-time unforgeability for BB84 states using OWF  $f$ . The message space of  $\text{SIG}$  is  $\{0, 1\}^\ell$ .

$\text{Gen}(1^\lambda)$ :

- Generate  $u_{i,b} \leftarrow \{0, 1\}^\lambda$  for every  $i \in [\ell]$  and  $b \in \{0, 1\}$ .
- Compute  $v_{i,b} \leftarrow f(u_{i,b})$  for every  $i \in [\ell]$  and  $b \in \{0, 1\}$ .
- Output  $\text{vk} := (v_{i,b})_{i \in [\ell], b \in \{0,1\}}$  and  $\text{sigk} := (u_{i,b})_{i \in [\ell], b \in \{0,1\}}$ .

$\text{Sign}(\text{sigk}, m)$ :

- Parse  $(u_{i,b})_{i \in [\ell], b \in \{0,1\}} \leftarrow \text{sigk}$ .
- Output  $(u_{i,m_i})_{i \in [\ell]}$ .

$\text{Vrfy}(\text{vk}, m, \sigma)$ :

- Parse  $(v_{i,b})_{i \in [\ell], b \in \{0,1\}} \leftarrow \text{vk}$  and  $(w_i)_{i \in [\ell]} \leftarrow \sigma$ .
- Output  $\top$  if  $u_{i,m_i} = f(w_i)$  holds for every  $i \in [\ell]$  and otherwise output  $\perp$ .

It is clear that  $\text{SIG}$  is a correct classical deterministic signature. Also, we have the following theorem.

**Theorem 3.2.** *Assume  $f$  is OWF. Then,  $\text{SIG}$  satisfies one-time unforgeability for BB84 states.*

*Proof.* For a computational basis position  $i \in [\lambda]$  (that is,  $i$  such that  $\theta_i = 0$ ), when we generate  $|\psi\rangle$  by applying the map  $|m\rangle |0 \dots 0\rangle \rightarrow |m\rangle |\text{Sign}(\text{sigk}, m)\rangle$  to  $|x\rangle_\theta \otimes |0 \dots 0\rangle$ ,  $|\psi\rangle$  contains only  $u_{i,b}$  and not  $u_{i,1-b}$  if  $x_i = b$ . From this fact, the one-time unforgeability of  $\text{SIG}$  directly follows from the security of  $f$ .  $\square$

## 4 Certified Everlasting Lemmas

In this section, we first review the certified everlasting lemma by Bartusek and Khurana [BK22], which provides cryptography with privately verifiable deletion. Next, we present our new certified everlasting lemma, which provides cryptography with PVD.

**Lemma 4.1 (Certified Everlasting Lemma [BK22]).** *Let  $\{\mathcal{Z}_\lambda(\cdot, \cdot, \cdot)\}_{\lambda \in \mathbb{N}}$  be an efficient quantum operation with three arguments: a  $\lambda$ -bit string  $\theta$ , a bit  $\beta$ , and a quantum register  $A$ . For any QPT  $\mathcal{B}$ , consider the following experiment  $\tilde{\mathcal{Z}}_\lambda^\mathcal{B}(b)$  over quantum states, obtained by running  $\mathcal{B}$  as follows.*

- Sample  $x, \theta \leftarrow \{0, 1\}^\lambda$  and initialize  $\mathcal{B}$  with  $\mathcal{Z}_\lambda(\theta, b \oplus \bigoplus_{i:\theta_i=1} x_i, |x\rangle_\theta)$ .
- $\mathcal{B}$ 's output is parsed as a string  $x' \in \{0, 1\}^\lambda$  and a residual state on register  $B$ .
- If  $x_i = x'_i$  for all  $i$  such that  $\theta_i = 0$ , output  $B$ , and otherwise output a special symbol  $\perp$ .

*Assume that for any QPT  $\mathcal{A}$ ,  $\theta \in \{0, 1\}^\lambda$ ,  $\beta \in \{0, 1\}$ , and an efficiently samplable state  $|\psi\rangle^{A,C}$  on registers  $A$  and  $C$ , we have*

$$\left| \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\theta, \beta, A), C) = 1] - \Pr[\mathcal{A}(\mathcal{Z}_\lambda(0^\lambda, \beta, A), C) = 1] \right| \leq \text{negl}(\lambda).$$

*Then, for any QPT  $\mathcal{B}$ , we have*

$$\text{TD}(\tilde{\mathcal{Z}}_\lambda^\mathcal{B}(0), \tilde{\mathcal{Z}}_\lambda^\mathcal{B}(1)) \leq \text{negl}(\lambda).$$

**Remark 4.2.** Besides notational differences, the above lemma has the following differences from the original lemma by [BK22].

