Quantum Encryption with Certified Deletion, Revisited: Public Key, Attribute-Based, and Classical Communication [*]

Taiga Hiroka[1], Tomoyuki Morimae[1,2], Ryo Nishimaki[3], Takashi Yamakawa[3]

[1]Yukawa Institute for Theoretical Physics, Kyoto University, Japan
{taiga.hiroka,tomoyuki.morimae}@yukawa.kyoto-u.ac.jp
[2]PRESTO, JST, Japan
[3]NTT Secure Platform Laboratories, Tokyo, Japan
{ryo.nishimaki.zk,takashi.yamakawa.ga}@hco.ntt.co.jp

May 12, 2021

Abstract

Broadbent and Islam (TCC '20) proposed a quantum cryptographic primitive called *quantum encryption with certified deletion*. In this primitive, a receiver in possession of a quantum ciphertext can generate a classical certificate that the encrypted message is deleted. Although their construction is information-theoretically secure, it is limited to the setting of one-time symmetric key encryption (SKE), where a sender and receiver have to share a common key in advance and the key can be used only once. Moreover, the sender has to generate a quantum state and send it to the receiver over a quantum channel in their construction. Although deletion certificates are privately verifiable, which means a verification key for a certificate has to be kept secret, in the definition by Broadbent and Islam, we can also consider public verifiability.

In this work, we present various constructions of encryption with certified deletion.

- Quantum communication case: We achieve (reusable-key) public key encryption (PKE) and attribute-based encryption (ABE) with certified deletion. Our PKE scheme with certified deletion is constructed assuming the existence of IND-CPA secure PKE, and our ABE scheme with certified deletion is constructed assuming the existence of indistinguishability obfuscation and one-way function. These two schemes are privately verifiable.
- Classical communication case: We also achieve PKE with certified deletion that uses only classical communication. We give two schemes, a privately verifiable one and a publicly verifiable one. The former is constructed assuming the LWE assumption in the quantum random oracle model. The latter is constructed assuming the existence of one-shot signatures and extractable witness encryption.

[*]This is a major update version of the paper by Nishimaki and Yamakawa [NY21] with many new results.

Contents

1 Introduction

The no-cloning theorem, which means that an unknown quantum state cannot be copied in general, is one of the most fundamental principles in quantum physics. As any classical information can be trivially copied, this indicates a fundamental difference between classical and quantum information. The no-cloning theorem has been the basis of many quantum cryptographic protocols, including quantum money [Wie83] and quantum key distribution [BB84].

Broadbent and Islam [BI20] used the principle to construct *quantum encryption with certified deletion*. In this primitive, a sender encrypts a classical message to generate a quantum ciphertext. A receiver in possession of the quantum ciphertext and a classical decryption key can either decrypt the ciphertext or "delete" the encrypted message by generating a classical certificate. After generating a valid certificate of deletion, no adversary can recover the message *even if the decryption key is given*.[1] We remark that this functionality is classically impossible to achieve since one can copy a classical ciphertext and keep it so that s/he can decrypt it at any later time. They prove the security of their construction without relying on any computational assumption, which ensures information-theoretical security. Although they achieved the exciting new functionality, their construction is limited to the one-time symmetric key encryption (SKE) setting. In one-time SKE, a sender and receiver have to share a common key in advance, and the key can be used only once.

A possible application scenario of quantum encryption with certified deletion is the following. A user uploads encrypted data on a quantum cloud server. Whenever the user wishes to delete the data, the cloud generates a deletion certificate and sends it to the user. After the user verifies the validity of the certificate, s/he is convinced that the data cannot be recovered even if the decryption key is accidentally leaked later. Such quantum encryption could prevent data retention and help to implement the right to be forgotten [GDP16]. In this scenario, one-time SKE is quite inconvenient. By the one-time restriction, the user has to locally keep as many decryption keys as the number of encrypted data in the cloud, in which case there seems to be no advantage of uploading the data to the cloud server: If the user has such large storage, s/he could have just locally kept the messages rather than uploading encryption of them to the cloud. Also, in some cases, a party other than the decryptor may want to upload data to the cloud. This usage would be possible if we can extend the quantum encryption with certified deletion to public key encryption (PKE). Remark that the one-time restriction is automatically resolved for PKE by a simple hybrid argument. Even more flexibly, a single encrypted data on the cloud may be supposed to be decrypted by multiple users according to some access control policy. Such an access control has been realized by attribute-based encryption (ABE) [SW05, GPSW06] in classical cryptography. Thus, it would be useful if we have ABE with certified deletion. Our first question in this work is:

Can we achieve PKE and ABE with certified deletion?

Moreover, a sender needs to send quantum states (random BB84 states [BB84]) over a quantum channel in the construction by Broadbent and Islam [BI20]. Although generating and sending random BB84 states are not so difficult tasks (and in fact they are already possible with current technologies), a classical sender and communication over only a classical channel are of course much easier. Besides, communicating over a classical channel is desirable in the application scenario above since many parties want to upload data to a cloud. In addition to these practical motivations, furthermore, achieving classical channel certified deletion is also an interesting theoretical research direction given the fact that many quantum cryptographic protocols have been "dequantized" recently [Mah18, CCKW19, RS19, AGKZ20, KNY20]. Thus, our second question in this work is:

Can we achieve PKE with certified deletion, a classical sender, and classical communication?

In the definition by Broadbent and Islam [BI20], a verification key for a deletion certificate must be kept secret (privately verifiable). If the verification key is revealed, the security is no longer guaranteed in their scheme. We can also consider public verifiability, which means the security holds even if a verification key is revealed to adversaries. Broadbent and Islam left the following question as an open problem:

Is publicly verifiable encryption with certified deletion possible?

[1] We note that if the adversary is given the decryption key before the deletion, it can decrypt the ciphertext to obtain the message and keep it even after the deletion, but such an "attack" is unavoidable.

1.1 Our Result

We solve the three questions above affirmatively in this work.

PKE and ABE with certified deletion and quantum communication. We present formal definitions of PKE and ABE with certified deletion, and present constructions of them:

- We construct a PKE scheme with certified deletion assuming the existence of (classical) IND-CPA secure PKE. We also observe that essentially the same construction gives a reusable SKE scheme with certified deletion if we use IND-CPA secure SKE, which exists under the existence of one-way function (OWF), instead of PKE.
- We construct a (public-key) ABE scheme with certified deletion assuming the existence of indistinguishability obfuscation (IO) [BGI+12] and OWF. This construction satisfies the collusion resistance and adaptive security, i.e., it is secure against adversaries that adaptively select a target attribute and obtain arbitrarily many decryption keys.

We note that our constructions rely on computational assumptions and thus not information-theoretically secure, unlike the construction in [BI20]. This is unavoidable since even plain PKE or ABE cannot be information-theoretically secure. We also note that the constructions above are privately verifiable as the definition of one-time SKE by Broadbent and Islam [BI20].

Our main technical insight is that we can combine the one-time secure SKE with certified deletion of [BI20] and plain PKE to construct PKE with certified deletion by a simple hybrid encryption technique if the latter satisfies *receiver non-committing* (RNC) security [CFGN96, JL00, CHK05]. Since it is known that PKE/SKE with RNC security can be constructed from any IND-CPA secure PKE/SKE [CHK05, KNTY19], our first result follows.

For the second result, we first give a suitable definition of RNC security for ABE that suffices for our purpose. Then we construct an ABE scheme with RNC security based on the existence of IO and OWF. By combining this with one-time SKE with certified deletion by hybrid encryption, we obtain an ABE scheme with certified deletion.

PKE with certified deletion, a classical sender, and classical communication. We also present formal definitions of PKE with certified deletion and classical communication, and present two constructions:

- We construct a PKE scheme with privately verifiable certified deletion and classical communication in the quantum random oracle model (QROM) [BDF+11]. Our construction is secure under the LWE assumption in the QROM.
- We construct a PKE scheme with publicly verifiable certified deletion and classical communication. Our construction uses one-shot signatures [AGKZ20] and extractable witness encryption [GGSW13, GKP+13]. This solves the open problem by Broadbent and Islam [BI20].

In both constructions, a sender is a classical algorithm, but needs to interact with a receiver during ciphertext generation.

In the classical communication case, an encryption algorithm must be interactive even if we consider computationally bounded adversaries (and even in the QROM). The reason is that a malicious QPT receiver can generate two copies of a quantum ciphertext from classical messages sent from a sender, and one is used for generating a deletion certificate and the other is used for decryption.

Moreover, both constructions rely on computational assumptions and thus not information-theoretically secure, unlike the construction by Broadbent and Islam [BI20]. This is unavoidable even if an encryption algorithm is interactive (and even in the QROM). The reason is that a computationally unbounded malicious receiver can classically simulate its honest behavior to get a classical description of the quantum ciphertext.

For the first construction, we use a new property of noisy trapdoor claw-free (NTCF) functions, *the cut-and-choose adaptive hardcore property* (Lemma 5.6), which we introduce in this work. We prove that the cut-and-choose adaptive hardcore property is reduced to the adaptive hardcore bit property [BCM+18] and injective invariance [Mah18]. Those properties hold under the LWE assumption [BCM+18, Mah18]. This new technique is of independent interest. The idea of the second construction is to encrypt a plaintext by witness encryption so that a valid witness is a one-shot signature for bit 0. We use a valid one-shot signature for bit 1 as a deletion certificate. The one-shot property of one-shot signatures prevents decryption of witness encryption after issuing a valid deletion certificate. Georgiou and Zhandry [GZ20] used a similar combination of one-shot signatures and witness encryption to construct unclonable decryption keys.

1.2 Related work

Before the work by Broadbent and Islam [BI20], Fu and Miller [FM18] and Coiteux-Roy and Wolf [CRW19] also studied the concept of certifying deletion of information in different settings. (See [BI20] for the comparison with these works.)

The quantum encryption scheme with certified deletion by Broadbent and Islam [BI20] is based on Wiesner's conjugate coding, which is the backbone of quantum money [Wie83] and quantum key distribution [BB84]. A similar idea has been used in many constructions in quantum cryptography that include (but not limited to) revocable quantum timed-release encryption [Unr15], uncloneable quantum encryption [BL20], single-decryptor encryption [GZ20], and copy protection/secure software leasing [CMP20]. Among them, revocable quantum timed-release encryption is conceptually similar to quantum encryption with certified deletion. In this primitive, a receiver can decrypt a quantum ciphertext only after spending a certain amount of time T. The receiver can also choose to return the ciphertext before the time T is over, in which case it is ensured that the message can no longer be recovered. As observed by Broadbent and Islam [BI20], an essential difference from quantum encryption with certified deletion is that the revocable quantum timed-release encryption does not have a mechanism to generate a *classical* certificate of deletion. Moreover, the construction by Unruh [Unr15] heavily relies on the random oracle heuristic [BR97, BDF$^+$11], and there is no known construction without random oracles.

Kundu and Tan [KT20] constructed (one-time symmetric key) quantum encryption with certified deletion with the device-independent security, i.e., the security holds even if quantum devices are untrusted. Moreover, they show that their construction satisfies composable security.

The notion of NTCF functions was first introduced by Brakerski et al. [BCM$^+$18], and further extended to construct a classical verification of quantum computing by Mahadev [Mah18]. (See also a related primitive so-called QFactory [CCKW19].) The adaptive hardcore bit property of NTCF functions was also used for semi-quantum money [RS19] and secure software leasing with classical communication [KNY20].

Ananth and Kaleoglu concurrently and independently present reusable secret key and public key uncloneable encryption schemes [AK21]. Uncloneable encryption [BL20] is related to but different from quatum encryption with certified deletion. Uncloneable encryption prevents adversaries from creating multiple ciphertexts whose plaintext is the same as that of the original ciphertext. Their constructions are based on a similar idea to one of our main ideas. Specifically, their construction is obtained by combining one-time secret key uncloneable encryption and standard SKE/PKE with the “fake-key property", which is similar to the RNC security.

1.3 Technical Overview Part I: Quantum Communication Case

We provide an overview of how to achieve PKE and ABE with certified deletion using quantum communication in this section. To explain our idea, we introduce the definition of PKE with certified deletion.

Definition of quantum encryption with certified deletion. A PKE with certified deletion consists of the following algorithms.

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{sk})$**:** This is a key generation algorithm that generates a pair of public and secret keys.

$\mathsf{Enc}(\mathsf{pk}, m) \rightarrow (\mathsf{vk}, \mathsf{CT})$**:** This is an encryption algorithm that generates a ciphertext of plaintext and a verification key for this ciphertext.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \rightarrow m'$**:** This is a decryption algorithm that decrypts a ciphertext.

$\mathsf{Del}(\mathsf{CT}) \rightarrow \mathsf{cert}$**:** This is a deletion algorithm that generates a certificate to guarantee that the ciphertext CT was deleted.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) \to \top$ **or** $\bot$**:** This is a verification algorithm that checks the validity of a certificate cert by using a verification key. As correctness, we require that this algorithm returns $\top$ (i.e., it accepts) if cert was honestly generated by $\mathsf{Del}(\mathsf{CT})$ and $(\mathsf{vk}, \mathsf{CT})$ was honestly generated by Enc.

Roughly speaking, certified deletion security requires that no quantum polynomial time (QPT) adversary given pk and CT can obtain any information about the plaintext in CT *even if* sk *is given after a valid certificate* $\mathsf{cert} \leftarrow \mathsf{Del}(\mathsf{CT})$ *is generated*. The difference between PKE and reusable SKE with certified deletion is that, in reusable SKE, KeyGen outputs only sk. In the one-time SKE case by Broadbent and Islam [BI20], Enc does not output vk and Vrfy uses sk instead of vk.

Our idea for PKE. We use the construction of one-time SKE with certified deletion by Broadbent and Islam [BI20]. However, we do not need to know the detail of the SKE scheme since we use it in a black-box way in our PKE scheme. What we need to understand about the SKE scheme are the following abstracted properties: (1) A secret key and a plaintext are classical strings. (2) A ciphertext is a quantum state. (3) The encryption algorithm does not output a verification key since the verification key is equal to the secret key. (4) It satisfies the verification correctness and certified deletion security explained above.

Our idea is to convert the SKE with certified deletion scheme into a PKE with certified deletion scheme by combining with a standard PKE scheme (standard hybrid encryption technique). This conversion is possible since a secret key of the SKE scheme is a classical string. Let $\mathsf{PKE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ and $\mathsf{SKE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ be normal PKE and one-time SKE with certified deletion schemes, respectively. Our PKE with certified deletion scheme is described as follows.

$\mathsf{KeyGen}(1^\lambda)$**:** This outputs $(\mathsf{pke.pk}, \mathsf{pke.sk}) \leftarrow \mathsf{PKE.KeyGen}(1^\lambda)$.

$\mathsf{Enc}(\mathsf{pk}, m)$**:** This generates $\mathsf{ske.sk} \leftarrow \mathsf{SKE.KeyGen}(1^\lambda)$, $\mathsf{ske.CT} \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.sk}, m)$, and $\mathsf{pke.CT} \leftarrow \mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathsf{ske.sk})$, and outputs $\mathsf{vk} := \mathsf{ske.sk}$ and $\mathsf{CT} := (\mathsf{ske.CT}, \mathsf{pke.CT})$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$**:** This computes $\mathsf{ske.sk}' \leftarrow \mathsf{PKE.Dec}(\mathsf{pke.sk}, \mathsf{pke.CT})$ and $m' \leftarrow \mathsf{SKE.Dec}(\mathsf{ske.sk}', \mathsf{ske.CT})$, and outputs m'.

$\mathsf{Del}(\mathsf{CT})$**:** This generates and outputs $\mathsf{cert} \leftarrow \mathsf{SKE.Del}(\mathsf{ske.CT})$.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert})$**:** This outputs the output of $\mathsf{SKE.Vrfy}(\mathsf{ske.sk}, \mathsf{cert})$ (note that $\mathsf{vk} = \mathsf{ske.sk}$).

At first glance, this naive idea seems to work since even if $\mathsf{pke.sk}$ is given to an adversary after a valid cert is generated, $\mathsf{ske.CT}$ does not leak information about the plaintext by certified deletion security of the SKE scheme. Note that PKE is used to encrypt $\mathsf{ske.sk}$ (not m). One-time SKE is sufficient since $\mathsf{ske.sk}$ is freshly generated in Enc. The proof outline is as follows. First, we use IND-CPA security of normal PKE to erase information about $\mathsf{ske.sk}$. Then, we use the one-time certified deletion security of SKE. Unfortunately, we do not know how to prove the first step above because we must give $\mathsf{pke.sk}$ to an adversary in a security reduction. In the first step, we need to show that if a distinguisher detects that $\mathsf{PKE.Enc}(\mathsf{pke.pk}, \mathsf{ske.sk})$ is changed to $\mathsf{PKE.Enc}(\mathsf{pke.pk}, 0^{|\mathsf{ske.sk}|})$, we can break IND-CPA security of the normal PKE. However, to run the distinguisher, we need to give $\mathsf{pke.sk}$ to the distinguisher after it sends a valid certificate for deletion. The reduction has no way to give $\mathsf{pke.sk}$ to the distinguisher since the reduction is trying to break the PKE scheme!

To solve this problem, we use RNC encryption (RNCE) [JL00, CHK05]. RNCE consists of algorithms $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$. The key generation algorithm outputs not only a key pair $(\mathsf{pk}, \mathsf{sk})$ but also an auxiliary trapdoor information aux. The fake ciphertext generation algorithm $\mathsf{Fake}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux})$ can generate a fake ciphertext $\widetilde{\mathsf{CT}}$ that does not include information about a plaintext. The reveal algorithm $\mathsf{Reveal}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m)$ can generate a fake secret key that decrypts $\widetilde{\mathsf{CT}}$ to m. The RNC security notion requires that $(\widetilde{\mathsf{CT}} = \mathsf{Fake}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux}), \mathsf{Reveal}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m))$ is computationally indistinguishable from $(\mathsf{Enc}(\mathsf{pk}, m), \mathsf{sk})$.

RNCE perfectly fits the scenario of certified deletion. We use an RNCE scheme $\mathsf{NCE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ instead of a normal PKE in the PKE with certified deletion scheme above. To erase $\mathsf{ske.sk}$, we use the RNC security. We change $\mathsf{NCE.Enc}(\mathsf{nce.pk}, \mathsf{ske.sk})$ and $\mathsf{nce.sk}$ into $\mathsf{nce}.\widetilde{\mathsf{CT}} = \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux})$ and $\mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce}.\widetilde{\mathsf{CT}}, \mathsf{ske.sk})$, respectively. Thus, as long as $\mathsf{ske.sk}$ is given after a valid certification is generated, we can simulate the secret key of the PKE with certified deletion scheme. Using RNCE solves the problem above since the reduction obtains both a target ciphertext and a secret key (real or fake) in the RNC security game. To complete the security proof, we use the certified deletion security of SKE. Here, the point is that the reduction can simulate a secret key by Reveal since the reduction is given $\mathsf{ske.sk}$ after a valid certificate is sent in the certified deletion security game.

If we use secret key RNCE instead of public key RNCE, we can achieve reusable SKE with certified deletion via the design idea above. Secret/public key RNCE can be constructed from IND-CPA SKE/PKE, respectively [CHK05, KNTY19], and SKE with certified deletion exists unconditionally [BI20]. Thus, we can achieve PKE (resp. reusable SKE) with certified deletion from IND-CPA PKE (resp. OWFs).

Note that the RNCE technique above is the fundamental technique in this work.[2] We use this technique both in the quantum communication case and in the classical communication case.

Our idea for ABE. We can extend the idea for PKE to the ABE setting. In this work, we focus on key-policy ABE, where a policy (resp. attribute) is embedded in a secret key (resp. ciphertext). The crucial tool is (receiver) non-committing ABE (NCABE), which we introduce in this work.

Although the definition of NCABE is basically a natural extension of that of RNCE, we describe algorithms of NCABE for clarity. It helps readers who are not familiar with normal ABE. The first four algorithms below are algorithms of normal ABE.

$\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{msk})$**:** This is a setup algorithm that generates a public key and a master secret key.

$\mathsf{KeyGen}(\mathsf{msk}, P) \rightarrow \mathsf{sk}_P$**:** This is a key generation algorithm that generates a secret key for a policy P.

$\mathsf{Enc}(\mathsf{pk}, X, m) \rightarrow \mathsf{CT}_X$**:** This is an encryption algorithm that generates a ciphertext of m under an attribute X.

$\mathsf{Dec}(\mathsf{sk}_P, \mathsf{CT}_X) \rightarrow m'$ **or** $\bot$**:** This is a decryption algorithm that decrypts CT_X if $P(X) = \top$. If $P(X) = \bot$, it outputs $\bot$.

$\mathsf{FakeSetup}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{aux})$**:** This is a fake setup algorithm that generates a public key and a trapdoor auxiliary information aux.

$\mathsf{FakeCT}(\mathsf{pk}, \mathsf{aux}, X) \rightarrow \widetilde{\mathsf{CT}}_X$**:** This is a fake ciphertext generation algorithm that generates a fake ciphertext $\widetilde{\mathsf{CT}}_X$ under an attribute X.

$\mathsf{FakeSK}(\mathsf{pk}, \mathsf{aux}, P) \rightarrow \widetilde{\mathsf{sk}}_P$**:** This is a fake key generation algorithm that generates a fake secret key $\widetilde{\mathsf{sk}}_P$ for P.

$\mathsf{Reveal}(\mathsf{pk}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m) \rightarrow \widetilde{\mathsf{msk}}$**:** This is a reveal algorithm that generates a fake master secret key $\widetilde{\mathsf{msk}}$.

Roughly speaking, the NCABE security notion requires that the fake public key, master secret key, ciphertext, and secret keys are computationally indistinguishable from the normal public key, master key, ciphertext, and secret keys. It is easy to see that the hybrid encryption approach works in the ABE setting as well. Thus, the goal is achieving an NCABE scheme.

Our NCABE construction follows the RNCE construction based on IND-CPA PKE [CHK05, KNTY19]. However, the crucial difference between the PKE and ABE settings is that, in the ABE setting, adversaries are given many secret keys for queried policies (that is, we consider collusion-resistance). There is an obstacle to achieving collusion resistance because secret keys for policies depend on a master secret key. Note that adversaries can send secret key queries *both before and after* the target ciphertext is given.

First, we explain the RNCE scheme from PKE. Although we explain the 1-bit plaintext case, it is easy to extend to the multi-bit case. The idea is the simple double encryption technique by Naor and Yung [NY90], but we do not need non-interactive zero-knowledge (NIZK). We generate two key pairs $(\mathsf{pk}_0, \mathsf{sk}_0)$ and $(\mathsf{pk}_1, \mathsf{sk}_1)$ and set $\mathsf{pk} := (\mathsf{pk}_0, \mathsf{pk}_1)$, $\mathsf{sk} := \mathsf{sk}_z$, and $\mathsf{aux} = (\mathsf{sk}_0, \mathsf{sk}_1, z^*)$ where $z, z^* \leftarrow \{0, 1\}$. A ciphertext consists of $\mathsf{Enc}(\mathsf{pk}_0, b)$ and $\mathsf{Enc}(\mathsf{pk}_1, b)$. We can decrypt the ciphertext by using sk_z. A fake ciphertext $\widetilde{\mathsf{CT}}$ is $(\mathsf{Enc}(\mathsf{pk}_{z^*}, 0), \mathsf{Enc}(\mathsf{pk}_{1-z^*}, 1))$. To generate a fake

[2] Ananth and Kaleoglu concurrently and independently present essentially the same technique in the uncloneable encryption setting [AK21].

secret key for a plaintext m^*, the reveal algorithm outputs $\mathsf{sk}_{z^*\oplus m^*}$. It is easy to see decrypting $\widetilde{\mathsf{CT}}$ by $\mathsf{sk}_{z^*\oplus m^*}$ yields m^*.

Our NCABE is based on the idea above. That is, we use two key pairs $(\mathsf{pk}_0, \mathsf{msk}_0)$ and $(\mathsf{pk}_1, \mathsf{msk}_1)$ of a normal ABE scheme $\mathsf{ABE}.(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$, and a ciphertext consists of $(\mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_0, X, b), \mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_1, X, b))$ where X is an attribute. Our reveal algorithm outputs $\mathsf{msk}_{z^*\oplus m^*}$ for a plaintext m^* as in the PKE case. The problem is a secret key for a policy P. A naive idea is that a key generation algorithm outputs $\mathsf{sk}_P \leftarrow \mathsf{ABE}.\mathsf{KeyGen}(\mathsf{msk}_z, P)$ where $z \leftarrow \{0,1\}$ is chosen in the setup algorithm, and a fake key generation algorithm outputs $\widetilde{\mathsf{sk}}_P \leftarrow \mathsf{ABE}.\mathsf{KeyGen}(\mathsf{msk}_{z^*\oplus m^*}, P)$. However, this apparently does not work since $\widetilde{\mathsf{sk}}_P$ depends on m^*. Unless $\widetilde{\mathsf{sk}}_P$ is independent of m^*, we cannot use NCABE to achieve ABE with certified deletion because $\mathsf{ske}.\mathsf{sk}$ of SKE with certified deletion is sent *after* a valid certification is generated ($\mathsf{ske}.\mathsf{sk}$ would be a plaintext of ABE in the hybrid encryption). To make a fake key generation be independent of m^*, we need to hide which master secret key is used to generate a secret key for P. If a secret key leaks information about which secret key (extracted from msk_0 or msk_1) is used, we cannot adaptively select a fake master secret key in the reveal algorithm.

IO helps us to overcome this hurdle. Our idea is as follows. A key generation algorithm outputs an obfuscated circuit of a circuit $\mathsf{D}[\mathsf{sk}_z]$ that takes a ciphertext $(\mathsf{abe}.\mathsf{CT}_0, \mathsf{abe}.\mathsf{CT}_1) := (\mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_0, X, b), \mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_1, X, b))$ and outputs $\mathsf{ABE}.\mathsf{Dec}(\mathsf{sk}_z, \mathsf{abe}.\mathsf{CT}_z)$ where $z \leftarrow \{0,1\}$ and $\mathsf{sk}_z \leftarrow \mathsf{ABE}.\mathsf{KeyGen}(\mathsf{msk}_z, P)$ is hard-coded in D. A fake key generation algorithm outputs an obfuscated circuit of a circuit $\mathsf{D}_0[\mathsf{sk}_0]$ that takes $(\mathsf{abe}.\mathsf{CT}_0, \mathsf{abe}.\mathsf{CT}_1)$ and outputs $\mathsf{ABE}.\mathsf{Dec}(\mathsf{sk}_0, \mathsf{abe}.\mathsf{CT}_0)$ where $\mathsf{sk}_0 \leftarrow \mathsf{ABE}.\mathsf{KeyGen}(\mathsf{msk}_0, P)$ is hard-coded in D_0. Note that the fake secret key cannot be used to decrypt a fake ciphertext $(\mathsf{abe}.\mathsf{CT}_{z^*}, \mathsf{abe}.\mathsf{CT}_{1-z^*}) := (\mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_{z^*}, X, 0), \mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_{1-z^*}, X, 1))$ where $z^* \leftarrow \{0,1\}$ since $P(X) = \bot$ must hold by the requirement on ABE security. Since the decryption circuits D and D_0 are obfuscated, adversaries have no idea about which secret key (sk_0 or sk_1) is used for decryption. This idea is inspired by the functional encryption (FE) scheme by Garg et al. [GGH$^+$16].

The final issue is that adversaries can detect whether a secret key is real or fake if they use an invalid ciphertext $(\mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_0, b), \mathsf{ABE}.\mathsf{Enc}(\mathsf{pk}_1, 1-b))$ as an input to the obfuscated circuits. To prevent this attack, we use statistically sound NIZK to check the consistency of double encryption as the FE scheme by Garg et al. [GGH$^+$16]. By the statistical soundness of NIZK, we can guarantee that the obfuscated decryption circuit does not accept invalid ciphertexts, and D and D_0 are functionally equivalent. Note that a secret key for policy P outputs $\bot$ for the target ciphertext since a target attribute X^* in the target ciphertext satisfies $P(X) = \bot$. We do not need the simulation-soundness, unlike the FE scheme by Garg et al. due to the following reason. In the FE scheme, plain PKE schemes are used for the double encryption technique and a secret key sk_0 or sk_1 is hard-coded in a functional decryption key. Before we use PKE security under pk_b, we need to switch decryption from by sk_b to sk_{1-b} by IO security. During this phase, we need to use a fake simulated proof of NIZK. Thus, the simulation-soundness is required. However, in our ABE setting, a secret key for P (not the master secret keys $\mathsf{msk}_0, \mathsf{msk}_1$) is hard-coded in D (or D_0) above. Thanks to the ABE key oracle, sk_0 and sk_1 for P are always available in reductions. We can first use IO security to switch from D to D_0. After that, we change a real NIZK proof into a fake one. Thus, our NCABE scheme does not need the simulation-soundness. This observation enables us to achieve the adaptive security rather than the selective security unlike the FE scheme by Garg et al.[3] See Section 4.2 for the detail. Thus, we can achieve NCABE from IO and OWFs since adaptively secure standard ABE can be constructed from IO and OWFs.

1.4 Technical Overview Part II: Classical Communication Case

We provide an overview of how to achieve privately verifiable and publicly verifiable PKE with certified deletion using classical communication in this section. We note that both of them rely on interactive encryption algorithms.

Privately verifiable construction. For realizing a privately verifiable construction with classical communication, we rely on *NTCF functions* [BCM$^+$18, Mah18]. In this overview, we consider an ideal version, noise-free claw-free permutations for simplicity. A trapdoor claw-free permutation is $f : \{0,1\} \times \{0,1\}^w \rightarrow \{0,1\}^w$ such that (1) $f(0,\cdot)$ and $f(1,\cdot)$ are permutations over $\{0,1\}^w$, (2) given the description of f, it is hard to find x_0 and x_1 such that $f(0,x_0) = f(1,x_1)$, and (3) there is a trapdoor td that enables one to efficiently find x_0 and x_1 such that

[3]In the initial version of this work [NY21], we achieve only the selective security because we use statistical simulation-sound NIZK as the FE scheme by Garg et al. [GGH$^+$16]. We improve the result.

$f(0,x_0) = f(1,x_1) = y$ for any y. In addition, the existing work showed that (a noisy version of) it satisfies a property called the *adaptive hardcore bit property* under the LWE assumption [BCM+18]. To explain this, suppose that one generates the state $\sum_{b,x} |b\rangle |x\rangle |f(b,x)\rangle$, and measures the third register in the computational basis to get a result y. Then the first and second registers collapse to the state $\frac{1}{\sqrt{2}}(|0\rangle |x_0\rangle + |1\rangle |x_1\rangle)$ with $f(0,x_0) = f(1,x_1) = y$. If one measures the state in the computational basis, the measurement outcome is $(0,x_0)$ or $(1,x_1)$. If, on the other hand, one measures the state in the Hadamard basis, the measurement outcome is (e,d) such that $e = d \cdot (x_0 \oplus x_1)$. The adaptive hardcore bit property roughly means that once one gets $(0,x_0)$ or $(1,x_1)$, it cannot output (e,d) such that $d \neq 0$ and $e = d \cdot (x_0 \oplus x_1)$ with probability better than $1/2 + \mathsf{negl}(\lambda)$. Note that this is a tight bound since $e = d \cdot (x_0 \oplus x_1)$ holds with probability $1/2$ if we randomly choose e. Existing works showed that this property can be amplified by parallel repetition [RS19, KNY20]: Specifically, let $(0, x_{i,0})$ and $(1, x_{i,1})$ be the preimages of y_i under f_i for $i \in [n]$ where $n = \omega(\log \lambda)$. Then once one gets a sequence $\{b_i, x_{i,b_i}\}_{i\in[n]}$ for some $b_1\|...\|b_n \in \{0,1\}^n$, it can get a sequence $\{e_i, d_i\}_{i\in[n]}$ such that $d_i \neq 0$ and $e_i = d_i \cdot (x_{i,0} \oplus x_{i,1})$ only with negligible probability.

We use this property to construct an encryption scheme with certified deletion. A natural idea would be as follows: The sender sends n functions $\{f_i\}_{i\in[n]}$ to the receiver, the receiver generates $\{y_i\}_{i\in[n]}$ along with states $\{\frac{1}{\sqrt{2}}(|0\rangle |x_{i,0}\rangle + |1\rangle |x_{i,1}\rangle)\}_{i\in[n]}$ as above and sends $\{y_i\}_{i\in[n]}$ to the sender, and the sender sends receiver a ciphertext CT decryptable only when $\{b_i, x_{i,b_i}\}_{i\in[n]}$ for some $b_1\|...\|b_n \in \{0,1\}^n$ is available. We discuss how to implement such a ciphertext later. We use $\{e_i, d_i\}_{i\in[n]}$ such that $e_i = d_i \cdot (x_{i,0} \oplus x_{i,1})$ as a deletion certificate. The receiver can decrypt the ciphertext by measuring the states in the computational basis, and once it outputs a valid deletion certificate, it must "forget" preimages by the amplified adaptive hardcore property and thus cannot decrypt the ciphertext. This idea can be implemented by a straightforward manner if we generate CT by (extractable) witness encryption [GGSW13, GKP+13] under the corresponding **NP** language. However, since witness encryption is a strong assumption, we want to avoid this. Indeed, we can find the following candidate construction using a hash function H modeled as a random oracle. We set the ciphertext as $\mathsf{CT} := \{\mathsf{CT}_{i,b}\}_{i\in[n],b\in\{0,1\}}$ where $\{m_i\}_{i\in[n]}$ is an n-out-of-n secret sharing of the message m and $\mathsf{CT}_{i,b} := m_i \oplus H(b\|x_{i,b})$. The intuition is that an adversary has to get m_i for all $i \in [n]$ to get m and it has to know $(0, x_{i,0})$ or $(1, x_{i,1})$ to know m_i. Therefore, it seems that any adversary that gets any information of m can be used to extract a sequence $\{b_i, x_{i,b_i}\}_{i\in[n]}$ for some $b_1\|...\|b_n \in \{0,1\}^n$. If this is shown, then it is straightforward to prove that the adversary can get no information of m once it submits a valid deletion certificate by the amplified adaptive hardcore property as explained above. However, turning this intuition into a formal proof seems difficult. A common technique to extract information from adversary's random oracle queries is the one-way to hiding lemma [Unr15, AHU19], which roughly claims that if the adversary distinguishes $H(X)$ from random, then we would get X with non-negligible probability by measuring a randomly chosen query. Here, a problem is that we have to extract n strings $\{b_i, x_{i,b_i}\}_{i\in[n]}$ simultaneously. On the other hand, the extraction by the one-way to hiding lemma disturbs adversary's state by a measurement, and thus we cannot use this technique sequentially.[4]

The difficulty above comes from the fact that the sender cannot know which of $(0, x_{i,0})$ and $(1, x_{i,1})$ the receiver will get, and thus it has to send a ciphertext that can be decrypted in either case. To resolve this issue, we rely on the injective invariance, which roughly says that there is an injective function g that is computationally indistinguishable from f [Mah18]. First, suppose that we just use g instead of f in the above idea. Since g is injective, there is a unique preimage (b_i, x_i) of y_i, in which case the sender knows that the receiver will get $\{(b_i, x_i)\}_{i\in[n]}$ by the standard basis measurement. In this case, the aforementioned problem can be easily resolved by setting $\mathsf{CT} := m \oplus H(b_1\|x_1\|...\|b_n\|x_n)$ as the ciphertext. In this case, it is easy to prove that we can extract $\{b_i, x_i\}_{i\in[n]}$ if an adversary obtains some information of m by applying the standard one-way to hiding lemma. However, an obvious problem is that the deletion certificate no longer works for g since the receiver's state collapses to a classical state after the measurement of $\{y_i\}_{i\in[n]}$ and thus the Hadamard basis measurement results in just uniform bits.

Our idea is to take advantages of both of them. Specifically, the sender sends functions $\{\eta_i\}_{i\in[n]}$, where η_i is the g-type function for $i \in S$ and it is the f-type function for $i \in [n] \setminus S$ with a certain set $S \subset [n]$. The receiver generates a set of states each of which is a superposition of two preimages of a f-type function or a state encoding the unique preimage of a g-type function. The preimages of g-type functions are used for encryption/decryption, and the Hadamard measurement results are used for deletion certificate, whose validity is only checked on positions where f-type functions are used. We also include an encryption of the description of the subset S in the ciphertext

[4] A recent work by Coladangelo, Majenz, and Poremba [CMP20] studied what is called "simultaneous one-way to hiding lemma", but their setting is different from ours and their lemma cannot be used in our setting.

so that a legitimate receiver can know which position should be used in the decryption. More precisely, we set $\mathsf{CT} := (\mathsf{Enc}(S), m \oplus H(\{b_i, x_i\}_{i\in[S]}))$ where Enc is a PKE scheme with the RNC security.[5][6] A deletion certificate $\{e_i, d_i\}_{i\in[n]}$ is valid if we have $d_i \neq 0$ and $e_i = d_i \cdot (x_{i,0} \oplus x_{i,1})$ for all $i \in [n] \setminus S$. For the security proof of this construction, the amplified adaptive hardcore property cannot be directly used, because it is a property about f-type functions whereas the above construction mixes f-type functions and g-type functions, and what we want to have is the mutually-exclusive property between preimages of g-type functions and deletion certificates of f-type functions. To solve the problem, we introduce a new property which we call *the cut-and-choose adaptive hardcore property* (Lemma 5.6). The cut-and-choose adaptive hardcore property intuitively means that once the receiver issues a deletion certificate $\{e_i, d_i\}_{i\in[n]}$ that is valid for all $i \in [n] \setminus S$ before knowing S, it can no longer generate correct preimages $\{b_i, x_i\}_{i\in[S]}$ even if it receives S later. Intuitively, this holds because the only way to obtain such $\{e_i, d_i\}_{i\in[n]}$ before knowing S would be to measure the states in the Hadamard basis for all $i \in [n]$, in which case the receiver should forget all preimages. We show that the cut-and-choose adaptive hardcore property can be reduced to the adaptive hardcore bit property and injective invariance. The new property we show itself is of independent interest, and we believe it will be useful in many other applications of quantum cryptography.

Because the only known construction of NTCF functions [BCM+18, Mah18] assumes the LWE assumption, our construction of the PKE with privately verifiable certified deletion with classical communication is also based on the LWE assumption and our security proof is done in the QROM. We note that the construction only achieves private verification because verification of deletion certificates requires both of two preimages of f-type functions, which cannot be made public.

Publicly verifiable construction. The above construction is not publicly verifiable because the verification of the validity of (e_i, d_i) requires both preimages $x_{i,0}$ and $x_{i,1}$, which cannot be made public. One might notice that the validity check of the preimage can be done publicly, and might suggest the following construction: preimages are used for deletion certificate, and Hadamard measurement outcomes $\{e_i, d_i\}_{i\in[n]}$ are used as the decryption key of the encryption. Because a valid $\{e_i, d_i\}_{i\in[n]}$ is a witness of an **NP** statement, we could use (extractable) witness encryption [GGSW13, GKP+13] to ensure that a receiver can decrypt the message only if it knows a valid $\{e_i, d_i\}_{i\in[n]}$. This idea, however, does not work, because the statement of the witness encryption contains private information (i.e., preimages), and witness encryption ensures nothing about privacy of the statement under which a message is encrypted.