- We focus on QPT adversaries  $\mathcal{A}$  and  $\mathcal{B}$ , though the original lemma by [BK22] captures more general classes of adversaries.
- The roles of computational basis position and Hadamard basis position in  $\tilde{\mathcal{Z}}_\lambda^{\mathcal{A}'}(b)$  are switched.

We can upgrade Lemma 4.1 to a publicly verifiable one by using signatures with one-time unforgeability for BB84 state introduced in Section 3.

**Lemma 4.3 (Publicly Verifiable Certified Everlasting Lemma).** *Let  $\text{SIG} = (\text{Gen}, \text{Sign}, \text{Vrfy})$  be a signature scheme satisfying one-time unforgeability for BB84 states. Let  $\{\mathcal{Z}_\lambda(\cdot, \cdot, \cdot, \cdot, \cdot)\}_{\lambda \in \mathbb{N}}$  be an efficient quantum operation with five arguments: a verification key  $\text{vk}$  and a signing key  $\text{sigk}$  of  $\text{SIG}$ , a  $\lambda$ -bit string  $\theta$ , a bit  $\beta$ , and a quantum register  $A$ . For any QPT  $\mathcal{B}$ , consider the following experiment  $\tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(b)$  over quantum states, obtained by running  $\mathcal{B}$  as follows.*

- Sample  $x, \theta \leftarrow \{0, 1\}^\lambda$  and generate  $(\text{vk}, \text{sigk}) \leftarrow \text{Gen}(1^\lambda)$ . Generate a quantum state  $|\psi\rangle$  by applying the map  $|m\rangle |0 \dots 0\rangle \rightarrow |m\rangle |\text{Sign}(\text{sigk}, m)\rangle$  to  $|x\rangle_\theta \otimes |0 \dots 0\rangle$ . Initialize  $\mathcal{B}$  with  $\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, \theta, b \oplus \bigoplus_{i:\theta_i=1} x_i, |\psi\rangle)$ .
- $\mathcal{B}$ 's output is parsed as a pair of strings  $(x', \sigma')$  and a residual state on register  $B$ .
- If  $\text{Vrfy}(\text{vk}, x', \sigma') = \top$ , output  $B$ , and otherwise output a special symbol  $\perp$ .

Assume that for any QPT  $\mathcal{A}$ , key pair  $(\text{vk}, \text{sigk})$  of  $\text{SIG}$ ,  $\theta \in \{0, 1\}^\lambda$ ,  $\beta \in \{0, 1\}$ , and an efficiently samplable state  $|\psi\rangle^{A,C}$  on registers  $A$  and  $C$ , we have

$$\left| \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, \theta, \beta, A), C) = 1] - \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, 0^{\ell_{\text{sigk}}}, \theta, \beta, A), C) = 1] \right| \leq \text{negl}(\lambda), \quad \text{and} \quad (1)$$

$$\left| \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, \theta, \beta, A), C) = 1] - \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, 0^\lambda, \beta, A), C) = 1] \right| \leq \text{negl}(\lambda). \quad (2)$$

Then, for any QPT  $\mathcal{B}$ , we have

$$\text{TD}(\tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(0), \tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(1)) \leq \text{negl}(\lambda).$$

*Proof.* We first define the following event  $\text{Forge}$ .

**Forge:** For  $(x', \sigma')$  output by  $\mathcal{B}$  in  $\tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(b)$ , the followings are satisfied.

- $x_i \neq x'_i$  holds for some  $i$  such that  $\theta_i = 0$ .
- $\text{Vrfy}(\text{vk}, x', \sigma') = \top$  holds.

We also define the event  $\text{Forge}^*$  in the same way as  $\text{Forge}$  except that  $\mathcal{B}$  is initialized with  $\mathcal{Z}_\lambda(\text{vk}, 0^{\ell_{\text{sigk}}}, \theta, b \oplus \bigoplus_{i:\theta_i=1} x_i, |\psi\rangle)$ . From Equation (1), we have  $\Pr[\text{Forge}] = \Pr[\text{Forge}^*] + \text{negl}(\lambda)$ . Also, from the one-time unforgeability for BB84 states, we have  $\Pr[\text{Forge}^*] = \text{negl}(\lambda)$ . Thus, we obtain  $\Pr[\text{Forge}] = \text{negl}(\lambda)$ .

We define  $\hat{\mathcal{Z}}_\lambda^{\mathcal{B}}(b)$  as the experiment defined in the same way as  $\tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(b)$  except that the experiment outputs the register  $B$  if and only if  $x_i = x'_i$  holds for all  $i$  such that  $\theta_i = 0$ . Since  $\Pr[\text{Forge}] = \text{negl}(\lambda)$ , if  $\text{Vrfy}(\text{vk}, x', \sigma') = \top$  holds, then  $x_i = x'_i$  holds for all  $i$  such that  $\theta_i = 0$ , except negligible probability. Thus, we have

$$\text{TD}(\tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(0), \tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(1)) \leq \text{TD}(\hat{\mathcal{Z}}_\lambda^{\mathcal{B}}(0), \hat{\mathcal{Z}}_\lambda^{\mathcal{B}}(1)) + \text{negl}(\lambda).$$

From Equation (2) and Lemma 4.1, we have  $\text{TD}(\tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(0), \tilde{\mathcal{Z}}_\lambda^{\mathcal{B}}(1)) = \text{negl}(\lambda)$ . This completes the proof.  $\square$

## 5 Instantiations

We can apply Lemma 4.3 to various cryptographic primitives.

### 5.1 Public-Key Encryption

First, we show how to apply Lemma 4.3 to PKE and obtain PKE with PVD.

**Definition.** We define PKE with publicly verifiable deletion (PKE-PVD).

**Definition 5.1 (PKE with Publicly Verifiable Deletion).** A PKE scheme with publicly verifiable deletion is a tuple of efficient algorithms  $\text{PKE-PVD} = (\text{Gen}, \text{Enc}, \text{Dec}, \text{Del}, \text{Vrfy})$ .

$\text{Gen}(1^\lambda) \rightarrow (\text{pk}, \text{sk})$ : The key generation algorithm takes the security parameter  $1^\lambda$  as input and outputs a public key  $\text{pk}$  and a secret key  $\text{sk}$ .

$\text{Enc}(\text{pk}, m) \rightarrow (\text{vk}, ct)$ : The encryption algorithm takes  $\text{pk}$  and a plaintext  $m \in \{0, 1\}$  as input, and outputs a verification key  $\text{vk}$  and a ciphertext  $ct$ .

$\text{Dec}(\text{sk}, ct) \rightarrow m'$ : The decryption algorithm takes  $\text{sk}$  and  $ct$  as input, and outputs a plaintext  $m' \in \mathcal{M}$  or  $\perp$ .

$\text{Del}(ct) \rightarrow \text{cert}$ : The deletion algorithm takes  $ct$  as input and outputs a certification  $\text{cert}$ .

$\text{Vrfy}(\text{vk}, \text{cert}) \rightarrow \top$  or  $\perp$ : The verification algorithm is a deterministic algorithm that takes  $\text{vk}$  and  $\text{cert}$  as input, and outputs  $\top$  or  $\perp$ .

**Decryption Correctness:** For any  $m \in \mathcal{M}$ , we have

$$\Pr \left[ m' \neq m \mid \begin{array}{l} (\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda) \\ (\text{vk}, ct) \leftarrow \text{Enc}(\text{pk}, m) \\ m' \leftarrow \text{Dec}(\text{sk}, ct) \end{array} \right] \leq \text{negl}(\lambda).$$

**Verification Correctness:** For any  $m \in \mathcal{M}$ , we have

$$\Pr \left[ \text{Vrfy}(\text{vk}, \text{cert}) = \perp \mid \begin{array}{l} (\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda) \\ (\text{vk}, ct) \leftarrow \text{Enc}(\text{pk}, m) \\ \text{cert} \leftarrow \text{Del}(ct) \end{array} \right] \leq \text{negl}(\lambda).$$

**Semantic Security:** For any QPT  $\mathcal{A}$ , we have

$$\left| \Pr \left[ \mathcal{A}(\text{pk}, \text{vk}, ct) = 1 \mid \begin{array}{l} (\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda) \\ (\text{vk}, ct) \leftarrow \text{Enc}(\text{pk}, 0) \end{array} \right] - \Pr \left[ \mathcal{A}(\text{pk}, \text{vk}, ct_1) = 1 \mid \begin{array}{l} (\text{pk}, \text{sk}) \leftarrow \text{KeyGen}(1^\lambda) \\ (\text{vk}, ct) \leftarrow \text{Enc}(\text{pk}, 1) \end{array} \right] \right| = \text{negl}(\lambda).$$