Our idea to solve the problem is to use the one-shot signature [AGKZ20]. Roughly speaking, one-shot signature (with a message space $\{0, 1\}$) enables one to generate a classical public key pk along with a quantum secret key sk, which can be used to generate either of a signature σ_0 for message 0 or σ_1 for message 1, but not both. We note that a signature can be verified publicly.

We combine one-shot signatures with extractable witness encryption.[7] The encryption $\mathsf{Enc}(m)$ of a message m in our construction is a ciphertext of witness encryption of message m under the statement corresponding to the verification of one-shot signature for message 0. The deletion certificate is, on the other hand, a one-shot signature for message 1. Once a valid signature of 1 is issued, a valid signature of 0, which is a decryption key of our witness encryption, is no longer possible to generate due to the security of the one-shot signature. This intuitively ensures the certified deletion security of our construction. Because signatures are publicly verifiable, the verification of our construction is also publicly verifiable. In the actual construction, in order to prevent an adversary from decrypting the ciphertext before issuing the deletion certificate, we add an additional layer of encryption, for which we use RNCE due to the similar reason to that in Section 1.3.

Unfortunately, the only known construction of the one-shot signature needs classical oracles. It is an open question whether we can construct a PKE with publicly verifiable certified deletion with classical communication based on only standard assumptions such as the LWE assumption.

[5]We require Enc to satisfy the RNC security due to a similar reason to that in Section 1.3, which we omit to explain here.

[6]In the actual construction, there is an additional component that is needed for preventing an adversary from decrypting the ciphertext *before* outputting a valid deletion certificate without the decryption key. This is just a security as standard PKE and can be added easily. Thus, we omit this and focus on the security *after* outputting a valid deletion certificate.

[7]We note that a combination of one-shot signatures and extractable witness encryption appeared in the work of Georgiou and Zhandry [GZ20] in a related but different context.

2 Preliminaries

2.1 Notations and Mathematical Tools

We introduce basic notations and mathematical tools used in this paper.

In this paper, $x \leftarrow X$ denotes selecting an element from a finite set X uniformly at random, and $y \leftarrow \mathsf{A}(x)$ denotes assigning to y the output of a probabilistic or deterministic algorithm A on an input x. When we explicitly show that A uses randomness r, we write $y \leftarrow \mathsf{A}(x;r)$. When D is a distribution, $x \leftarrow D$ denotes sampling an element from D. Let $[\ell]$ denote the set of integers $\{1, \cdots, \ell\}$, λ denote a security parameter, and $y \coloneqq z$ denote that y is set, defined, or substituted by z. For a string $s \in \{0,1\}^\ell$, $s[i]$ denotes i-th bit of s. QPT stands for quantum polynomial time. PPT stands for (classical) probabilistic polynomial time. For a subset $S \subseteq W$ of a set W, $\overline{S}$ is the complement of S, i.e., $\overline{S} := W \setminus S$.

A function $f : \mathbb{N} \to \mathbb{R}$ 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 $f(\lambda) \leq \mathsf{negl}(\lambda)$ to denote $f(\lambda)$ being a negligible function. A function $g : \mathbb{N} \to \mathbb{R}$ is a noticeable function if there exist constants c and λ_0 such that for any $\lambda \geq \lambda_0$, $g(\lambda) \geq \lambda^{-c}$. The trace distance between two states ρ and σ is given by $\| \rho - \sigma \|_{tr}$, where $\| A \|_{tr} := \mathrm{Tr}\sqrt{A^\dagger A}$ is the trace norm. We call a function f a density on X if $f : X \to [0,1]$ such that $\sum_{x \in X} f(x) = 1$. For two densities f_0 and f_1 over the same finite domain X, the Hellinger distance between f_0 and f_1 is $\mathbf{H}^2(f_0, f_1) \coloneqq 1 - \sum_{x \in X} \sqrt{f_0(x) f_1(x)}$.

2.2 Cryptographic Tools

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

Public key encryption.

Definition 2.1 (Public Key Encryption (Syntax)). *A public key encryption scheme* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *is a triple of PPT algorithms, a key generation algorithm* KeyGen*, an encryption algorithm* Enc *and a decryption algorithm* Dec*, with plaintext space* $\mathcal{M}$.

$\mathsf{KeyGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a public key* pk *and a secret key* sk.

$\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{CT}$**:** *The encryption algorithm takes as input* pk *and a plaintext* $m \in \mathcal{M}$*, and outputs a ciphertext* CT.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \to m'$ **or** $\bot$**:** *The decryption algorithm takes as input* sk *and* CT*, and outputs a plaintext* m' *or* $\bot$.

Definition 2.2 (Correctness for PKE). *For any* $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$,

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \neq m \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{pk}, m) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 2.3 (OW-CPA security). *Let* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be a PKE scheme. For QPT adversaries* $\mathcal{A}$*, we define the following security experiment* $\mathsf{Exp}^{\mathsf{ow\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda)$.

1. *The challenger generates* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, chooses* $m \leftarrow \mathcal{M}$*, computes* $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$*, and sends* $(\mathsf{pk}, \mathsf{CT})$ *to* $\mathcal{A}$.
2. $\mathcal{A}$ *outputs* m'*. The experiment outputs* 1 *if* $m' = m$ *and otherwise* 0.

We say that the Σ *is OW-CPA secure if for any QPT* $\mathcal{A}$*, it holds that*

$$\mathsf{Adv}^{\mathsf{ow\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda) \coloneqq \Pr[\mathsf{Exp}^{\mathsf{ow\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda) = 1] \leq \mathsf{negl}(\lambda).$$

Note that we assume $1/|\mathcal{M}|$ *is negligible.*

Definition 2.4 (IND-CPA security). *Let $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a PKE scheme. For QPT adversaries $\mathcal{A}$, we define the following security experiment $\mathsf{Exp}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, b)$.*

1. *The challenger generates $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, and sends pk to $\mathcal{A}$.*
2. *$\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M}^2$ to the challenger.*
3. *The challenger computes $\mathsf{CT}_b \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$, and sends CT_b to $\mathcal{A}$.*
4. *$\mathcal{A}$ outputs $b' \in \{0, 1\}$. This is the output of the experiment.*

We say that the Σ is IND-CPA secure if for any QPT $\mathcal{A}$, it holds that

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda) \coloneqq |\Pr[\mathsf{Exp}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \leq \mathsf{negl}(\lambda).$$

It is well-known that the IND-CPA security implies the OW-CPA security. There are many IND-CPA secure PKE schemes against QPT adversaries under standard cryptographic assumptions. A famous one is Regev PKE scheme, which is IND-CPA secure if the learning with errors (LWE) assumption holds against QPT adversaries [Reg09]. See the references for the LWE assumption and constructions of post-quantum secure PKE [Reg09, GPV08].

Definition 2.5 (Indistinguishability Obfuscator [BGI+12]). *A PPT algorithm $i\mathcal{O}$ is an IO for a circuit class $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ if it satisfies the following two conditions.*

Functionality: *For any security parameter $\lambda \in \mathbb{N}$, circuit $C \in \mathcal{C}_\lambda$, and input x, we have that*

$$\Pr[C'(x) = C(x) \mid C' \leftarrow i\mathcal{O}(C)] = 1 \ .$$

Indistinguishability: *For any QPT distinguisher $\mathcal{D}$ and for any pair of circuits $C_0, C_1 \in \mathcal{C}_\lambda$ such that for any input x, $C_0(x) = C_1(x)$ and $|C_0| = |C_1|$, it holds that*

$$\mathsf{Adv}^{\mathsf{io}}_{i\mathcal{O},\mathcal{D}}(\lambda) \coloneqq |\Pr[\mathcal{D}(i\mathcal{O}(C_0)) = 1] - \Pr[\mathcal{D}(i\mathcal{O}(C_1)) = 1]| \leq \mathsf{negl}(\lambda) \ .$$

There exist candidate constructions of IO against QPT adversaries [GP21, WW21, BDGM20, AP20, CHVW19, BGMZ18].

Attribute-based encryption. We review the notion of (key-policy) attribute-based encryption (ABE) [SW05, GPSW06].

Definition 2.6 (Attribute-Based Encryption (Syntax)). *An ABE scheme is a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with plaintext space $\mathcal{M}$, attribute space $\mathcal{X}$, and policy space $\mathcal{P}$.*

$\mathsf{Setup}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{msk})$**:** *The setup algorithm takes as input the security parameter 1^λ and outputs a public key pk and a master secret key msk.*

$\mathsf{KeyGen}(\mathsf{msk}, P) \rightarrow \mathsf{sk}_P$**:** *The key generation algorithm takes as msk and a policy $P \in \mathcal{P}$, and outputs a secret key sk_P.*

$\mathsf{Enc}(\mathsf{pk}, X, m) \rightarrow \mathsf{CT}_X$**:** *The encryption algorithm takes as input pk, an attribute $X \in \mathcal{X}$, and a plaintext $m \in \mathcal{M}$, and outputs a ciphertext CT_X.*

$\mathsf{Dec}(\mathsf{sk}_P, \mathsf{CT}_X) \rightarrow m'$ **or** $\perp$**:** *The decryption algorithm takes as input sk_P and CT_X, and outputs a plaintext m' or $\perp$.*

Definition 2.7 (Perfect Correctness for ABE). *For any $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$, $P \in \mathcal{P}$, and $X \in \mathcal{X}$ such that $P(X) = \top$,*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}_P, \mathsf{CT}_X) \neq m \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{sk}_P \leftarrow \mathsf{KeyGen}(\mathsf{msk}, P) \\ \mathsf{CT}_X \leftarrow \mathsf{Enc}(\mathsf{pk}, X, m) \end{array}\right] = 0.$$

Remark 2.8. Though we allow negligible decryption error for other primitives in this paper, we require perfect correctness for ABE. This is needed for the security proof of the non-committing ABE scheme in Section 4.2.

Definition 2.9 ((Adaptive) IND-CPA Security for ABE). *Let* $\Sigma = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be an ABE scheme with plaintext space* $\mathcal{M}$*, attribute space* $\mathcal{X}$*, and policy space* $\mathcal{P}$*. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, b)$*.*

1. *The challenger computes* $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and sends* pk *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *sends a query* $P \in \mathcal{P}$ *to the challenger and it returns* $\mathsf{sk}_P \leftarrow \mathsf{KeyGen}(\mathsf{msk}, P)$ *to* $\mathcal{A}$*. This process can be repeated polynomially many times.*
3. $\mathcal{A}$ *sends* $X^* \in \mathcal{X}$ *and* $(m_0, m_1) \in \mathcal{M}^2$ *to the challenger where* X^* *must satisfy* $P(X^*) = \bot$ *for all key queries* P *sent so far.*
4. *The challenger computes* $\mathsf{CT}_b \leftarrow \mathsf{Enc}(\mathsf{pk}, X^*, m_b)$ *and sends* CT_b *to* $\mathcal{A}$*.*
5. *Again,* $\mathcal{A}$ *can send key queries* P *that must satisfy* $P(X^*) = \bot$*.*
6. *Finally,* $\mathcal{A}$ *outputs* $b' \in \{0, 1\}$*.*

We say that the Σ *is IND-CPA secure if for any QPT adversary* $\mathcal{A}$*, it holds that*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ind\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \leq \mathsf{negl}(\lambda).$$

If $\mathcal{X} = \{0, 1\}^\ell$ where ℓ is some polynomial and $\mathcal{P}$ consists of all polynomial-sized Boolean circuits, we say ABE for circuits in this paper.

It is known that *selectively secure* ABE for circuits exists under the LWE assumption, which can be upgraded into adaptively secure one by complexity leveraging if we assume subexponential hardness of the LWE problem [GVW15]. Alternatively, if there exist IO and OWFs, there exists adaptively secure functional encryption for **P/poly** [Wat15, ABSV15], which can be trivially downgraded into (adaptively) IND-CPA secure ABE for circuits.

Theorem 2.10. *If one of the following holds, there exists (adaptively) IND-CPA secure (key-policy) ABE scheme for circuits against QPT adversaries.*

- *the LWE problem is subexponentially hard against QPT adversaries [GVW15].*
- *there exist IO and OWFs secure against QPT adversaries [Wat15, ABSV15].*

Encryption with certified deletion. Broadbent and Islam introduced the notion of encryption with certified deletion [BI20]. Their notion is for secret key encryption (SKE). They consider a setting where a secret key is used only once (that is, one time SKE). Although it is easy to extend the definition to reusable secret key setting, we describe the definition for the one-time setting in this section. We provide a definition that is accommodated to the reusable setting in Appendix A.

Definition 2.11 (One-Time SKE with Certified Deletion (Syntax)). *A one-time secret key encryption scheme with certified deletion is a tuple of QPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *with plaintext space* $\mathcal{M}$ *and key space* $\mathcal{K}$*.*

$\mathsf{KeyGen}(1^\lambda) \rightarrow \mathsf{sk}$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a secret key* $\mathsf{sk} \in \mathcal{K}$*.*

$\mathsf{Enc}(\mathsf{sk}, m) \rightarrow \mathsf{CT}$**:** *The encryption algorithm takes as input* sk *and a plaintext* $m \in \mathcal{M}$ *and outputs a ciphertext* CT*.*

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

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

$\mathsf{Vrfy}(\mathsf{sk},\mathsf{cert}) \to \top$ **or** $\bot$**:** *The verification algorithm takes* sk *and* cert *and outputs* $\top$ *or* $\bot$.

Definition 2.12 (Correctness for One-Time SKE with Certified Deletion). *There are two types of correctness. One is decryption correctness and the other is verification correctness.*

Decryption correctness: *For any* $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$,

$$\Pr\left[\mathsf{Dec}(\mathsf{sk},\mathsf{CT}) \neq m \;\middle|\; \begin{array}{l} \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{sk},m) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Verification correctness: *For any* $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$,

$$\Pr\left[\mathsf{Vrfy}(\mathsf{sk},\mathsf{cert}) = \bot \;\middle|\; \begin{array}{l} \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ \mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{sk},m) \\ \mathsf{cert} \leftarrow \mathsf{Del}(\mathsf{CT}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 2.13 (Certified Deletion Security for One-Time SKE). *Let* $\Sigma = (\mathsf{KeyGen},\mathsf{Enc},\mathsf{Dec},\mathsf{Del},\mathsf{Vrfy})$ *be a secret key encryption with certified deletion. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda,b)$.

1. *The challenger computes* $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$.
2. $\mathcal{A}$ *sends* $(m_0,m_1) \in \mathcal{M}^2$ *to the challenger.*
3. *The challenger computes* $\mathsf{CT}_b \leftarrow \mathsf{Enc}(\mathsf{sk},m_b)$ *and sends* CT_b *to* $\mathcal{A}$.
4. $\mathcal{A}$ *sends* cert *to the challenger.*
5. *The challenger computes* $\mathsf{Vrfy}(\mathsf{sk},\mathsf{cert})$. *If the output is* $\bot$, *the challenger sends* $\bot$ *to* $\mathcal{A}$. *If the output is* $\top$, *the challenger sends* sk *to* $\mathcal{A}$.
6. $\mathcal{A}$ *outputs* $b' \in \{0,1\}$.

We say that the Σ *is OT-CD secure if for any QPT* $\mathcal{A}$, *it holds that*

$$\mathsf{Adv}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda,0)=1] - \Pr[\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda,1)=1]| \leq \mathsf{negl}(\lambda).$$

We sometimes call it one-time SKE with certified deletion if it satisfies OT-CD security.

Remark 2.14. Definition 2.13 intuitively means that once the valid certificate is issued, decrypting the ciphertext becomes impossible. One might think that it would be also possible to define the inverse: once the chiphertext is decrypted, the valid certificate can no longer be issued. This property is, however, impossible to achieve due to the decryption correctness (Definition 2.12). In fact, if the quantum decryption algorithm Dec on a quantum ciphertext CT succeeds with probability at least $1-\mathsf{negl}(\lambda)$, then the gentle measurement lemma guarantees that CT is only negligibly disturbed, from which the valid certificate can be issued.

We emphasize that in the existing construction of SKE with certified deletion, a secret key is a classical string though a ciphertext must be a quantum state. Broadbent and Islam prove the following theorem.

Theorem 2.15 ([BI20]). *There exists OT-CD secure SKE with certified deletion with* $\mathcal{M} = \{0,1\}^{\ell_m}$ *and* $\mathcal{K} = \{0,1\}^{\ell_k}$ *where* ℓ_m *and* ℓ_k *are some polynomials, unconditionally.*

Receiver non-committing encryption. We introduce the notion of (public key) receiver non-committing encryption (RNCE) [CFGN96, JL00, CHK05], which is used in Sections 3.2, 5.3 and 6.2. We sometimes simply write NCE to mean RNCE since we consider only RNCE in this paper. See Appendix A for the definition of secret key NCE.

Definition 2.16 (RNCE (Syntax)). *An NCE scheme is a tuple of PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ *with plaintext space* $\mathcal{M}$.

$\mathsf{KeyGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk}, \mathsf{aux})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a key pair* $(\mathsf{pk}, \mathsf{sk})$ *and an auxiliary information* aux.

$\mathsf{Enc}(\mathsf{pk}, m) \to \mathsf{CT}$**:** *The encryption algorithm takes as input* pk *and a plaintext* $m \in \mathcal{M}$ *and outputs a ciphertext* CT.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \to m'$ **or** $\bot$**:** *The decryption algorithm takes as input* sk *and* CT *and outputs a plaintext* m' *or* $\bot$.

$\mathsf{Fake}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux}) \to \widetilde{\mathsf{CT}}$**:** *The fake encryption algorithm takes* pk, sk *and* aux, *and outputs a fake ciphertext* $\widetilde{\mathsf{CT}}$.

$\mathsf{Reveal}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m) \to \widetilde{\mathsf{sk}}$**:** *The reveal algorithm takes* $\mathsf{pk}, \mathsf{sk}, \mathsf{aux}, \widetilde{\mathsf{CT}}$ *and* m, *and outputs a fake secret key* $\widetilde{\mathsf{sk}}$.

Correctness is the same as that of PKE.

Definition 2.17 (Receiver Non-Committing (RNC) Security). *An NCE scheme is RNC secure if it satisfies the following. Let* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ *be an NCE scheme. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda, b)$.

1. *The challenger computes* $(\mathsf{pk}, \mathsf{sk}, \mathsf{aux}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and sends* pk *to* $\mathcal{A}$.
2. $\mathcal{A}$ *sends a query* $m \in \mathcal{M}$ *to the challenger.*
3. *The challenger does the following.*
   - *If* $b = 0$, *the challenger generates* $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{pk}, m)$ *and returns* $(\mathsf{CT}, \mathsf{sk})$ *to* $\mathcal{A}$.
   - *If* $b = 1$, *the challenger generates* $\widetilde{\mathsf{CT}} \leftarrow \mathsf{Fake}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux})$ *and* $\widetilde{\mathsf{sk}} \leftarrow \mathsf{Reveal}(\mathsf{pk}, \mathsf{sk}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m)$ *and returns* $(\widetilde{\mathsf{CT}}, \widetilde{\mathsf{sk}})$ *to* $\mathcal{A}$.
4. $\mathcal{A}$ *outputs* $b' \in \{0, 1\}$.