**Definition 5.2 (Certified Deletion Security for PKE-PVD).** Let  $\text{PKE-PVD} = (\text{Gen}, \text{Enc}, \text{Dec}, \text{Del}, \text{Vrfy})$  be a PKE-PVD scheme. We consider the following security experiment  $\text{Exp}_{\text{PKE-PVD}, \mathcal{A}}^{\text{cert-del}}(\lambda, b)$  against a QPT adversary  $\mathcal{A}$ .

1. The challenger computes  $(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda)$ , and sends  $\text{pk}$  to  $\mathcal{A}$ .
2. The challenger computes  $(\text{vk}, ct) \leftarrow \text{Enc}(\text{pk}, b)$ , and sends  $\text{vk}$  and  $ct$  to  $\mathcal{A}$ .
3. At some point,  $\mathcal{A}$  sends  $\text{cert}$  and its internal state  $\rho$  to the challenger.
4. The challenger computes  $\text{Vrfy}(\text{vk}, \text{cert})$ . If the outcome is  $\top$ , the challenger outputs  $\rho$  and otherwise outputs  $\perp$ .

We say that PKE-PVD satisfies certified deletion security if for any QPT  $\mathcal{A}$ , it holds that

$$\text{TD}(\text{Exp}_{\text{PKE-PVD}, \mathcal{A}}^{\text{cert-del}}(\lambda, 0), \text{Exp}_{\text{PKE-PVD}, \mathcal{A}}^{\text{cert-del}}(\lambda, 1)) \leq \text{negl}(\lambda).$$

*Remark 5.3.* We define PKE-PVD and the certified deletion security for it as the plaintext space of PKE-PVD is  $\{0, 1\}$  by default. We can generalize them into one for the plaintext space  $\{0, 1\}^\ell$  for any polynomial  $\ell$ . Also, such a multi-bit plaintext PKE-PVD can be constructed from a single-bit plaintext PKE-PVD by the standard hybrid argument.

**Construction.** We construct a PKE-PVD scheme  $\text{PKE-PVD} = (\text{Gen}, \text{Enc}, \text{Dec}, \text{Del}, \text{Vrfy})$  for the plaintext space  $\{0, 1\}$ . The building blocks are as follows.

- A public-key encryption scheme  $\text{PKE} = \text{PKE}(\text{Gen}, \text{Enc}, \text{Dec})$ .
- A deterministic signature  $\text{SIG} = \text{SIG}(\text{Gen}, \text{Sign}, \text{Vrfy})$ .

$\text{Gen}(1^\lambda)$ :

- Output  $(\text{pk}, \text{sk}) \leftarrow \text{PKE.Gen}(1^\lambda)$ .

$\text{Enc}(\text{pk}, m \in \{0, 1\})$ :

- Generate  $x, \theta \leftarrow \{0, 1\}^\lambda$  and generate  $(\text{vk}, \text{sigk}) \leftarrow \text{SIG.Gen}(1^\lambda)$ . Generate a quantum state  $|\psi\rangle$  by applying the map  $|m\rangle |0 \dots 0\rangle \rightarrow |m\rangle |\text{SIG.Sign}(\text{sigk}, m)\rangle$  to  $|x\rangle_\theta \otimes |0 \dots 0\rangle$ .
- Generate  $\text{pke.ct} \leftarrow \text{PKE.Enc}(\text{pk}, (\text{sigk}, \theta, m \oplus \bigoplus_{i:\theta_i=1} x_i))$ .
- Output  $\text{vk}$  and  $\text{ct} := (|\psi\rangle, \text{pke.ct})$ .

$\text{Dec}(\text{sk}, \text{ct})$ :

- Parse  $\text{ct}$  into a quantum states  $\rho$  and classical bit string  $\text{pke.ct}$ .
- Compute  $(\text{sigk}, \theta, \beta) \leftarrow \text{PKE.Dec}(\text{sk}, \text{pke.ct})$ .
- Apply the map  $|m\rangle |0 \dots 0\rangle \rightarrow |m\rangle |\text{SIG.Sign}(\text{sigk}, m)\rangle$  to  $\rho$ , measure the first  $\lambda$  qubits of the resulting state in Hadamard basis, and obtain  $\bar{x}$ .
- Output  $m \leftarrow \beta \oplus \bigoplus_{i:\theta_i=1} \bar{x}_i$ .