Let $\mathsf{Adv}^{\mathsf{rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda)$ *be the advantage of the experiment above. We say that the* Σ *is RNC secure if for any QPT adversary, it holds that*

$$\mathsf{Adv}^{\mathsf{rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \leq \mathsf{negl}(\lambda).$$

Theorem 2.18 ([KNTY19]). *If there exists an IND-CPA secure SKE/PKE scheme (against QPT adversaries), there exists an RNC secure secret/public key NCE scheme (against QPT adversaries) with plaintext space* $\{0,1\}^\ell$, *where* ℓ *is some polynomial, respectively.*

Note that Kitagawa, Nishimaki, Tanaka, and Yamakawa [KNTY19] prove the theorem above for the SKE case, but it is easy to extend their theorem to the PKE setting. We also note that the core idea of Kitagawa et al. is based on the observation by Canetti, Halevi, and Katz [CHK05].

Non-interactive zero-knowledge. We review non-interactive zero-knowledge (NIZK) which is used in Section 4.2.

Definition 2.19 (Non-Interactive Zero-Knowledge Proofs (Syntax)). *A non-interactive zero-knowledge (NIZK) proof for an* **NP** *language* $\mathcal{L}$ *consists of PPT algorithms* $(\mathsf{Setup}, \mathsf{Prove}, \mathsf{Vrfy})$.

$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$**:** *The setup algorithm takes as input the security parameter* 1^λ *and outputs a common reference string* crs.

$\mathsf{Prove}(\mathsf{crs}, x, w) \to \pi$**:** *The prover's algorithm takes as input a common reference string* crs*, a statement* x*, and a witness* w *and outputs a proof* π*.*

$\mathsf{Vrfy}(\mathsf{crs}, x, \pi) \to \top$ **or** $\bot$**:** *The verifier's algorithm takes as input a common reference string* crs*, a statement* x*, and a proof* π *and outputs* $\top$ *to indicate acceptance of the proof and* $\bot$ *otherwise.*

A non-interactive proof must satisfy the following requirements.

Completeness: *For all* $\lambda \in \mathbb{N}$ *and all pairs* $(x, w) \in \mathcal{R}_{\mathcal{L}}$*, where* $\mathcal{R}_{\mathcal{L}}$ *is the witness relation corresponding to* $\mathcal{L}$*, we have*

$$\Pr[\mathsf{Vrfy}(\mathsf{crs}, x, \pi) = \top \mid \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda), \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w)] = 1.$$

Statistical Soundness: *For all unbounded time adversaries* $\mathcal{A}$*, if we run* $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$*, then we have*

$$\Pr[x \notin \mathcal{L} \wedge \mathsf{Vrfy}(\mathsf{crs}, x, \pi) = \top \mid (x, \pi) \leftarrow \mathcal{A}(1^\lambda, \mathsf{crs})] \leq \mathsf{negl}(\lambda).$$

(Computational) Zero-Knowledge: *If there exists a PPT simulator* $\mathsf{Sim} = (\mathsf{Sim}_1, \mathsf{Sim}_2)$ *such that for all QPT adversaries* $\mathcal{A}$ *and for all* $(x, w) \in \mathcal{R}_{\mathcal{L}}$*, we have*

$$\left|\Pr\left[\mathcal{A}(1^\lambda, \mathsf{crs}, x, \pi) = 1 \;\middle|\; \begin{array}{l} \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda), \\ \pi \leftarrow \mathsf{Prove}(\mathsf{crs}, x, w) \end{array}\right] - \Pr\left[\mathcal{A}(1^\lambda, \widetilde{\mathsf{crs}}, x, \pi) = 1 \;\middle|\; \begin{array}{l} (\widetilde{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{Sim}_1(1^\lambda, x), \\ \pi \leftarrow \mathsf{Sim}_2(\widetilde{\mathsf{crs}}, \mathsf{td}, x) \end{array}\right]\right| \leq \mathsf{negl}(\lambda).$$

Theorem 2.20. *If one of the following holds, then there exists computational NIZK proof for* **NP** *against QPT adversaries.*

- *the LWE assumption holds against QPT adversaries [PS19].*
- *there exist IO and OWFs [BP15].*

One-shot signature. We review the one-shot signature scheme that will be used in Section 6.2. The one-shot signature scheme was introduced and a construction relative to a classical oracle was given in [AGKZ20].

Definition 2.21 (One-Shot Signature (Syntax)). *A one-shot signature scheme is a tuple of QPT algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy})$ *with the following syntax:*

$\mathsf{Setup}(1^\lambda) \to \mathsf{crs}$**:** *The setup algorithm takes as input a security parameter* 1^λ*, and outputs a classical common reference string* crs*.*

$\mathsf{KeyGen}(\mathsf{crs}) \to (\mathsf{pk}, \mathsf{sk})$**:** *The key generation algorithm takes as input a common reference string* crs*, and outputs a classical public key* pk *and a quantum secret key* sk*.*

$\mathsf{Sign}(\mathsf{sk}, m) \to \sigma$**:** *The signing algorithm takes as input the secret key* sk *and a message* m*, and outputs a classical signature* σ*.*

$\mathsf{Vrfy}(\mathsf{crs}, \mathsf{pk}, \sigma, m) \to \top$ **or** $\bot$**:** *The verification algorithm takes as input the common reference string* crs*, the public key* pk*, the signature* σ*, and the message* m*, and outputs* $\top$ *or* $\bot$*.*

Definition 2.22 (Correctness for One-Shot Signature). *We say that a one-shot signature scheme is correct if*

$$\Pr[\mathsf{Vrfy}(\mathsf{crs}, \mathsf{pk}, \mathsf{Sign}(\mathsf{sk}, m), m) = \top | (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}), \mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)] \geq 1 - \mathsf{negl}(\lambda)$$

for any message m*.*

Definition 2.23 (Security for One-Shot Signature). *We say that a one-shot signature scheme is secure if for any QPT adversary* $\mathcal{A}$*,*

$$\Pr[m_0 \neq m_1 \wedge \mathsf{Vrfy}(\mathsf{crs}, \mathsf{pk}, \sigma_0, m_0) = \top \wedge \mathsf{Vrfy}(\mathsf{crs}, \mathsf{pk}, \sigma_1, m_1) = \top] \leq \mathsf{negl}(\lambda)$$

is satisfied over $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda)$ *and* $(\mathsf{pk}, m_0, \sigma_0, m_1, \sigma_1) \leftarrow \mathcal{A}(\mathsf{crs})$*.*

Witness encryption. We review the notion of witness encryption, which will be used in Section 6.2. The witness encryption was introduced in [GGSW13]. The extractable witness encryption was introduced in [GKP$^+$13].

Definition 2.24 (Witness Encryption for NP (Syntax)). *A witness encryption scheme for an **NP** language $\mathcal{L}$ is a pair of algorithms* $(\mathsf{Enc}, \mathsf{Dec})$ *with the following syntax:*

$\mathsf{Enc}(1^\lambda, x, m) \to \mathsf{CT}$**:** *The encryption algorithm takes as input the security parameter 1^λ, a statement x, and a message m, and outputs a ciphertext* CT.

$\mathsf{Dec}(\mathsf{CT}, w) \to m$ **or** $\perp$**:** *The decryption algorithm takes as input the ciphertext* CT *and a witness w, and outputs m or* $\perp$.

Definition 2.25 (Correctness for Witness Encryption). *We say that a witness encryption scheme is correct if the following holds with overwhelming probability over the randomness of* Enc *and* Dec*. For any $(x, w) \in R_{\mathcal{L}}$, where $R_{\mathcal{L}}$ is the corresponding witness relation, and any message m, it holds that* $\mathsf{Dec}(\mathsf{Enc}(1^\lambda, x, m), w) = m$.

Definition 2.26 (Security for Witness Encryption). *We say that a witness encryption scheme is secure if for any instance $x \notin \mathcal{L}$ and any two messages m_0 and m_1, it holds that for any QPT adversary $\mathcal{A}$,*

$$|\Pr[\mathcal{A}(\mathsf{Enc}(1^\lambda, x, m_0)) = 1] - \Pr[\mathcal{A}(\mathsf{Enc}(1^\lambda, x, m_1)) = 1]| \leq \mathsf{negl}(\lambda).$$

Definition 2.27 (Extractable Security for Witness Encryption). *For any QPT adversary $\mathcal{A}$, polynomial p and messages m_0 and m_1, there is a QPT extractor $\mathcal{E}$ and polynomial q such that for any mixed state* aux *potentially entangled with an external register, if* $|\Pr[\mathcal{A}(\mathsf{aux}, \mathsf{Enc}(1^\lambda, x, m_0)) = 1] - \Pr[\mathcal{A}(\mathsf{aux}, \mathsf{Enc}(1^\lambda, x, m_1)) = 1]| \geq \frac{1}{p(\lambda)}$ *then* $\Pr[(x, \mathcal{E}(1^\lambda, x, \mathsf{aux})) \in R_{\mathcal{L}}] \geq \frac{1}{q(\lambda)}$.

2.3 Noisy Trapdoor Claw-Free Functions

We define noisy trapdoor claw-free function family and its injective invariance following [BCM$^+$18, Mah18].

Definition 2.28 (NTCF Family). *Let $\mathcal{X}$, $\mathcal{Y}$ be finite sets, $\mathcal{D}_{\mathcal{Y}}$ the set of probability distribution over $\mathcal{Y}$, and $\mathcal{K}_{\mathcal{F}}$ a finite set of keys. A family of functions*

$$\mathcal{F} = \{f_{\mathsf{k},b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}\}_{\mathsf{k} \in \mathcal{K}_{\mathcal{F}}, b \in \{0,1\}}$$

is a noisy trapdoor claw-free function (NTCF) family if the following holds.

- **Efficient Function Generation:** *There exists a PPT algorithm* $\mathsf{Gen}_{\mathcal{F}}$ *which takes the security parameter 1^λ as input and outputs a key* $\mathsf{k} \in \mathcal{K}_{\mathcal{F}}$ *and a trapdoor* td.
- **Trapdoor Injective Pair:** *For all keys* $\mathsf{k} \in \mathcal{K}_{\mathcal{F}}$*, the following holds.*
  1. *Trapdoor: For all $b \in \{0,1\}$ and $x \neq x' \in \mathcal{X}$, $\mathrm{Supp}(f_{\mathsf{k},b}(x)) \cap \mathrm{Supp}(f_{\mathsf{k},b}(x')) = \emptyset$. In addition, there exists an efficient deterministic algorithm* $\mathsf{Inv}_{\mathcal{F}}$ *such that for all $b \in \{0,1\}$, $x \in \mathcal{X}$ and $y \in \mathrm{Supp}(f_{\mathsf{k},b}(x))$, $\mathsf{Inv}_{\mathcal{F}}(\mathsf{td}, b, y) = x$.*
  2. *Injective pair: There exists a perfect matching relation $\mathcal{R}_{\mathsf{k}} \subseteq \mathcal{X} \times \mathcal{X}$ such that $f_{\mathsf{k},0}(x_0) = f_{\mathsf{k},1}(x_1)$ if and only if $(x_0, x_1) \in \mathcal{R}_{\mathsf{k}}$.*
- **Efficient Range Superposition:** *For all keys $\mathsf{k} \in \mathcal{K}_{\mathcal{F}}$ and $b \in \{0,1\}$, there exists a function $f'_{\mathsf{k},b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}$ such that the following holds*
  1. *For all $(x_0, x_1) \in \mathcal{R}_{\mathsf{k}}$ and $y \in \mathrm{Supp}(f'_{\mathsf{k},b}(x_b))$, $\mathsf{Inv}_{\mathcal{F}}(\mathsf{td}, b, y) = x_b$ and $\mathsf{Inv}_{\mathcal{F}}(\mathsf{td}, b \oplus 1, y) = x_{b \oplus 1}$.*
  2. *There exists an efficient deterministic algorithm* $\mathsf{Chk}_{\mathcal{F}}$ *that takes as input $\mathsf{k}, b \in \{0,1\}$, $x \in \mathcal{X}$, and $y \in \mathcal{Y}$ and outputs 1 if $y \in \mathrm{Supp}(f'_{\mathsf{k},b}(x))$ and 0 otherwise. Note that this algorithm does not take the trapdoor* td *as input.*

3. *For all* $\mathsf{k} \in \mathcal{K}_{\mathcal{F}}$ *and* $b \in \{0,1\}$,

$$\mathbb{E}_{x \leftarrow \mathcal{X}}[\mathbf{H}^2(f_{\mathsf{k},b}(x), f'_{\mathsf{k},b}(x))] \leq \mathsf{negl}(\lambda).$$

Here $\mathbf{H}^2$ *is the Hellinger distance(See Section 2.1). In addition, there exists a QPT algorithm* $\mathsf{Samp}_{\mathcal{F}}$ *that takes as input* k *and* $b \in \{0,1\}$ *and prepares the quantum state*

$$|\psi'\rangle = \frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \sqrt{(f'_{\mathsf{k},b}(x))(y)} |x\rangle|y\rangle.$$

This property immediately means that

$$\| \, |\psi\rangle\langle\psi| - |\psi'\rangle\langle\psi'| \, \|_{tr} \leq \mathsf{negl}(\lambda)$$

where $|\psi\rangle = \frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \sqrt{(f_{\mathsf{k},b}(x))(y)} |x\rangle|y\rangle$.

- **Adaptive Hardcore Bit:** *For all keys* $\mathsf{k} \in \mathcal{K}_{\mathcal{F}}$*, the following holds. For some integer* w *that is a polynomially bounded function of* λ,

  1. *For all* $b \in \{0,1\}$ *and* $x \in \mathcal{X}$*, there exists a set* $G_{\mathsf{k},b,x} \subseteq \{0,1\}^w$ *such that* $\Pr_{d \leftarrow \{0,1\}^w}[d \notin G_{\mathsf{k},b,x}] \leq \mathsf{negl}(\lambda)$*. In addition, there exists a PPT algorithm that checks for membership in* $G_{\mathsf{k},b,x}$ *given* k, b, x*, and* td.
  2. *There is an efficiently computable injection* $J : \mathcal{X} \to \{0,1\}^w$ *such that* J *can be inverted efficiently on its range, and such that the following holds. Let*

  $$H_{\mathsf{k}} := \{(b, x_b, d, d \cdot (J(x_0) \oplus J(x_1))) \mid b \in \{0,1\}, (x_0, x_1) \in \mathcal{R}_{\mathsf{k}}, d \in G_{\mathsf{k},0,x_0} \cap G_{\mathsf{k},1,x_1}\},$$
  $$\overline{H}_{\mathsf{k}} := \{(b, x_b, d, e) \mid (b, x, d, e \oplus 1) \in H_{\mathsf{k}}\},$$

  then for any QPT $\mathcal{A}$*, it holds that*

  $$|\Pr_{(\mathsf{k},\mathsf{td}) \leftarrow \mathsf{Gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(\mathsf{k}) \in H_{\mathsf{k}}] - \Pr_{(\mathsf{k},\mathsf{td}) \leftarrow \mathsf{Gen}_{\mathcal{F}}(1^\lambda)}[\mathcal{A}(\mathsf{k}) \in \overline{H}_{\mathsf{k}}]| \leq \mathsf{negl}(\lambda).$$

It is known that we can amplify the adaptive hardcore property by parallel repetition in the following sense.

Lemma 2.29 (Amplified Adaptive Hardcore Property). *[RS19, KNY20] Any NTCF family* $\mathcal{F}$ *satisfies the amplified adaptive hardcore property, which means that for any QPT adversary* $\mathcal{A}$ *and* $n = \omega(\log \lambda)$,

$$\Pr\left[\begin{array}{l|l} \forall i \in [n] \ \mathsf{Chk}_{\mathcal{F}}(\mathsf{k}_i, b_i, x_i, y_i) = 1, & (\mathsf{k}_i, \mathsf{td}_i) \leftarrow \mathsf{Gen}_{\mathcal{F}}(1^\lambda) \textit{ for } i \in [n] \\ d_i \in G_{\mathsf{k}_i,0,x_{i,0}} \cap G_{\mathsf{k}_i,1,x_{i,1}}, & \{(b_i, x_i, y_i, d_i, e_i)\}_{i \in [n]} \leftarrow \mathcal{A}(\{\mathsf{k}_i\}_{i \in [n]}) \\ e_i = d_i \cdot (J(x_{i,0}) \oplus J(x_{i,1})) & x_{i,\beta} \leftarrow \mathsf{Inv}_{\mathcal{F}}(\mathsf{td}_i, \beta, y_i) \textit{ for } (i,\beta) \in [n] \times \{0,1\} \end{array}\right] \leq \mathsf{negl}(\lambda).$$

We call the procedures in the right half of the above probability the *amplified adaptive hardcore game* and we say that $\mathcal{A}$ wins the game if the conditions in the left half of the probability are satisfied. By using this terminology, the above lemma says that no QPT adversary wins the amplified adaptive hardcore game with non-negligible probability.

Remark 2.30. A similar lemma is presented in [KNY20] with the difference that the first condition $\mathsf{Chk}_{\mathcal{F}}(\mathsf{k}_i, b_i, x_i, y_i) = 1$ is replaced with $x_i = x_{i,b_i}$. Since $\mathsf{Chk}_{\mathcal{F}}(\mathsf{k}_i, b_i, x_i, y_i) = 1$ implies $x_i = x_{i,b_i}$ by the first and second items of efficient range superposition property, the lemma in the above form also follows.

Next, we define the injective invariance for an NTCF family. For defining this, we first define a trapdoor injective function family.

Definition 2.31 (Trapdoor Injective Function Family). *Let* $\mathcal{X}$*,* $\mathcal{Y}$ *be finite sets,* $\mathcal{D}_{\mathcal{Y}}$ *the set of probability distribution over* $\mathcal{Y}$*, and* $\mathcal{K}_{\mathcal{G}}$ *a finite set of keys. A family of functions*

$$\mathcal{G} = \{g_{\mathsf{k},b} : \mathcal{X} \to \mathcal{D}_{\mathcal{Y}}\}_{\mathsf{k} \in \mathcal{K}_{\mathcal{G}}, b \in \{0,1\}}$$

is a trapdoor injective function family if the following holds.

- **Efficient Function Generation:** *There exists a PPT algorithm* $\mathsf{Gen}_{\mathcal{G}}$ *which takes the security parameter* 1^λ *as input and outputs a key* $\mathsf{k} \in \mathcal{K}_{\mathcal{G}}$ *and a trapdoor* td.
- **Disjoint Trapdoor Injective Pair:** *For all keys* $\mathsf{k} \in \mathcal{K}_{\mathcal{G}}$, *for all* $b, b' \in \{0,1\}$, *and* $x, x' \in \mathcal{X}$, *if* $(b,x) \neq (b',x')$, $\mathrm{Supp}(g_{\mathsf{k},b}(x)) \cap \mathrm{Supp}(g_{\mathsf{k},b'}(x')) = \emptyset$. *Moreover, there exists an efficient deterministic algorithm* $\mathsf{Inv}_{\mathcal{G}}$ *such that for all* $b \in \{0,1\}$, $x \in \mathcal{X}$ *and* $y \in \mathrm{Supp}(g_{\mathsf{k},b}(x))$, $\mathsf{Inv}_{\mathcal{G}}(\mathsf{td}, y) = (b, x)$.
- **Efficient Range Superposition:** *For all keys* $\mathsf{k} \in \mathcal{K}_{\mathcal{G}}$ *and* $b \in \{0,1\}$,
  1. *There exists an efficient deterministic algorithm* $\mathsf{Chk}_{\mathcal{G}}$ *that takes as input* $\mathsf{k}, b \in \{0,1\}$, $x \in \mathcal{X}$, *and* $y \in \mathcal{Y}$ *and outputs 1 if* $y \in \mathrm{Supp}(g_{\mathsf{k},b}(x))$ *and 0 otherwise. Note that this algorithm does not take the trapdoor* td *as input.*
  2. *There exists a QPT algorithm* $\mathsf{Samp}_{\mathcal{G}}$ *that takes as input* k *and* $b \in \{0,1\}$ *and outputs the quantum state*
$$\frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} \sqrt{(g_{\mathsf{k},b}(x))(y)} |x\rangle |y\rangle.$$

Definition 2.32 (Injective Invariance). *We say that a NTCF family* $\mathcal{F}$ *is injective invariant if there exists a trapdoor injective family* $\mathcal{G}$ *such that:*

1. *The algorithms* $\mathsf{Chk}_{\mathcal{F}}$ *and* $\mathsf{Samp}_{\mathcal{F}}$ *are the same as the algorithms* $\mathsf{Chk}_{\mathcal{G}}$ *and* $\mathsf{Samp}_{\mathcal{G}}$. *In this case, we simply write* Chk *and* Samp *to mean them.*
2. *For all QPT algorithm* $\mathcal{A}$, *we have*
$$|\Pr[\mathcal{A}(\mathsf{k}) = 1 : (\mathsf{k}, \mathsf{td}) \leftarrow \mathsf{Gen}_{\mathcal{F}}(1^\lambda)] - \Pr[\mathcal{A}(\mathsf{k}) = 1 : (\mathsf{k}, \mathsf{td}) \leftarrow \mathsf{Gen}_{\mathcal{G}}(1^\lambda)]| \leq \mathsf{negl}(\lambda).$$

Lemma 2.33 ([Mah18]). *If the LWE assumption holds against QPT adversaries, there exists an injective invariant NTCF family.*

2.4 Quantum Random Oracle Model

In Section 5, we rely on the quantum random oracle model (QROM) [BDF+11]. In the QROM, a uniformly random function with certain domain and range is chosen at the beginning, and quantum access to this function is given for all parties including an adversary. Zhandry showed that quantum access to random function can be efficiently simulatable by using so called the compressed oracle technique [Zha19].

We review the one-way to hiding lemma [Unr15, AHU19], which is often useful when analyzing schemes in the QROM. The following form of the lemma is based on [AHU19].

Lemma 2.34 (One-Way to Hiding Lemma [AHU19]). *Let* $S \subseteq \mathcal{X}$ *be random. Let* $G, H : \mathcal{X} \to \mathcal{Y}$ *be random functions satisfying* $\forall x \notin S\ [G(x) = H(x)]$. *Let* z *be a random classical bit string or quantum state. (*S, G, H, z *may have an arbitrary joint distribution.) Let* $\mathcal{A}$ *be an oracle-aided quantum algorithm that makes at most* q *quantum queries. Let* $\mathcal{B}$ *be an algorithm that on input* z *chooses* $i \leftarrow [q]$, *runs* $\mathcal{A}^H(z)$, *measures* $\mathcal{A}$*'s* i*-th query, and outputs the measurement outcome. Then we have*
$$|\Pr[\mathcal{A}^G(z) = 1] - \Pr[\mathcal{A}^H(z) = 1]| \leq 2q\sqrt{\Pr[\mathcal{B}^H(z) \in S]}.$$

Remark 2.35. In [AHU19], z is assumed to be classical. However, as observed in [HXY19], the lemma holds even if z is quantum since any quantum state can be described by an exponentially large classical string, and there is no restriction on the size of z in the lemma.

3 Public Key Encryption with Certified Deletion

In this section, we define the notion of PKE with certified deletion, which is a natural extension of SKE with certified deletion and present how to achieve PKE with certified deletion from OT-CD secure SKE and IND-CPA secure (standard) PKE.

3.1 Definition of PKE with Certified Deletion

The definition of PKE with certified deletion is an extension of SKE with certified deletion. Note that a verification key for verifying a certificate is generated in the encryption algorithm.

Definition 3.1 (PKE with Certified Deletion (Syntax)). *A PKE with certified deletion is a tuple of QPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *with plaintext space* $\mathcal{M}$.

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{pk}, \mathsf{sk})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a classical key pair* $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Enc}(\mathsf{pk}, m) \rightarrow (\mathsf{vk}, \mathsf{CT})$**:** *The encryption algorithm takes as input the public key* pk *and a plaintext* $m \in \mathcal{M}$ *and outputs a classical verification key* vk *and a quantum ciphertext* CT.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \rightarrow m'$ **or** $\bot$**:** *The decryption algorithm takes as input the secret key* sk *and the ciphertext* CT*, and outputs a classical plaintext* m' *or* $\bot$.

$\mathsf{Del}(\mathsf{CT}) \rightarrow \mathsf{cert}$**:** *The deletion algorithm takes as input the ciphertext* CT *and outputs a classical certificate* cert.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) \rightarrow \top$ **or** $\bot$**:** *The verification algorithm takes the verification key* vk *and the certificate* cert*, and outputs* $\top$ *or* $\bot$.

Definition 3.2 (Correctness for PKE with Certified Deletion). *There are two types of correctness. One is decryption correctness and the other is verification correctness.*

Decryption correctness: *For any* $\lambda \in \mathbb{N}$*,* $m \in \mathcal{M}$*,*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \neq m \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (\mathsf{vk}, \mathsf{CT}) \leftarrow \mathsf{Enc}(\mathsf{pk}, m) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Verification correctness: *For any* $\lambda \in \mathbb{N}$*,* $m \in \mathcal{M}$*,*

$$\Pr\left[\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) = \bot \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (\mathsf{vk}, \mathsf{CT}) \leftarrow \mathsf{Enc}(\mathsf{pk}, m) \\ \mathsf{cert} \leftarrow \mathsf{Del}(\mathsf{CT}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 3.3 (Certified Deletion Security for PKE). *Let* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *be a PKE with certified deletion scheme. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{pk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, b)$.

1. *The challenger computes* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and sends* pk *to* $\mathcal{A}$.
2. $\mathcal{A}$ *sends* $(m_0, m_1) \in \mathcal{M}^2$ *to the challenger.*
3. *The challenger computes* $(\mathsf{vk}_b, \mathsf{CT}_b) \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$ *and sends* CT_b *to* $\mathcal{A}$.
4. *At some point,* $\mathcal{A}$ *sends* cert *to the challenger.*
5. *The challenger computes* $\mathsf{Vrfy}(\mathsf{vk}_b, \mathsf{cert})$*. If the output is* $\bot$*, it sends* $\bot$ *to* $\mathcal{A}$*. If the output is* $\top$*, it sends* sk *to* $\mathcal{A}$.
6. $\mathcal{A}$ *outputs its guess* $b' \in \{0, 1\}$.

Let $\mathsf{Adv}^{\mathsf{pk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda)$ *be the advantage of the experiment above. We say that the* Σ *is IND-CPA-CD secure if for any QPT adversary* $\mathcal{A}$*, it holds that*

$$\mathsf{Adv}^{\mathsf{pk\text{-}cert\text{-}del}}_{\mathcal{E},\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{pk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{pk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \leq \mathsf{negl}(\lambda).$$

3.2 PKE with Certified Deletion from PKE and SKE with Certified Deletion

In this section, we present how to construct a PKE scheme with certified deletion from an SKE scheme with certified deletion and an NCE scheme, which can be constructed from standard IND-CPA PKE schemes.

Our PKE Scheme. We construct $\Sigma_{\mathsf{pkcd}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ with plaintext space $\mathcal{M}$ from an SKE with certified deletion scheme $\Sigma_{\mathsf{skcd}} = \mathsf{SKE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ with plaintext space $\mathcal{M}$ and key space $\mathcal{K}$ and a public key NCE scheme $\Sigma_{\mathsf{nce}} = \mathsf{NCE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ with plaintext space $\mathcal{K}$.

$\mathsf{KeyGen}(1^\lambda)$**:**

- Generate $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^\lambda)$ and output $(\mathsf{pk}, \mathsf{sk}) := (\mathsf{nce.pk}, \mathsf{nce.sk})$.

$\mathsf{Enc}(\mathsf{pk}, m)$**:**

- Parse $\mathsf{pk} = \mathsf{nce.pk}$.
- Generate $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$.
- Compute $\mathsf{nce.CT} \leftarrow \mathsf{NCE.Enc}(\mathsf{nce.pk}, \mathsf{ske.sk})$ and $\mathsf{ske.CT} \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.sk}, m)$.
- Output $\mathsf{CT} := (\mathsf{nce.CT}, \mathsf{ske.CT})$ and $\mathsf{vk} := \mathsf{ske.sk}$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$**:**

- Parse $\mathsf{sk} = \mathsf{nce.sk}$ and $\mathsf{CT} = (\mathsf{nce.CT}, \mathsf{ske.CT})$.
- Compute $\mathsf{sk}' \leftarrow \mathsf{NCE.Dec}(\mathsf{nce.sk}, \mathsf{nce.CT})$.
- Compute and output $m' \leftarrow \mathsf{SKE.Dec}(\mathsf{sk}', \mathsf{ske.CT})$.

$\mathsf{Del}(\mathsf{CT})$**:**

- Parse $\mathsf{CT} = (\mathsf{nce.CT}, \mathsf{ske.CT})$.
- Generate $\mathsf{ske.cert} \leftarrow \mathsf{SKE.Del}(\mathsf{ske.CT})$.
- Output $\mathsf{cert} := \mathsf{ske.cert}$.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert})$**:**

- Parse $\mathsf{vk} = \mathsf{ske.sk}$ and $\mathsf{cert} = \mathsf{ske.cert}$.
- Output $b \leftarrow \mathsf{SKE.Vrfy}(\mathsf{ske.sk}, \mathsf{ske.cert})$.

Correctness. The decryption and verification correctness easily follow from the correctness of Σ_{nce} and Σ_{skcd}.

Security. We prove the following theorem.

Theorem 3.4. *If Σ_{nce} is RNC secure and Σ_{skcd} is OT-CD secure, Σ_{pkcd} is IND-CPA-CD secure.*

Proof. Let $\mathcal{A}$ be a QPT adversary and $b \in \{0,1\}$ be a bit. We define the following hybrid game $\mathsf{Hyb}(b)$.

$\mathsf{Hyb}(b)$**:** This is the same as $\mathsf{Exp}^{\mathsf{pk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{pkcd}},\mathcal{A}}(\lambda, b)$ except that the challenger generate the target ciphertext as follows. It generates $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$ and computes $\mathsf{nce.CT}^* \leftarrow \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux})$ and $\mathsf{ske.CT}^* \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_b)$. The target ciphertext is $\mathsf{CT}^* := (\mathsf{nce.CT}^*, \mathsf{ske.CT}^*)$. In addition, we reveal $\widetilde{\mathsf{sk}} \leftarrow \mathsf{Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce.CT}^*, \mathsf{ske.sk})$ instead of $\mathsf{nce.sk}$.

Proposition 3.5. *If Σ_{nce} is RNC secure,* $|\Pr[\mathsf{Exp}^{\mathsf{pk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{pkcd}},\mathcal{A}}(\lambda, b) = 1] - \Pr[\mathsf{Hyb}(b) = 1]| \leq \mathsf{negl}(\lambda)$.

Proof. We construct an adversary $\mathcal{B}_{\mathsf{nce}}$ that breaks the RNC security of Σ_{nce} by assuming that $\mathcal{A}$ distinguishes these two experiments. First, $\mathcal{B}_{\mathsf{nce}}$ is given $\mathsf{nce.pk}$ from the challenger of $\mathsf{Exp}^{\mathsf{rec\text{-}nc}}_{\Sigma_{\mathsf{nce}},\mathcal{B}_{\mathsf{nce}}}(\lambda, b')$ for $b' \in \{0,1\}$. $\mathcal{B}_{\mathsf{nce}}$ generates $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$ and sends $\mathsf{nce.pk}$ to $\mathcal{A}$. When $\mathcal{A}$ sends (m_0, m_1), $\mathcal{B}_{\mathsf{nce}}$ sends $\mathsf{ske.sk}$ to the challenger of $\mathsf{Exp}^{\mathsf{rec\text{-}nc}}_{\Sigma_{\mathsf{nce}},\mathcal{B}_{\mathsf{nce}}}(\lambda, b')$, receives $(\mathsf{nce.CT}^*, \widetilde{\mathsf{sk}})$, and generates $\mathsf{ske.CT} \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_b)$. $\mathcal{B}_{\mathsf{nce}}$ sends $(\mathsf{nce.CT}^*, \mathsf{ske.CT})$ to $\mathcal{A}$ as the challenge ciphertext. At some point, $\mathcal{A}$ outputs cert. If $\mathsf{SKE.Vrfy}(\mathsf{ske.sk}, \mathsf{cert}) = \top$, $\mathcal{B}_{\mathsf{nce}}$ sends $\widetilde{\mathsf{sk}}$ to $\mathcal{A}$. Otherwise, $\mathcal{B}_{\mathsf{nce}}$ sends $\perp$ to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathsf{nce}}$ outputs whatever $\mathcal{A}$ outputs.

- If $b' = 0$, i.e., $(\mathsf{nce.CT}^*, \widetilde{\mathsf{sk}}) = (\mathsf{NCE.Enc}(\mathsf{nce.pk}, \mathsf{ske.sk}), \mathsf{nce.sk})$, $\mathcal{B}_{\mathsf{nce}}$ perfectly simulates $\mathsf{Exp}^{\mathsf{pk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{pkcd}},\mathcal{A}}(\lambda, b)$.
- If $b' = 1$, i.e., $(\mathsf{nce.CT}^*, \widetilde{\mathsf{sk}}) = (\mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}), \mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce.CT}^*, \mathsf{ske.sk}))$, $\mathcal{B}_{\mathsf{nce}}$ perfectly simulates $\mathsf{Hyb}(b)$.

Thus, if $\mathcal{A}$ distinguishes the two experiments, $\mathcal{B}_{\mathsf{nce}}$ breaks the RNC security of Σ_{nce}. This completes the proof. □

Proposition 3.6. *If Σ_{skcd} is OT-CD secure,* $|\Pr[\mathsf{Hyb}(0) = 1] - \Pr[\mathsf{Hyb}(1) = 1]| \le \mathsf{negl}(\lambda)$.

Proof. We construct an adversary $\mathcal{B}_{\mathsf{skcd}}$ that breaks the OT-CD security of Σ_{skcd} assuming that $\mathcal{A}$ distinguishes these two experiments. $\mathcal{B}_{\mathsf{skcd}}$ plays the experiment $\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{skcd}},\mathcal{B}_{\mathsf{skcd}}}(\lambda, b')$ for some $b' \in \{0,1\}$. First, $\mathcal{B}_{\mathsf{skcd}}$ generates $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^\lambda)$ and sends $\mathsf{nce.pk}$ to $\mathcal{A}$. When $\mathcal{A}$ sends (m_0, m_1), $\mathcal{B}_{\mathsf{skcd}}$ sends (m_0, m_1) to the challenger of $\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{skcd}},\mathcal{B}_{\mathsf{skcd}}}(\lambda, b')$, receives $\mathsf{ske.CT}^*$, and generates $\mathsf{nce.}\widetilde{\mathsf{CT}} \leftarrow \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux})$. $\mathcal{B}_{\mathsf{skcd}}$ sends $(\mathsf{nce.}\widetilde{\mathsf{CT}}, \mathsf{ske.CT}^*)$ to $\mathcal{A}$ as the challenge ciphertext. At some point, $\mathcal{A}$ outputs cert. $\mathcal{B}_{\mathsf{skcd}}$ passes cert to the challenger of OT-CD SKE. If the challenger returns $\mathsf{ske.sk}$, $\mathcal{B}_{\mathsf{skcd}}$ generates $\widetilde{\mathsf{sk}} \leftarrow \mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce.}\widetilde{\mathsf{CT}}, \mathsf{ske.sk})$ and sends $\widetilde{\mathsf{sk}}$ to $\mathcal{A}$. Otherwise, $\mathcal{B}_{\mathsf{skcd}}$ sends $\perp$ to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathsf{skcd}}$ outputs whatever $\mathcal{A}$ outputs.

- If $b' = 0$, i.e., $\mathsf{ske.CT}^* = \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_0)$, $\mathcal{B}_{\mathsf{skcd}}$ perfectly simulates $\mathsf{Hyb}(0)$.
- If $b' = 1$, i.e., $\mathsf{ske.CT}^* = \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_1)$, $\mathcal{B}_{\mathsf{skcd}}$ perfectly simulates $\mathsf{Hyb}(1)$.

Thus, if $\mathcal{A}$ distinguishes the two experiments, $\mathcal{B}_{\mathsf{skcd}}$ breaks the OT-CD security. This completes the proof. □

By Propositions 3.5 and 3.6, we immediately obtain Theorem 3.4. □

By Theorems 2.15, 2.18 and 3.4, we immediately obtain the following corollary.

Corollary 3.7. *If there exists IND-CPA secure PKE against QPT adversaries, there exists IND-CPA-CD secure PKE with certified deletion.*

Reusable SKE with certified deletion. We can construct a secret key variant of Σ_{pkcd} above (that is, reusable SKE with certified deletion) by replacing Σ_{nce} with a secret key NCE scheme. We omit the proof since it is almost the same as that of Theorem 3.4. By Theorem 2.18 and the fact that OWFs imply (reusable) SKE [HILL99, GGM86], we also obtain the following theorem.

Theorem 3.8. *If there exists OWF against QPT adversaries, there exists IND-CPA-CD secure SKE with certified deletion.*

See the definition and construction of reusable SKE with certified deletion in Appendix A.

4 Attribute-Based Encryption with Certified Deletion

In this section, we define the notion of attribute-based encryption (ABE) with certified deletion, which is a natural extension of ABE and PKE with certified deletion and present how to achieve ABE with certified deletion from OT-CD secure SKE, IO, and OWFs. In Section 4.1, we present the definition of ABE with certified deletion and non-committing ABE (NCABE), which is a crucial tool to achieve ABE with certified deletion. In Section 4.2, we present how to achieve NCABE from IO and standard ABE. In Section 4.3, we present how to achieve ABE with certified deletion from NCABE and OT-CD secure SKE with certified deletion.

4.1 Definition of ABE with Certified Deletion

The definition of ABE with certified deletion is a natural combination of ABE and PKE with certified deletion.

Definition 4.1 (Attribute-Based Encryption with Certified Deletion (Syntax)). *An ABE scheme with certified deletion is a tuple of QPT algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *with plaintext space* $\mathcal{M}$*, attribute space* $\mathcal{X}$*, and policy space* $\mathcal{P}$*.*

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pk}, \mathsf{msk})$**:** *The setup algorithm takes as input the security parameter* 1^λ *and outputs a public key* pk *and a master secret key* msk.