$\text{Del}(\text{ct})$ :

- Parse  $\text{ct}$  into a quantum state  $\rho$  and a classical string  $\text{pke.ct}$ .
- Measure  $\rho$  in the computational basis and obtain  $x'$  and  $\sigma'$ .
- Output  $(x', \sigma')$ .

$\text{Vrfy}(\text{vk}, \text{cert})$ :

- Parse  $(z', \sigma') \leftarrow \text{cert}$ .
- Output the result of  $\text{SIG.Vrfy}(\text{vk}, z', \sigma')$ .

We see that PKE-PVD satisfies decryption correctness and verification correctness if SIG and PKE satisfy their correctness notions. Also, the semantic security of PKE-PVD immediately follows from that of PKE. Also, we have the following theorem.

**Theorem 5.4.** *Assume SIG satisfies one-time unforgeability for BB84 states and PKE satisfies semantic security. Then, PKE-PVD satisfies certified deletion security.*

*Proof.* We define  $\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, \theta, \beta, A)$  be an efficient quantum process such that it generates  $(\text{pk}, \text{sk}) \leftarrow \text{Gen}(1^\lambda)$  and outputs  $(\text{pk}, \text{vk}, \text{Enc}(\text{pk}, (\text{sigk}, \theta, \beta)))$ . Then, from the semantic security of PKE, for any QPT  $\mathcal{A}$ , key pair  $(\text{vk}, \text{sk})$  of SIG,  $\theta \in \{0, 1\}^\lambda$ ,  $\beta \in \{0, 1\}$ , and an efficiently samplable state  $|\psi\rangle^{A,C}$  on registers A and C, we have

$$\left| \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, \theta, \beta, A), C) = 1] - \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, 0^{\ell_{\text{sigk}}}, \theta, \beta, A), C) = 1] \right| \leq \text{negl}(\lambda), \quad \text{and}$$

$$\left| \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, \theta, \beta, A), C) = 1] - \Pr[\mathcal{A}(\mathcal{Z}_\lambda(\text{vk}, \text{sigk}, 0^\lambda, \beta, A), C) = 1] \right| \leq \text{negl}(\lambda).$$

Then, the certified deletion security of PKE-PVD follows from Lemma 4.3.  $\square$

In Section 3, we show that a deterministic signature scheme with one-time unforgeability for BB84 states is implied by OWF that is implied by a PKE scheme. Thus, we obtain the following theorem.

**Theorem 5.5.** *Assume that there exists PKE. Then, there exists PKE-PVD.*

## 5.2 Other Primitives

By combining Lemma 4.3 with other types of primitives instead of PKE, we immediately obtain them with publicly verifiable deletion. That is, we instantiate  $\mathcal{Z}_\lambda$  in Lemma 4.3 with these primitives. Formally, we have the following theorem.

**Theorem 5.6.** *If there exists*

$$Z' \in \{SKE, COM, PKE, ABE, QFHE, TRE\},$$

*then, there exists  $Z'$  with publicly verifiable deletion. If there exist WE and one-way functions, then there exists WE with publicly verifiable deletion.*

*Remark 5.7.* We additionally assume one-way functions for the case of WE since WE is unlikely to imply one-way functions while any of  $Z' \in \{SKE, COM, PKE, ABE, QFHE, TRE\}$  implies them.

We omit the definitions of these primitives (with publicly verifiable deletion) and refer the reader to [BK22, BKP23] for them.<sup>10</sup>

## 6 Making Assumptions Minimal

In Section 5, we show that  $Z \in \{SKE, COM, PKE, ABE, QFHE, TRE, WE\}$  (and one-way functions in the case of  $Z = WE$ ) imply  $Z$  with publicly verifiable deletion. However,  $Z$  with publicly verifiable deletion does not seem to imply either of plain (classical)  $Z$  or one-way functions in general. In this section, we explain how to weaken the assumptions to minimal ones implied by  $Z$  with publicly verifiable deletion.