$\mathsf{KeyGen}(\mathsf{msk}, P) \to \mathsf{sk}_P$**:** *The key generation algorithm takes as input* msk *and a policy* $P \in \mathcal{P}$*, and outputs a secret key* sk_P*.*

$\mathsf{Enc}(\mathsf{pk}, X, m) \to (\mathsf{vk}, \mathsf{CT}_X)$**:** *The encryption algorithm takes as input* pk*, an attribute* $X \in \mathcal{X}$*, and a plaintext* $m \in \mathcal{M}$*, and outputs a verification key* vk *and ciphertext* CT_X*.*

$\mathsf{Dec}(\mathsf{sk}_P, \mathsf{CT}_X) \to m'$ **or** $\bot$**:** *The decryption algorithm takes as input* sk_P *and* CT_X*, and outputs a plaintext* $m' \in \mathcal{M}$ *or* $\bot$*.*

$\mathsf{Del}(\mathsf{CT}_X) \to \mathsf{cert}$**:** *The deletion algorithm takes as input* CT_X *and outputs a certification* cert.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) \to \top$ **or** $\bot$**:** *The verification algorithm takes as input* vk *and* cert*, and outputs* $\top$ *or* $\bot$*.*

Definition 4.2 (Correctness for ABE with Certified Deletion). *There are two types of correctness. One is decryption correctness and the other is verification correctness.*

Decryption correctness: *For any* $\lambda \in \mathbb{N}$*,* $m \in \mathcal{M}$*,* $P \in \mathcal{P}$*, and* $X \in \mathcal{X}$ *such that* $P(X) = \top$*,*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}_P, \mathsf{CT}_X) \neq m \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ \mathsf{sk}_P \leftarrow \mathsf{KeyGen}(\mathsf{msk}, P) \\ (\mathsf{vk}, \mathsf{CT}_X) \leftarrow \mathsf{Enc}(\mathsf{pk}, X, m) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Verification correctness: *For any* $\lambda \in \mathbb{N}$*,* $P \in \mathcal{P}$*,* $X \in \mathcal{X}$*,* $m \in \mathcal{M}$*,*

$$\Pr\left[\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) = \bot \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda) \\ (\mathsf{vk}, \mathsf{CT}_X) \leftarrow \mathsf{Enc}(\mathsf{pk}, X, m) \\ \mathsf{cert} \leftarrow \mathsf{Del}(\mathsf{CT}_X) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 4.3 (ABE Certified Deletion Security). *Let* $\Sigma = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *be an ABE with certified deletion. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}cd}}_{\Sigma,\mathcal{A}}(\lambda, b)$*.*

1. *The challenger computes* $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and sends* pk *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *sends a key query* $P \in \mathcal{P}$ *to the challenger and it returns* $\mathsf{sk}_P \leftarrow \mathsf{KeyGen}(\mathsf{msk}, P)$ *to* $\mathcal{A}$*. This process can be repeated polynomially many times.*
3. $\mathcal{A}$ *sends* $X^* \in \mathcal{X}$ *and* $(m_0, m_1) \in \mathcal{M}^2$ *to the challenger where* X^* *must satisfy* $P(X^*) = \bot$ *for all key queries* P *sent so far.*
4. *The challenger computes* $(\mathsf{vk}_b, \mathsf{CT}_b) \leftarrow \mathsf{Enc}(\mathsf{pk}, X^*, m_b)$ *and sends* CT_b *to* $\mathcal{A}$*.*
5. *Again,* $\mathcal{A}$ *can send key queries* P *that must satisfy* $P(X^*) = \bot$*.*
6. $\mathcal{A}$ *computes* $\mathsf{cert} \leftarrow \mathsf{Del}(\mathsf{CT}_b)$ *and sends* cert *to the challenger.*
7. *The challenger computes* $\mathsf{Vrfy}(\mathsf{vk}_b, \mathsf{cert})$*. If the output is* $\bot$*, the challenger sends* $\bot$ *to* $\mathcal{A}$*. If the output is* $\top$*, the challenger sends* msk *to* $\mathcal{A}$*.*

8. *Again, $\mathcal{A}$ can send key queries P that must satisfy $P(X^*) = \bot$.*[8]

9. *$\mathcal{A}$ outputs its guess $b' \in \{0,1\}$.*

We say that the Σ is IND-CPA-CD secure if for any QPT adversary $\mathcal{A}$, it holds that

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa\text{-}cd\text{-}(}}_{\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}cd}}_{\Sigma,\mathcal{A}}(\lambda,0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}cd}}_{\Sigma,\mathcal{A}}(\lambda,1) = 1]| \leq \mathsf{negl}(\lambda).$$

Next, we define receiver non-committing ABE, which is a receiver non-committing encryption version of ABE.

Definition 4.4 (Receiver Non-Committing Attribute-Based Encryption (Syntax)). *A receiver non-committing (key policy) attributed-based encryption (NCABE) is a tuple of PPT algorithms* $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FakeSetup}, \mathsf{FakeSK}, \mathsf{FakeCT}, \mathsf{Reveal})$ *with plaintext space $\mathcal{M}$, attribute space $\mathcal{X}$, and policy space $\mathcal{P}$.*

$\mathsf{Setup}(1^\lambda) \to (\mathsf{pk}, \mathsf{msk})$**:** *The setup algorithm takes as input the security parameter 1^λ and outputs a public key* pk *and a master secret key* msk.

$\mathsf{KeyGen}(\mathsf{msk}, P) \to \mathsf{sk}_P$**:** *The key generation algorithm takes as input* msk *and a policy $P \in \mathcal{P}$, and outputs a secret key* sk_P.

$\mathsf{Enc}(\mathsf{pk}, X, m) \to \mathsf{CT}$**:** *The encryption algorithm takes as input* pk, *an attribute $X \in \mathcal{X}$, and a plaintext $m \in \mathcal{M}$, and outputs a ciphertext* CT.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \to m'$ **or** $\bot$**:** *The decryption algorithm takes as input* sk *and* CT *and outputs a plaintext $m' \in \mathcal{M}$ or $\bot$.*

$\mathsf{FakeSetup}(1^\lambda) \to (\mathsf{pk}, \mathsf{aux})$**:** *The fake setup algorithm takes as input the security parameter 1^λ, and outputs a public key* pk *and an auxiliary information* aux.

$\mathsf{FakeCT}(\mathsf{pk}, \mathsf{aux}, X) \to \widetilde{\mathsf{CT}}$**:** *The fake encryption algorithm takes* pk, aux, *and $X \in \mathcal{X}$, and outputs a fake ciphertext* $\widetilde{\mathsf{CT}}$.

$\mathsf{FakeSK}(\mathsf{pk}, \mathsf{aux}, P) \to \widetilde{\mathsf{sk}}$**:** *The fake key generation algorithm takes* pk, aux, *and $P \in \mathcal{P}$, and outputs a fake secret key* $\widetilde{\mathsf{sk}}$.

$\mathsf{Reveal}(\mathsf{pk}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m) \to \widetilde{\mathsf{msk}}$**:** *The reveal algorithm takes* $\mathsf{pk}, \mathsf{aux}$, *a fake ciphertext* $\widetilde{\mathsf{CT}}$, *and a plaintext $m \in \mathcal{M}$, and outputs a fake master secret key* $\widetilde{\mathsf{msk}}$.

Correctness is the same as that of ABE.

Definition 4.5 (RNC Security for ABE). *An NCABE scheme is RNC secure if it satisfies the following. Let* $\Sigma = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FakeSetup}, \mathsf{FakeCT}, \mathsf{FakeSK}, \mathsf{Reveal})$ *be an NCABE scheme. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{rnc\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, b)$.

1. *The challenger does the following.*
   - *If $b = 0$, the challenger computes* $(\mathsf{pk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and sends* pk *to $\mathcal{A}$.*
   - *If $b = 1$, the challenger computes* $(\mathsf{pk}, \mathsf{aux}) \leftarrow \mathsf{FakeSetup}(1^\lambda)$ *and sends* pk *to $\mathcal{A}$.*

2. *$\mathcal{A}$ sends a query $P_i \in \mathcal{P}$ to the challenger.*
   - *If $b = 0$, the challenger returns a secret key* $\mathsf{sk}_P \leftarrow \mathsf{KeyGen}(\mathsf{msk}, P)$.
   - *If $b = 1$, the challenger returns a fake secret key* $\widetilde{\mathsf{sk}}_P \leftarrow \mathsf{FakeSK}(\mathsf{pk}, \mathsf{aux}, P)$.

   $\mathcal{A}$ can send polynomially many key queries.

[8]Such queries are useless if $\mathcal{A}$ obtains msk in the previous item, but may be useful if the challenger returns $\bot$ there.

3. *At some point, $\mathcal{A}$ sends the target attribute $X^* \in \mathcal{X}$ and message $m \in \mathcal{M}$ to the challenger where X^* must satisfy $P(X^*) = \bot$ for all key queries P sent so far. The challenger does the following.*
   - *If $b = 0$, the challenger generates $\mathsf{CT}^* \leftarrow \mathsf{Enc}(\mathsf{pk}, X^*, m)$ and returns $(\mathsf{CT}^*, \mathsf{msk})$ to $\mathcal{A}$.*
   - *If $b = 1$, the challenger generates $\widetilde{\mathsf{CT}}^* \leftarrow \mathsf{FakeCT}(\mathsf{pk}, \mathsf{aux}, X^*)$ and $\widetilde{\mathsf{msk}} \leftarrow \mathsf{Reveal}(\mathsf{pk}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m)$ and returns $(\widetilde{\mathsf{CT}}, \widetilde{\mathsf{msk}})$ to $\mathcal{A}$.*
4. *Again, $\mathcal{A}$ can send key queries P that must satisfy $P(X^*) = \bot$.*
5. *$\mathcal{A}$ outputs $b' \in \{0,1\}$.*

We say that Σ is RNC secure if for any QPT adversary, it holds that

$$\mathsf{Adv}^{\mathsf{rnc\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}cpa}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \le \mathsf{negl}(\lambda).$$

4.2 Non-Committing ABE from IO

In this section, we construct NCABE scheme with plaintext space $\{0,1\}^{\ell_m}$, attribute space $\mathcal{X}$ where ℓ_m are some polynomials, and policy space $\mathcal{P}$ from IO for **P/poly** and ABE scheme with plaintext space $\{0,1\}$, attribute space $\mathcal{X}$, and policy space $\mathcal{P}$.

Our NCABE scheme. Let $\Sigma_{\mathsf{abe}} = \mathsf{ABE}.(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA secure ABE scheme on the message space $\{0,1\}$ and Π_{nizk} be a NIZK proof for the **NP** language $\mathcal{L}$ corresponding to the following relation $\mathcal{R}$.

$$\mathcal{R} := \{((\mathsf{pk}, \{\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1}\}_{i\in[\ell_m]}, X), \{(m[i], r_{i,0}, r_{i,1})\}_{i\in[\ell_m]}) \mid \forall i \forall b\ \mathsf{CT}_{i,b} = \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,b}, X, m[i]; r_{i,b})\}$$

where $\mathsf{pk} = \{\mathsf{abe.pk}_{i,0}, \mathsf{abe.pk}_{i,1}\}_{i\in\ell_m}$.

We construct an NCABE scheme $\Sigma_{\mathsf{nce}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FakeSetup}, \mathsf{FakeCT}, \mathsf{FakeSK}, \mathsf{Reveal})$ as follows.

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

1. Generate $(\mathsf{abe.pk}_{i,b}, \mathsf{abe.msk}_{i,b}) \leftarrow \mathsf{ABE.Setup}(1^\lambda)$ for every $i \in [\ell_m]$ and $b \in \{0,1\}$.
2. Choose $z \leftarrow \{0,1\}^{\ell_m}$.
3. Computes $\mathsf{crs} \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$.
4. Output $\mathsf{pk} := (\{\mathsf{abe.pk}_{i,b}\}_{i\in[\ell_m], b\in\{0,1\}}, \mathsf{crs})$ and $\mathsf{msk} := (\mathsf{pk}, \{\mathsf{abe.msk}_{i,z[i]}\}_{i\in[\ell_m]}, z)$.

$\mathsf{KeyGen}(\mathsf{msk}, P)$**:**

1. Parse $\mathsf{msk} = (\mathsf{pk}, \{\mathsf{abe.msk}_{i,z[i]}\}_{i\in[\ell_m]}, z)$.
2. Generate $\mathsf{sk}_i \leftarrow \mathsf{ABE.KeyGen}(\mathsf{abe.msk}_{i,z[i]}, P)$ for every $i \in [\ell_m]$.
3. Generate and output $\mathsf{sk}_P := i\mathcal{O}(\mathsf{D}[\mathsf{crs}, \{\mathsf{sk}_i\}_{i\in[\ell_m]}, z])$, where circuit D is described in Figure 1.

$\mathsf{Enc}(\mathsf{pk}, X, m)$:

1. Parse $\mathsf{pk} = (\{\mathsf{abe.pk}_{i,b}\}_{i\in[\ell_m], b\in\{0,1\}}, \mathsf{crs})$.
2. Generate $\mathsf{CT}_{i,b} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,b}, X, m[i]; r_{i,b})$ for every $i \in [\ell_m]$ and $b \in \{0,1\}$ where $r_{i,b}$ is uniformly chosen from the randomness space for $\mathsf{ABE.Enc}$.
3. Generate $\pi \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, d, w)$ where $d = (\{(\mathsf{abe.pk}_{i,0}, \mathsf{abe.pk}_{i,1}, \mathsf{CT}_{i,0}, \mathsf{CT}_{i,1})\}_{i\in[\ell_m]}, X)$ and $w = (m, \{r_{i,0}, r_{i,1}\}_{i\in[\ell_m]})$.
4. Output $\mathsf{CT}_X := (\{\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1}\}_{i\in[\ell_m]}, \pi)$.

$\mathsf{Dec}(\mathsf{sk}_P, \mathsf{CT}_X)$:

1. Parse $\mathsf{sk}_P = \widetilde{\mathsf{D}}$.
2. Compute and output $m \coloneqq \widetilde{\mathsf{D}}(\mathsf{CT}_X)$.

$\mathsf{FakeSetup}(1^\lambda)$:

1. Generate $(\mathsf{abe.pk}_{i,b}, \mathsf{abe.msk}_{i,b}) \leftarrow \mathsf{ABE.Setup}(1^\lambda)$ for every $i \in [\ell_m]$ and $b \in \{0,1\}$.
2. Choose $z^* \leftarrow \{0,1\}^{\ell_m}$.
3. Computes $(\widetilde{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{Sim}_1(1^\lambda)$.
4. Output $\mathsf{pk} \coloneqq (\{\mathsf{abe.pk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, \widetilde{\mathsf{crs}})$ and $\mathsf{aux} \coloneqq (\mathsf{pk}, \mathsf{td}, \{\mathsf{abe.msk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, z^*)$.

$\mathsf{FakeSK}(\mathsf{pk}, \mathsf{aux}, P)$:

1. Parse $\mathsf{aux} = (\mathsf{pk}, \mathsf{td}, \{\mathsf{abe.msk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, z^*)$.
2. Generate $\mathsf{sk}_i^0 \leftarrow \mathsf{ABE.KeyGen}(\mathsf{abe.msk}_{i,0}, P)$ for every $i \in [\ell_m]$ and set $\mathsf{sk}_P^0 \coloneqq \{\mathsf{sk}_i^0\}_{i\in[\ell_m]}$.
3. Generate and output $\widetilde{\mathsf{sk}} \coloneqq i\mathcal{O}(\mathsf{D}_0[\widetilde{\mathsf{crs}}, \mathsf{sk}_P^0])$, where circuit D_0 is described in Figure 2.

$\mathsf{FakeCT}(\mathsf{pk}, \mathsf{aux}, X)$:

1. Parse $\mathsf{pk} = (\{\mathsf{abe.pk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, \widetilde{\mathsf{crs}})$ and $\mathsf{aux} = (\mathsf{pk}, \mathsf{td}, \{\mathsf{abe.msk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, z^*)$.
2. Compute $\mathsf{CT}^*_{i,z^*[i]} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,z^*[i]}, X, 0)$ and $\mathsf{CT}^*_{i,1-z^*[i]} \leftarrow \mathsf{ABE.Enc}(\mathsf{pk}_{i,1-z^*[i]}, X, 1)$ for every $i \in [\ell_m]$.
3. Compute $\widetilde{\pi} \leftarrow \mathsf{Sim}_2(\widetilde{\mathsf{crs}}, \mathsf{td}, d^*)$ where $d^* = (\{(\mathsf{abe.pk}_{i,0}, \mathsf{abe.pk}_{i,1}, \mathsf{CT}^*_{i,0}, \mathsf{CT}^*_{i,1})\}_{i\in[\ell_m]}, X)$.
4. Outputs $\widetilde{\mathsf{CT}}_X \coloneqq (\{\mathsf{CT}^*_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, \widetilde{\pi})$.

$\mathsf{Reveal}(\mathsf{pk}, \mathsf{aux}, \widetilde{\mathsf{CT}}_X, m)$:

1. Parse $\mathsf{aux} = (\mathsf{pk}, \mathsf{td}, \{\mathsf{abe.msk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, z^*)$.
2. Outputs $\widetilde{\mathsf{msk}} \coloneqq (\mathsf{pk}, \{\mathsf{abe.msk}_{i,z^*[i]\oplus m[i]}\}_{i\in[\ell_m]}, z^* \oplus m)$.

Left-or-Right Decryption Circuit D

Input: A ciphertext CT_X.

Hardwired value: crs, z, and $\{\mathsf{sk}_i\}_{i\in[\ell_m]}$.

1. Parse $\mathsf{CT}_X = (\{\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1}\}_{i\in[\ell_m]}, \pi)$
2. If $\mathsf{NIZK.Vrfy}(\mathsf{crs}, d, \pi) \neq \top$, output $\bot$.
3. Compute $m[i] \leftarrow \mathsf{ABE.Dec}(\mathsf{sk}_i, \mathsf{CT}_{i,z[i]})$ for $i \in [\ell_m]$.
4. Output $m \coloneqq m[1] \| \cdots \| m[\ell_m]$.

Figure 1: The description of the left-or-right decryption circuit

Correctness. Correctness of Σ_{nce} easily follows from correctness of Σ_{abe} and completeness of Π_{nizk}.

Left Decryption Circuit D_0

Input: A ciphertext CT_X.

Hardwired value: $\widetilde{\mathsf{crs}}$ and $\mathsf{sk}_P^0 = \{\mathsf{sk}_i^0\}_{i \in [\ell_m]}$.

```
1. Parse $\mathsf{CT}_X = (\{\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1}\}_{i \in [\ell_m]}, \pi)$
2. If $\mathsf{NIZK.Vrfy}(\widetilde{\mathsf{crs}}, d, \pi) \neq \top$, output $\bot$.
3. Compute $m[i] \leftarrow \mathsf{ABE.Dec}(\mathsf{sk}_i^0, \mathsf{CT}_{i,0})$ for $i \in [\ell_m]$.
4. Output $m := m[1] \| \cdots \| m[\ell_m]$.
```

Figure 2: The description of the left decryption circuit

Security. We prove the following theorem.

Theorem 4.6. *If Σ_{abe} is perfectly correct and IND-CPA secure, $i\mathcal{O}$ is secure IO for **P/poly**, and Π_{nizk} is a NIZK proof system for **NP**, Σ_{nce} is RNC secure NCABE.*

Proof. Let $\mathcal{A}$ be a QPT adversary. We define the following sequence of hybrid games.

- Hyb_0: This is the same as $\mathsf{Exp}^{\mathsf{rnc\text{-}cpa}}_{\Sigma_{\mathsf{nce}},\mathcal{A}}(\lambda, 0)$. Let X^* and m be the target attribute and message, respectively, as in Definition 4.5.
- Hyb_1: This is the same as Hyb_0 except that the challenger uses the circuit $\mathsf{D}_0[\widetilde{\mathsf{crs}}, \mathsf{sk}_P^0]$ instead of $\mathsf{D}[\mathsf{crs}, \{\mathsf{sk}_i\}_{i \in [\ell_m]}, z]$ to generate secret keys for key queries. That is, it returns $\widetilde{\mathsf{sk}} = i\mathcal{O}(\mathsf{D}_0[\widetilde{\mathsf{crs}}, \mathsf{sk}_P^0])$ instead of $\mathsf{sk} = i\mathcal{O}(\mathsf{D}[\mathsf{crs}, \{\mathsf{sk}_i\}_{i \in [\ell_m]}, z])$. This change is indistinguishable by the IO security and statistical soundness of Π_{nizk}. Note that secret keys do not depend on z in this game.
- Hyb_2: This is the same as Hyb_1 except that the challenger generates a common reference string and proof of the NIZK by using simulators. That is, it generates $(\widetilde{\mathsf{crs}}, \mathsf{td}) \leftarrow \mathsf{Sim}_1(1^\lambda)$ and $\widetilde{\pi} \leftarrow \mathsf{Sim}_2(\widetilde{\mathsf{crs}}, \mathsf{td}, d^*)$ where $d^* = (\{\mathsf{abe.pk}_{i,0}, \mathsf{abe.pk}_{i,1}, \mathsf{CT}^*_{i,0}, \mathsf{CT}^*_{i,1}\}_{i \in [\ell_m]}, X^*)$. This change is indistinguishable by the computational zero-knowledge property of Π_{nizk}.
- Hyb_3: This is the same as Hyb_2 except that the challenger generates an inconsistent target ciphertext. That is, it generates $\mathsf{CT}^*_{i,1-z[i]} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,1-z[i]}, X^*, 1-m[i])$ and $\mathsf{CT}^*_{i,z[i]} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,z[i]}, X^*, m[i])$ instead of double encryption of $m[i]$ for all i. Note that the NIZK proof in the target ciphertext is generated by the simulator in this game.
- Hyb_4: This is the same as Hyb_3 except that the challenger chooses $z^* \leftarrow \{0,1\}^{\ell_m}$, computes $\mathsf{CT}^*_{i,z[i]^*} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,z^*[i]}, X^*, 0)$ and $\mathsf{CT}^*_{i,1-z^*[i]} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,1-z^*[i]}, X^*, 1)$, and sets $\widetilde{\mathsf{msk}} := (z^* \oplus m, \{\mathsf{msk}_{i,z[i]^* \oplus m[i]}\}_{i \in [\ell_m]})$ as a master secret key.

We prove Propositions 4.7 to 4.10.

Proposition 4.7. *If Π_{abe} is perfectly correct, Π_{nizk} is statistically sound, and $i\mathcal{O}$ is secure,* $|\Pr[\mathsf{Hyb}_0 = 1] - \Pr[\mathsf{Hyb}_1 = 1]| \leq \mathsf{negl}(\lambda)$.

Proof. We define more hybrid games. Let q be the total number of key queries.

Hyb_0^j**:** This is the same as Hyb_0 except that

- for $j < k \leq q$, the challenger generates $\mathsf{sk} = i\mathcal{O}(\mathsf{D}[\mathsf{crs}, \{\mathsf{sk}_i\}_{i \in [\ell_m]}, z])$ for the k-th key query.
- for $1 \leq k \leq j$, the challenger generates $\widetilde{\mathsf{sk}} = i\mathcal{O}(\mathsf{D}_0[\widetilde{\mathsf{crs}}, \mathsf{sk}_P^0])$ for the k-th key query.

Clearly, $\mathsf{Hyb}_0^0 = \mathsf{Hyb}_0$ and $\mathsf{Hyb}_0^q = \mathsf{Hyb}_1$.

Let $\mathsf{Invalid}$ be an event that there exists $d^\dagger \notin \mathcal{L}$ and $\pi^\dagger$ such that $\mathsf{NIZK.Vrfy}(\mathsf{crs}, d^\dagger, \pi^\dagger) = \top$. By statistical soundness of Π_{nizk}, this happens with negligible probability. If $\mathsf{Invalid}$ does not occur, by the definitions of D and D_0 and perfect correctness of Π_{abe}, their functionalities are equivalent for all inputs Therefore, adversary's distinguishing advantage of obfuscation of these two circuits is negligible by the iO security. The difference between Hyb_1^{j-1} and Hyb_1^{j} is that the j-th key query answer is generated by D_0 instead of D. Therefore we have $|\Pr[\mathsf{Hyb}_i^{j-1} = 1] - \Pr[\mathsf{Hyb}_i^{j} = 1]| \le \mathsf{negl}(\lambda)$. By a standard hybrid argument, we obtain Proposition 4.7. $\square$

Proposition 4.8. *If* Π_{nizk} *is computationally zero-knowledge,* $|\Pr[\mathsf{Hyb}_1 = 1] - \Pr[\mathsf{Hyb}_2 = 1]| \le \mathsf{negl}(\lambda)$.

Proof. The only difference of these two games is how to generate the common reference string and proof of NIZK. Thus, there is a straightforward reduction to the zero-knowledge property of Π_{nizk}. Specifically, we assume that $|\Pr[\mathsf{Hyb}_1 = 1] - \Pr[\mathsf{Hyb}_2 = 1]|$ is non-negligible and construct an adversary $\mathcal{B}_{\mathsf{nizk}}$ that breaks the zero-knowledge property of Π_{nizk}.

Let $d = (\{(\mathsf{abe.pk}_{i,0}, \mathsf{abe.pk}_{i,1}, \mathsf{CT}_{i,0}, \mathsf{CT}_{i,1})\}_{i\in[\ell_m]}, X^*)$ and $w = (m, \{r_{i,0}, r_{i,1}\}_{i\in[\ell_m]})$ be as in Hyb_1 (or Hyb_2, equivalently). $\mathcal{B}_{\mathsf{nizk}}$ is given (crs^*, π^*) which is generated by the real setup and proving algorithms or simulators. $\mathcal{B}_{\mathsf{nizk}}$ runs Hyb_1 for $\mathcal{A}$ while embedding (crs^*, π^*) in the appropriate part. Finally, $\mathcal{B}_{\mathsf{nizk}}$ outputs whatever $\mathcal{A}$ outputs.

- If (crs^*, π^*) is the real one, i.e., it is generated by $\mathsf{crs}^* \leftarrow \mathsf{NIZK.Setup}(1^\lambda)$ and $\pi^* \leftarrow \mathsf{NIZK.Prove}(\mathsf{crs}, d, w)$, $\mathcal{B}_{\mathsf{nizk}}$ perfectly simulates Hyb_1.
- If (crs^*, π^*) is the simulated one, i.e., it is generated by $(\mathsf{crs}^*, \mathsf{td}) \leftarrow \mathsf{Sim}_1(1^\lambda)$ and $\pi^* \leftarrow \mathsf{Sim}_2(\mathsf{crs}^*, \mathsf{td}, d)$, $\mathcal{B}_{\mathsf{nizk}}$ perfectly simulates Hyb_1.

Thus, if $\mathcal{A}$ distinguishes these two hybrids, $\mathcal{B}_{\mathsf{nizk}}$ breaks the computational zero-knowledge property of Π_{nizk}. This completes the proof. $\square$

Proposition 4.9. *If* Π_{abe} *is IND-CPA secure,* $|\Pr[\mathsf{Hyb}_2 = 1] - \Pr[\mathsf{Hyb}_3 = 1]| \le \mathsf{negl}(\lambda)$.

Proof. We define more hybrid games. Recall ℓ_m is the length of plaintexts.

Hyb_2^j: This is the same as Hyb_2 except that

- for $j < i \le \ell_m$, the challenger generates $\mathsf{CT}_{i,b}^* \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,b}, X^*, m[i])$ for $b \in \{0,1\}$.
- for $1 \le i \le j$, the challenger generates $\mathsf{CT}_{i,1-z[i]}^* \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,1-z[i]}, X^*, 1-m[i])$ and $\mathsf{CT}_{i,z[i]}^* \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{i,z[i]}, m[i])$.

Clearly, $\mathsf{Hyb}_2^0 = \mathsf{Hyb}_2$ and $\mathsf{Hyb}_2^{\ell_m} = \mathsf{Hyb}_3$.

The difference between Hyb_2^j and Hyb_2^{j-1} is the j-th component of the target ciphertext is valid or invalid. We can show that this is indistinguishable by observing that the master secret key is set to be $\mathsf{msk} := (\{\mathsf{abe.msk}_{i,z[i]}\}_{i\in[\ell_m]}, z)$ in these games and $\{\mathsf{msk}_{j,1-z[i]}\}_{i\in[\ell_m]}$ is never revealed to the adversary. Specifically, we can construct an adversary $\mathcal{B}_{\mathsf{abe}}$ that breaks IND-CPA security of Σ_{abe} under key $\mathsf{abe.pk}_{j,1-z[j]}$ assuming that $\mathcal{A}$ distinguishes these two games.

$\mathcal{B}_{\mathsf{abe}}$ receives $\mathsf{abe.pk}$ from the challenger of $\mathsf{Exp}^{\mathsf{ind\text{-}cpa}}_{\Sigma_{\mathsf{abe}},\mathcal{B}_{\mathsf{abe}}}(\lambda, b')$ for $b' \in \{0,1\}$, and sets $\mathsf{abe.pk}_{j,1-z[j]} := \mathsf{abe.pk}$. For other public keys (that is, $\{\mathsf{abe.pk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}} \setminus \{\mathsf{abe.pk}_{j,1-z[j]}\}$) and crs, $\mathcal{B}_{\mathsf{abe}}$ generates them by itself. $\mathcal{B}_{\mathsf{abe}}$ sends $\mathsf{pk} := (\{\mathsf{abe.pk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}, \mathsf{crs})$ to $\mathcal{A}$. When the distinguisher sends a key query P, $\mathcal{B}_{\mathsf{abe}}$ passes P to the challenger and receives $\mathsf{sk}_{j,1-z[j]} \leftarrow \mathsf{ABE.KeyGen}(\mathsf{msk}_{j,1-z[j]}, P)$.[9] For other secret keys for P (that is, $\{\mathsf{sk}_{i,b} \leftarrow \mathsf{ABE.KeyGen}(\mathsf{msk}_{i,b}, P)\}_{i\in[\ell_m],b\in\{0,1\}} \setminus \{\mathsf{sk}_{j,1-z[j]}\}$), $\mathcal{B}_{\mathsf{abe}}$ generates them by itself since it has $\{\mathsf{msk}_{i,b}\}_{i\in[\ell_m],b\in\{0,1\}}$ except for $\mathsf{msk}_{j,1-z[j]}$. Thus, $\mathcal{B}_{\mathsf{abe}}$ can compute $\widetilde{\mathsf{sk}} = i\mathcal{O}(\mathsf{D}_0[\widetilde{\mathsf{crs}}, \mathsf{sk}_P^0])$. At some point, $\mathcal{A}$ declares target attribute X^* and message m. $\mathcal{B}_{\mathsf{abe}}$ sends $(m[j], 1-m[j])$ to the challenger and receives $\mathsf{CT}_{j,1-z[j]}^*$. For $(i,b) \in [\ell_m] \times \{0,1\} \setminus (j, 1-z[j])$, $\mathcal{B}_{\mathsf{abe}}$ generates $\mathsf{CT}_{j,z[j]}^* \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{j,z[j]}, X^*, m[j])$ and $\{\mathsf{CT}_{i,b}\}_{i\in[\ell_m]\setminus\{j\},b\in\{0,1\}}$ as in Hyb_2^j and Hyb_2^{j-1}. Note that the difference between two games is the j-th component (and in particular $(j, 1-z[j])$ part) of the target ciphertext. Again, $\mathcal{B}_{\mathsf{abe}}$ simulates answers for secret key queries as above. $\mathcal{B}_{\mathsf{abe}}$ outputs whatever $\mathcal{A}$ outputs.

[9]In fact, $\mathcal{B}_{\mathsf{abe}}$ need not query the challenger when $z[j] = 0$ since $\mathsf{sk}_{j,1}$ is not needed for generating $\mathsf{D}_0[\widetilde{\mathsf{crs}}, \mathsf{sk}_P^0]$.

- If $b' = 0$, i.e., $\mathsf{CT}^*_{j,1-z[j]} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{j,1-z[j]}, X^*, m^*[j])$, $\mathcal{B}_{\mathsf{abe}}$ perfectly simulates Hyb_i^{j-1}.
- If $b' = 1$, i.e., $\mathsf{CT}^*_{j,1-z[j]} \leftarrow \mathsf{ABE.Enc}(\mathsf{abe.pk}_{j,1-z[j]}, X^*, 1 - m^*[j])$, $\mathcal{B}_{\mathsf{abe}}$ perfectly simulates Hyb_i^{j}.

Thus, if $\mathcal{A}$ distinguishes these two games, $\mathcal{B}_{\mathsf{abe}}$ breaks IND-CPA security of Σ_{abe}. This completes the proof. $\square$

Proposition 4.10. $\Pr[\mathsf{Hyb}_3 = 1] = \Pr[\mathsf{Hyb}_4 = 1]$.

Proof. This is a conceptual change. The advantage of distinguishing these two games is 0 since we can see that these two games are identical if we set $z := z^* \oplus m^*$. Note that secret keys do not depend on z in these games. $\square$

Clearly, $\mathsf{Hyb}_4 = \mathsf{Exp}^{\mathsf{rnc\text{-}cpa}}_{\Sigma_{\mathsf{nce}},\mathcal{A}}(\lambda, 1)$. Therefore, we complete the proof by Propositions 4.7 to 4.10. $\square$

4.3 ABE with Certified Deletion from NCABE and SKE with Certified Deletion

In this section, we construct ABE with certified deletion from NCABE and OT-CD secure SKE with certified deletion.

Our ABE with certified deletion scheme. We construct an ABE with certified deletion scheme $\Sigma_{\mathsf{cd}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ with plaintext space $\mathcal{M}$, attribute space $\mathcal{X}$, and policy space $\mathcal{P}$ from an NCABE scheme $\Sigma_{\mathsf{nce}} = \mathsf{NCE}.(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FakeSetup}, \mathsf{FakeSK}, \mathsf{FakeCT}, \mathsf{Reveal})$ with plaintext space $\{0,1\}^\ell$, attribute space $\mathcal{X}$, and policy space $\mathcal{P}$ and an SKE with certified deletion scheme $\Sigma_{\mathsf{skcd}} = \mathsf{SKE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ with plaintext space $\mathcal{M}$ and key space $\{0,1\}^\ell$.

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

- Generate $(\mathsf{nce.pk}, \mathsf{nce.msk}) \leftarrow \mathsf{NCE.Setup}(1^\lambda)$.
- Output $(\mathsf{pk}, \mathsf{msk}) := (\mathsf{nce.pk}, \mathsf{nce.msk})$.

$\mathsf{KeyGen}(\mathsf{msk}, P)$:

- Generate $\mathsf{nce.sk}_P \leftarrow \mathsf{NCE.KeyGen}(\mathsf{nce.msk}, P)$ and output $\mathsf{sk}_P := \mathsf{nce.sk}_P$.

$\mathsf{Enc}(\mathsf{pk}, X, m)$:

- Parse $\mathsf{pk} = \mathsf{nce.pk}$.
- Generate $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$.
- Compute $\mathsf{nce.CT}_X \leftarrow \mathsf{NCE.Enc}(\mathsf{nce.pk}, X, \mathsf{ske.sk})$ and $\mathsf{ske.CT} \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.sk}, m)$.
- Output $\mathsf{CT}_X := (\mathsf{nce.CT}_X, \mathsf{ske.CT})$ and $\mathsf{vk} := \mathsf{ske.sk}$.

$\mathsf{Dec}(\mathsf{sk}_P, \mathsf{CT}_X)$:

- Parse $\mathsf{sk}_P = \mathsf{nce.sk}_P$ and $\mathsf{CT}_X = (\mathsf{nce.CT}_X, \mathsf{ske.CT})$.
- Compute $\mathsf{sk}' \leftarrow \mathsf{NCE.Dec}(\mathsf{nce.sk}_P, \mathsf{nce.CT}_X)$.
- Compute and output $m' \leftarrow \mathsf{SKE.Dec}(\mathsf{sk}', \mathsf{ske.CT})$.

$\mathsf{Del}(\mathsf{CT})$:

- Parse $\mathsf{CT}_X = (\mathsf{nce.CT}_X, \mathsf{ske.CT})$.
- Generate $\mathsf{ske.cert} \leftarrow \mathsf{SKE.Del}(\mathsf{ske.CT})$.
- Output $\mathsf{cert} := \mathsf{ske.cert}$.

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

- Parse $\mathsf{vk} = \mathsf{ske.sk}$ and $\mathsf{cert} = \mathsf{ske.cert}$.
- Output $b \leftarrow \mathsf{SKE.Vrfy}(\mathsf{ske.sk}, \mathsf{ske.cert})$.

Correctness. Correctness easily follows from correctness of Σ_{skcd} and Σ_{nce}.

Theorem 4.11. *If Σ_{nce} is RNC secure ABE and Σ_{skcd} is OT-CD secure, Σ_{cd} is IND-CPA-CD secure ABE.*

Proof. Let $\mathcal{A}$ be a QPT adversary and b be a bit. We define the following hybrid game $\mathsf{Hyb}(b)$.

$\mathsf{Hyb}(b)$**:** This is the same as $\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}cd}}_{\Sigma_{\mathsf{cd}},\mathcal{A}}(\lambda, b)$ except for the following differences:

1. The challenger generates the public key as $(\mathsf{nce.pk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.FakeSetup}(1^\lambda)$.
2. The challenger generates the challenge ciphertext as follows. It generates $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$, $\mathsf{nce.CT}^*_{X^*} \leftarrow \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.aux}, X^*)$, and $\mathsf{ske.CT}^* \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_b)$. The challenge ciphertext is $\mathsf{CT}^*_{X^*} := (\mathsf{nce.CT}^*_{X^*}, \mathsf{ske.CT}^*)$.
3. The challenger generates secret keys as $\mathsf{nce}.\widetilde{\mathsf{sk}}_P \leftarrow \mathsf{NCE.FakeSK}(\mathsf{nce.pk}, \mathsf{nce.aux}, P)$.
4. The challenger reveals $\mathsf{nce}.\widetilde{\mathsf{msk}} \leftarrow \mathsf{Reveal}(\mathsf{nce.pk}, \mathsf{nce.aux}, \mathsf{nce.CT}^*_X, \mathsf{ske.sk})$ instead of $\mathsf{nce.msk}$.

Proposition 4.12. *If Σ_{nce} is RNC secure, $|\Pr[\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}cd}}_{\Sigma_{\mathsf{cd}},\mathcal{A}}(\lambda, b) = 1] - \Pr[\mathsf{Hyb}(b) = 1]| \le \mathsf{negl}(\lambda)$.*

Proof. We construct an adversary $\mathcal{B}_{\mathsf{nce}}$ that breaks the RNC security of Σ_{nce} by using $\mathcal{A}$ that distinguishes them.

$\mathcal{B}_{\mathsf{nce}}$ receives $\mathsf{nce.pk}$ from the challenger of $\mathsf{Exp}^{\mathsf{rnc\text{-}cpa}}_{\Sigma_{\mathsf{nce}},\mathcal{B}_{\mathsf{nce}}}(\lambda, b')$ for $b' \in \{0,1\}$ and sends $\mathsf{nce.pk}$ to $\mathcal{A}$. When $\mathcal{A}$ makes a key query P, $\mathcal{B}_{\mathsf{nce}}$ passes P to the challenger, receives sk_P, and passes it to $\mathcal{A}$. At some point, $\mathcal{A}$ sends the target attribute X^* and messages (m_0, m_1). $\mathcal{B}_{\mathsf{nce}}$ generates $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$, sends the target attribute X^* and message $\mathsf{ske.sk}$ to the challenger, and receives $\mathsf{nce.CT}^*_{X^*}$ and $\mathsf{nce.msk}^*$ from the challenger. $\mathcal{B}_{\mathsf{nce}}$ generates $\mathsf{ske.CT} \leftarrow \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_b)$ and sends $\mathsf{CT}^*_{X^*} := (\mathsf{nce.CT}^*_{X^*}, \mathsf{ske.CT})$ to $\mathcal{A}$. Again, when $\mathcal{A}$ makes a key query P, $\mathcal{B}_{\mathsf{nce}}$ responds similarly as above. At some point, $\mathcal{A}$ sends a certificate cert. If $\mathsf{SKE.Vrfy}(\mathsf{ske.sk}, \mathsf{cert}) = \top$, $\mathcal{B}_{\mathsf{nce}}$ sends $\mathsf{nce.msk}^*$ to $\mathcal{A}$. Otherwise, $\mathcal{B}_{\mathsf{nce}}$ sends $\bot$ to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathsf{nce}}$ outputs whatever $\mathcal{A}$ outputs.