First, we observe that our compiler works even if we start with  $Z \in \{SKE, COM, PKE, ABE, QFHE, TRE, WE\}$  that has quantum encryption and decryption (or committing) algorithms. Second, we observe our compiler works even if the underlying digital signature scheme has a quantum key generation algorithm since the quantum encryption algorithm runs it. Moreover, such a digital signature scheme with a quantum key generation algorithm that satisfies one-time security for BB84 states can be constructed from hard quantum planted problems for NP defined below.

**Definition 6.1 (Hard Quantum Planted Problem for NP).** *A quantum polynomial-time algorithm  $\mathcal{G}$  is a sampler for an NP relation  $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$  if, for every  $n$ ,  $\mathcal{G}(1^n)$  outputs a pair  $(x, w)$  such that  $(x, w) \in \mathcal{R}$  with probability 1. We say that the quantum planted problem corresponding to  $(\mathcal{G}, \mathcal{R})$  is hard if, for every QPT  $\mathcal{A}$ , it holds that*

$$\Pr[(x, \mathcal{A}(x)) \in \mathcal{R} \mid (x, w) \leftarrow \mathcal{G}(1^n)] \leq \text{negl}(n).$$

It is clear that the construction in Section 3 works using hard quantum planted problems for NP instead of one-way functions where input-output pairs of the one-way function are replaced with witness-instance pairs of the NP problem if we allow the key generation algorithm of the signature scheme to be quantum. Combining the above observations, we obtain the following theorem.

**Theorem 6.2.** *Assume that there exists*

$$Z \in \{SKE, COM, PKE, ABE, QFHE, TRE, WE\}$$

*with quantum encryption and decryption (or committing) algorithms and hard quantum planted problems for NP. Then, there exists  $Z$  with publicly verifiable deletion.*

The assumption in the above theorem is minimal since

1.  $Z$  with quantum encryption and decryption (or committing) algorithms is immediately implied by  $Z$  with publicly verifiable deletion by simply ignoring the deletion and verification algorithms, and

---

<sup>10</sup>The definitions in [BK22] only consider privately verifiable deletion, but it is straightforward to extend them to ones with publicly verifiable deletion.

2. Hard quantum planted problems for NP are implied by Z with publicly verifiable deletion by regarding the verification key and certificate as instance and certificate, respectively.

*Remark 6.3.* In the second item above, we assume that the verification algorithm of Z with publicly verifiable deletion is a classical deterministic algorithm. If it is allowed to be a quantum algorithm, we need to consider hard quantum planted problems for QCMA instead of NP. The construction in Section 3 works with such problems if we allow the verification algorithm to be quantum. Thus, the minimality of the assumption holds in this setting as well.

## References

- [AGKZ20] Ryan Amos, Marios Georgiou, Aggelos Kiayias, and Mark Zhandry. One-shot signatures and applications to hybrid quantum/classical authentication. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *52nd ACM STOC*, pages 255–268. ACM Press, June 2020. (Cited on page 3.)
- [AKN<sup>+</sup>23] Shweta Agrawal, Fuyuki Kitagawa, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Public key encryption with secure key leasing. *IACR Cryptol. ePrint Arch.*, page 264, 2023. Eurocrypt 2023 (to appear). (Cited on page 7.)
- [AL21] Prabhanjan Ananth and Rolando L. La Placa. Secure software leasing. In Anne Canteaut and François-Xavier Standaert, editors, *EUROCRYPT 2021, Part II*, volume 12697 of *LNCS*, pages 501–530. Springer, Heidelberg, October 2021. (Cited on page 7.)
- [ALL<sup>+</sup>21] Scott Aaronson, Jiahui Liu, Qipeng Liu, Mark Zhandry, and Ruizhe Zhang. New approaches for quantum copy-protection. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 526–555, Virtual Event, August 2021. Springer, Heidelberg. (Cited on page 7.)
- [APV23] Prabhanjan Ananth, Alexander Poremba, and Vinod Vaikuntanathan. Revocable cryptography from learning with errors. Cryptology ePrint Archive, Report 2023/325, 2023. <https://eprint.iacr.org/2023/325>. (Cited on page 7.)
- [BGG<sup>+</sup>23] James Bartusek, Sanjam Garg, Vipul Goyal, Dakshita Khurana, Giulio Malavolta, Justin Raizes, and Bhaskar Roberts. Obfuscation and outsourced computation with certified deletion. Cryptology ePrint Archive, Report 2023/265, 2023. <https://eprint.iacr.org/2023/265>. (Cited on page 3, 7.)
- [BGI<sup>+</sup>12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *Journal of the ACM*, 59(2):6:1–6:48, 2012. (Cited on page 3.)
- [BI20] Anne Broadbent and Rabib Islam. Quantum encryption with certified deletion. In Rafael Pass and Krzysztof Pietrzak, editors, *TCC 2020, Part III*, volume 12552 of *LNCS*, pages 92–122. Springer, Heidelberg, November 2020. (Cited on page 3, 6.)
- [BJL<sup>+</sup>21] Anne Broadbent, Stacey Jeffery, Sébastien Lord, Supartha Podder, and Aarthi Sundaram. Secure software leasing without assumptions. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part I*, volume 13042 of *LNCS*, pages 90–120. Springer, Heidelberg, November 2021. (Cited on page 7.)
- [BK22] James Bartusek and Dakshita Khurana. Cryptography with certified deletion. Cryptology ePrint Archive, Report 2022/1178, 2022. <https://eprint.iacr.org/2022/1178>. (Cited on page 3, 4, 6, 7, 9, 10, 13.)
- [BKP23] James Bartusek, Dakshita Khurana, and Alexander Poremba. Publicly-verifiable deletion via target-collapsing functions. *IACR Cryptol. ePrint Arch.*, page 370, 2023. (Cited on page 3, 4, 13.)

- [CLLZ21] Andrea Coladangelo, Jiahui Liu, Qipeng Liu, and Mark Zhandry. Hidden cosets and applications to unclonable cryptography. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part I*, volume 12825 of *LNCS*, pages 556–584, Virtual Event, August 2021. Springer, Heidelberg. (Cited on page 3.)
- [CMP20] Andrea Coladangelo, Christian Majenz, and Alexander Poremba. Quantum copy-protection of compute-and-compare programs in the quantum random oracle model. Cryptology ePrint Archive, Report 2020/1194, 2020. <https://eprint.iacr.org/2020/1194>. (Cited on page 7.)
- [GKP<sup>+</sup>13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nikolai Zeldovich. How to run turing machines on encrypted data. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part II*, volume 8043 of *LNCS*, pages 536–553. Springer, Heidelberg, August 2013. (Cited on page 3.)
- [HKM<sup>+</sup>23] 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, Report 2023/236, 2023. <https://eprint.iacr.org/2023/236>. (Cited on page 3, 7.)
- [HMNY21] Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Quantum encryption with certified deletion, revisited: Public key, attribute-based, and classical communication. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part I*, volume 13090 of *LNCS*, pages 606–636. Springer, Heidelberg, December 2021. (Cited on page 3, 6.)
- [HMNY22] Taiga Hiroka, Tomoyuki Morimae, Ryo Nishimaki, and Takashi Yamakawa. Certified everlasting zero-knowledge proof for QMA. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 239–268. Springer, Heidelberg, August 2022. (Cited on page 6.)
- [HMY22] Minki Hhan, Tomoyuki Morimae, and Takashi Yamakawa. From the hardness of detecting superpositions to cryptography: Quantum public key encryption and commitments. Cryptology ePrint Archive, Report 2022/1375, 2022. <https://eprint.iacr.org/2022/1375>. (Cited on page 3.)
- [KN22] Fuyuki Kitagawa and Ryo Nishimaki. Functional encryption with secure key leasing. In Shweta Agrawal and Dongdai Lin, editors, *ASIACRYPT 2022, Part IV*, volume 13794 of *LNCS*, pages 569–598. Springer, Heidelberg, December 2022. (Cited on page 7.)
- [KNY21] Fuyuki Kitagawa, Ryo Nishimaki, and Takashi Yamakawa. Secure software leasing from standard assumptions. In Kobbi Nissim and Brent Waters, editors, *TCC 2021, Part I*, volume 13042 of *LNCS*, pages 31–61. Springer, Heidelberg, November 2021. (Cited on page 7.)
- [Lam79] Leslie Lamport. Constructing digital signatures from a one-way function. Technical Report SRI-CSL-98, SRI International Computer Science Laboratory, October 1979. (Cited on page 4.)
- [MU10] Jörn Müller-Quade and Dominique Unruh. Long-term security and universal composability. *Journal of Cryptology*, 23(4):594–671, October 2010. (Cited on page 6.)
- [Por23] 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. (Cited on page 3.)
- [Unr15] Dominique Unruh. Revocable quantum timed-release encryption. *J. ACM*, 62(6):49:1–49:76, 2015. (Cited on page 3.)