We can see that $\mathcal{B}_{\mathsf{nce}}$ perfectly simulates $\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}cd}}_{\Sigma_{\mathsf{cd}},\mathcal{A}}(\lambda, b)$ if $b' = 0$ and $\mathsf{Hyb}(b)$ if $b' = 1$. Thus, if $\mathcal{A}$ distinguishes the two hybrids, $\mathcal{B}_{\mathsf{nce}}$ breaks the RNC security. This completes the proof. ☐

Proposition 4.13. *If Σ_{skcd} is OT-CD secure, $|\Pr[\mathsf{Hyb}(0) = 1] - \Pr[\mathsf{Hyb}(1) = 1]| \le \mathsf{negl}(\lambda)$.*

Proof. We construct an adversary $\mathcal{B}_{\mathsf{skcd}}$ that breaks the OT-CD security of Σ_{skcd} assuming that $\mathcal{A}$ distinguishes these two hybrids.

$\mathcal{B}_{\mathsf{skcd}}$ plays the experiment $\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{skcd}},\mathcal{B}_{\mathsf{skcd}}}(\lambda, b')$ for some $b' \in \{0,1\}$. $\mathcal{B}_{\mathsf{skcd}}$ generates $(\mathsf{nce.pk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.Setup}(1^\lambda)$ and sends $\mathsf{nce.pk}$ to $\mathcal{A}$. When $\mathcal{A}$ sends a key query P, $\mathcal{B}_{\mathsf{skcd}}$ generates $\widetilde{\mathsf{sk}}_P \leftarrow \mathsf{NCE.FakeSK}(\mathsf{nce.pk}, \mathsf{nce.aux}, P)$ and returns it to $\mathcal{A}$. When $\mathcal{A}$ sends the target attribute X^* and messages (m_0, m_1), $\mathcal{B}_{\mathsf{skcd}}$ sends (m_0, m_1) to the challenger of $\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{skcd}},\mathcal{B}_{\mathsf{skcd}}}(\lambda, b')$, receives $\mathsf{ske.CT}^*$ and generates $\mathsf{nce}.\widetilde{\mathsf{CT}}_X \leftarrow \mathsf{NCE.FakeCT}(\mathsf{nce.pk}, \mathsf{nce.aux}, X^*)$. $\mathcal{B}_{\mathsf{skcd}}$ sends $(\mathsf{nce}.\widetilde{\mathsf{CT}}_X, \mathsf{ske.CT}^*)$ to $\mathcal{A}$ as the challenge ciphertext. Again, when $\mathcal{A}$ makes a key query P, $\mathcal{B}_{\mathsf{nce}}$ responds similarly as above. At some point, $\mathcal{A}$ outputs cert. $\mathcal{B}_{\mathsf{skcd}}$ passes cert to the challenger of $\mathsf{Exp}^{\mathsf{otsk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{skcd}},\mathcal{B}_{\mathsf{skcd}}}(\lambda, b')$. If the challenger returns $\mathsf{ske.sk}$, $\mathcal{B}_{\mathsf{skcd}}$ generates $\widetilde{\mathsf{msk}} \leftarrow \mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.msk}, \mathsf{nce.aux}, \mathsf{nce}.\widetilde{\mathsf{CT}}_X, \mathsf{ske.sk})$ and sends $\widetilde{\mathsf{msk}}$ to $\mathcal{A}$. Otherwise, $\mathcal{B}_{\mathsf{skcd}}$ sends $\bot$ to $\mathcal{A}$. Finally, $\mathcal{B}_{\mathsf{skcd}}$ outputs whatever $\mathcal{A}$ outputs.

- If $b' = 0$, i.e., $\mathsf{ske.CT}^* = \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_0)$, $\mathcal{B}_{\mathsf{skcd}}$ perfectly simulates $\mathsf{Hyb}(0)$.
- If $b' = 1$, i.e., $\mathsf{ske.CT}^* = \mathsf{SKE.Enc}(\mathsf{ske.sk}, m_1)$, $\mathcal{B}_{\mathsf{skcd}}$ perfectly simulates $\mathsf{Hyb}(1)$.

Thus, if $\mathcal{A}$ distinguishes the two hybrids, $\mathcal{B}_{\mathsf{skcd}}$ breaks the OT-CD security of Σ_{skcd}. This completes the proof. ☐

By Propositions 4.12 and 4.13, we immediately obtain Theorem 3.4. ☐

Summary of this section. Since IO and OWFs imply computational NIZK proof for **NP** [BP15] and IND-CPA secure ABE for circuits, we immediately obtain the following corollary by using Theorems 2.10, 2.15, 4.6 and 4.11.

Corollary 4.14. *If there exist secure IO for **P/poly** and OWFs against QPT adversaries, there exists ABE with certified deletion for circuits.*

5 PKE with Certified Deletion and Classical Communication

In this section, we define the notion of public key encryption with certified deletion with classical communication, and construct it from the LWE assumption in the QROM. In Section 5.1, we present the definition of the public key encryption with certified deletion with classical communication. In Section 5.2, we introduce what we call the *cut-and-choose adaptive hardcore property*, which is used in the security proof of the PKE with certified deletion with classical communication. In Section 5.3, we construct a PKE with certified deletion with classical communication, and show its security.

5.1 Definition of PKE with Certified Deletion and Classical Communication

We define PKE with certified deletion with classical communication. Note that the encryption algorithm of a PKE with certified deletion with classical communication is interactive unlike PKE with certified deletion (with quantum communication) as defined in Definition 3.1. It is easy to see that the interaction is necessary if we only allow classical communication.

Definition 5.1 (PKE with Certified Deletion with Classical Communication (Syntax)). *A public key encryption scheme with certified deletion with classical communication is a tuple of quantum algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *with plaintext space* $\mathcal{M}$.

$\mathsf{KeyGen}(1^\lambda) \to (\mathsf{pk}, \mathsf{sk})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a classical key pair* $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{Enc}\langle \mathcal{S}(\mathsf{pk}, m), \mathcal{R}\rangle \to (\mathsf{vk}, \mathsf{CT})$**:** *This is an interactive process between a classical sender* $\mathcal{S}$ *with input* pk *and a plaintext* $m \in \mathcal{M}$, *and a quantum receiver* $\mathcal{R}$ *without input. After exchanging classical messages,* $\mathcal{S}$ *outputs a classical verification key* vk *and* $\mathcal{R}$ *outputs a quantum ciphertext* CT.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \to m'$ **or** $\bot$**:** *The decryption algorithm takes as input the secret key* sk *and the ciphertext* CT, *and outputs a plaintext* m' *or* $\bot$.

$\mathsf{Del}(\mathsf{CT}) \to \mathsf{cert}$**:** *The deletion algorithm takes as input the ciphertext* CT *and outputs a classical certificate* cert.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) \to \top$ **or** $\bot$**:** *The verification algorithm takes the verification key* vk *and the certificate* CT, *and outputs* $\top$ *or* $\bot$.

Definition 5.2 (Correctness for PKE with Certified Deletion with Classical Communication). *There are two types of correctness. One is decryption correctness and the other is verification correctness.*

Decryption correctness: *For any* $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$,

$$\Pr\left[\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \neq m \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (\mathsf{vk}, \mathsf{CT}) \leftarrow \mathsf{Enc}\langle \mathcal{S}(\mathsf{pk}, m), \mathcal{R}\rangle \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Verification correctness: *For any* $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$,

$$\Pr\left[\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) = \bot \;\middle|\; \begin{array}{l} (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (\mathsf{vk}, \mathsf{CT}) \leftarrow \mathsf{Enc}\langle \mathcal{S}(\mathsf{pk}, m), \mathcal{R}\rangle \\ \mathsf{cert} \leftarrow \mathsf{Del}(\mathsf{CT}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition 5.3 (Certified Deletion Security for PKE with Classical Communication). *Let* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *be a PKE with certified deletion scheme with classical communication. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, b)$.

1. *The challenger computes* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and sends* pk *to* $\mathcal{A}$.
2. $\mathcal{A}$ *sends* $(m_0, m_1) \in \mathcal{M}^2$ *to the challenger.*

3. *The challenger and $\mathcal{A}$ jointly execute* $(\mathsf{vk}_b, \mathsf{CT}_b) \leftarrow \mathsf{Enc}\langle \mathcal{S}(\mathsf{pk}, m_b), \mathcal{A}(\mathsf{pk})\rangle$ *where the challenger plays the role of the sender and $\mathcal{A}$ plays the role of the receiver.*

4. *At some point, $\mathcal{A}$ sends* cert *to the challenger.*

5. *The challenger computes* $\mathsf{Vrfy}(\mathsf{vk}_b, \mathsf{cert})$. *If the output is $\bot$, the challenger sends $\bot$ to $\mathcal{A}$. If the output is $\top$, the challenger sends* sk *to $\mathcal{A}$.*

6. *$\mathcal{A}$ outputs its guess $b' \in \{0,1\}$.*

Let $\mathsf{Adv}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda)$ *be the advantage of the experiment above. We say that the Σ is IND-CPA-CD secure if for any QPT adversary $\mathcal{A}$, it holds that*

$$\mathsf{Adv}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \leq \mathsf{negl}(\lambda).$$

5.2 Preparation

We prove that any injective invariant NTCF family satisfies a property which we call the *cut-and-choose adaptive hardcore property*, which is used in the security proof of our PKE with certified deletion with classical communication. To prove the cut-and-choose adaptive hardcore property, we first prove a simple combinatorial lemma and its corollary.

Lemma 5.4. *Let n, k be a positive integers and $T \subseteq [2n]$ be a subset. Let $S \subseteq [2n]$ be a uniformly random subset conditioned on that $|S| = n$. If $k \leq |T|$, we have*

$$\Pr[S \cap T = \emptyset] \leq \left(\frac{1}{2}\right)^k.$$

If $|T| \leq k \leq n$, we have

$$\Pr[S \cap T = \emptyset] > \left(\frac{n-k}{2n-k}\right)^k.$$

Proof. When $|T| \geq n+1$, we have $\Pr[S \cap T = \emptyset] = 0$ by the pigeonhole principle and thus the first inequality trivially holds. Note that we need not consider such a case for the second inequality since we assume $|T| \leq n$ for the second inequality. In the following, we assume $|T| \leq n$. Let $k' := |T|$. Then we have

$$\Pr[S \cap T = \emptyset] = \frac{\binom{2n-k'}{n}}{\binom{2n}{n}} = \frac{(2n-k')!}{(n-k')!n!} \cdot \frac{n!n!}{(2n)!} = \prod_{i=0}^{k'-1} \frac{n-i}{2n-i}.$$

For all $0 \leq i \leq k'-1$, we have

$$\frac{n-k'}{2n-k'} < \frac{n-i}{2n-i} \leq \frac{1}{2}.$$

Therefore we have

$$\left(\frac{n-k'}{2n-k'}\right)^{k'} < \Pr[S \cap T = \emptyset] \leq \left(\frac{1}{2}\right)^{k'}.$$

Then, Lemma 5.4 follows from $\left(\frac{1}{2}\right)^{k'} \leq \left(\frac{1}{2}\right)^{k}$ for $k \leq k'$ and $\left(\frac{n-k}{2n-k}\right)^{k} \leq \left(\frac{n-k'}{2n-k'}\right)^{k'}$ for $k' \leq k \leq n$. ☐

Corollary 5.5. *Let n, k be a positive integers and $T \subseteq [4n]$ be a subset. Let $S \subseteq [4n]$ be a uniformly random subset conditioned on that $|S| = 2n$ and $U \subseteq S$ be a uniformly random subset conditioned on that $|U| = n$. If $k \leq |T|$, we have*

$$\Pr[S \cap T = \emptyset] \leq \left(\frac{1}{2}\right)^k.$$

For any $S^* \subseteq [4n]$ such that $|S^*| = 2n$, if $|T| < k < n$, we have

$$\Pr\left[U \cap T = \emptyset | S = S^*\right] > \left(\frac{n-k}{2n-k}\right)^k.$$

Proof. The former part immediately follows from Lemma 5.4 by considering $2n$ as n in Lemma 5.4. For the latter part, suppose that S is fixed to be S^*. We have $U \cap T = U \cap (S^* \cap T)$ and $|S^* \cap T| \leq |T| < k < n$. By considering a one-to-one map from S^* to $[2n]$, we can apply the latter statement of Lemma 5.4, where we think of the images of U and $S^* \cap T$ as S and T in Lemma 5.4, respectively, to obtain Corollary 5.5. □

Then we define the cut-and-choose adaptive hardcore property and prove that any injective invariant NTCF family satisfies it.

Lemma 5.6 (Cut-and-Choose Adaptive Hardcore Property). *Let $\mathcal{F}$ be an injective invariant NTCF family and $\mathcal{G}$ be the corresponding trapdoor injective family. Then $\mathcal{F}$ and $\mathcal{G}$ satisfy what we call the* cut-and-choose adaptive hardcore property *defined below. For a QPT adversary $\mathcal{A}$ and a positive integer n, we consider the following experiment $\mathsf{Exp}^{\text{cut-and-choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{A}}(\lambda, n)$.*

1. *The challenger chooses a uniform subset $S \subseteq [4n]$ such that $|S| = 2n$.*[10]
2. *The challenger generates $(\mathsf{k}_i, \mathsf{td}_i) \leftarrow \mathsf{Gen}_{\mathcal{G}}(1^\lambda)$ for all $i \in S$ and $(\mathsf{k}_i, \mathsf{td}_i) \leftarrow \mathsf{Gen}_{\mathcal{F}}(1^\lambda)$ for all $i \in \overline{S}$ and sends $\{\mathsf{k}_i\}_{i \in [4n]}$ to $\mathcal{A}$.*
3. *$\mathcal{A}$ sends $\{y_i, d_i, e_i\}_{i \in [4n]}$ to the challenger.*
4. *The challenger computes $x_{i,\beta} \leftarrow \mathsf{Inv}_{\mathcal{F}}(\mathsf{td}_i, \beta, y_i)$ for all $(i, \beta) \in \overline{S} \times \{0,1\}$ and checks if $d_i \in G_{\mathsf{k}_i,0,x_{i,0}} \cap G_{\mathsf{k}_i,1,x_{i,1}}$ and $e_i = d_i \cdot (J(x_{i,0}) \oplus J(x_{i,1}))$ hold for all $i \in \overline{S}$. If they do not hold for some $i \in \overline{S}$, the challenger immediately aborts and the experiment returns 0.*
5. *The challenger sends S to $\mathcal{A}$.*
6. *$\mathcal{A}$ sends $\{b_i, x_i\}_{i \in S}$ to the challenger.*
7. *The challenger checks if $\mathsf{Chk}_{\mathcal{G}}(\mathsf{k}_i, b_i, x_i, y_i) = 1$ holds for all $i \in S$. If this holds for all $i \in S$, the experiment returns 1. Otherwise, it returns 0.*

Then for any n such that $n \leq \mathrm{poly}(\lambda)$ and $n = \omega(\log \lambda)$, it holds that

$$\mathsf{Adv}^{\text{cut-and-choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{A}}(\lambda, n) := \Pr[\mathsf{Exp}^{\text{cut-and-choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{A}}(\lambda, n) = 1] \leq \mathsf{negl}(\lambda).$$

Proof. We consider a modified experiment $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n)$ that works similarly to $\mathsf{Exp}^{\text{cut-and-choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{A}}(\lambda, n)$ except that the challenger generates $(\mathsf{k}_i, \mathsf{td}_i)$ by $\mathsf{Gen}_{\mathcal{F}}$ for all $i \in [4n]$. Since the challenger in these experiments does not use td_i for $i \in S$ at all, the injective invariance implies

$$|\Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1] - \Pr[\mathsf{Exp}^{\text{cut-and-choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{A}}(\lambda, n) = 1]| \leq \mathsf{negl}(\lambda)$$

by a straightforward hybrid argument. Therefore, it suffices to prove

$$\Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1] \leq \mathsf{negl}(\lambda).$$

In the following, we reduce this to the amplified adaptive hardcore property (Lemma 2.29). For the sake of contradiction, we assume that $\Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1]$ is non-negligible. Then there exists a polynomial p such that

$$\Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1] \geq \frac{1}{p(\lambda)}$$

[10]We can also take $S \subseteq [2n]$ such that $|S| = n$, but we do as above just for convenience in the proof.

for infinitely many $\lambda \in \mathbb{N}$. Let k be the minimal integer such that

$$\left(\frac{1}{2}\right)^k \le \frac{1}{2p(\lambda)}.$$

In $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n)$, let $T \subseteq [4n]$ be the subset consisting of $i \in [4n]$ such that $d_i \notin G_{\mathsf{k}_i,0,x_{i,0}} \cap G_{\mathsf{k}_i,1,x_{i,1}}$ or $e_i \neq d_i \cdot (J(x_{i,0}) \oplus J(x_{i,1}))$ where $x_{i,\beta} \leftarrow \mathsf{Inv}_{\mathcal{F}}(\mathsf{td}_i, \beta, y_i)$ for $\beta \in \{0,1\}$.[11] Let $\mathtt{Bad}$ be the event such that $|T| \ge k$. When $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n)$ returns 1, we must have $\overline{S} \cap T = \emptyset$. Therefore, by Corollary 5.5, we have

$$\begin{aligned}
&\Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \mathtt{Bad}] \\
&\le \Pr[\overline{S} \cap T = \emptyset \wedge \mathtt{Bad}] \\
&\le \Pr[\overline{S} \cap T = \emptyset | \mathtt{Bad}] \\
&\le \left(\frac{1}{2}\right)^k \\
&\le \frac{1}{2p(\lambda)}.
\end{aligned}$$

We remark that we can apply Corollary 5.5 to get the third inequality above since $|T| \ge k$ when $\mathtt{Bad}$ occurs and no information of S is given to $\mathcal{A}$ before Step 5 in $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n)$ and thus $\overline{S}$ is independent of $\mathtt{Bad}$. On the other hand, we have

$$\Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1] = \Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \mathtt{Bad}] + \Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathtt{Bad}}] \ge \frac{1}{p(\lambda)}$$

for infinitely many λ. Therefore we have

$$\Pr[\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathtt{Bad}}] \ge \frac{1}{2p(\lambda)}$$

for infinitely many λ. This naturally gives an adversary $\mathcal{B}$ that breaks the amplified adaptive hardcore property that is described below.

$\mathcal{B}(\{\mathsf{k}_j^*\}_{j \in [n]})$**:** Given a problem instance $\{\mathsf{k}_j^*\}_{j \in [n]}$, it works as follows.

1. Choose a uniform subset $S \subseteq [4n]$ such that $|S| = 2n$.
2. Choose a uniform subset $U \subseteq S$ such that $|U| = n$. Let $i_1, ..., i_n$ be the elements of U.
3. Set $\mathsf{k}_{i_j} := \mathsf{k}_j^*$ for all $j \in [n]$ and generate $(\mathsf{k}_i, \mathsf{td}_i) \leftarrow \mathsf{Gen}_{\mathcal{F}}(1^\lambda)$ for all $i \in \overline{U}$.
4. Send $\{\mathsf{k}_i\}_{i \in [4n]}$ to $\mathcal{A}$ and receives the response $\{y_i, d_i, e_i\}_{i \in [4n]}$ from $\mathcal{A}$.
5. Send S to $\mathcal{A}$ and receives the response $\{b_i, x_i\}_{i \in S}$ from $\mathcal{A}$.
6. Output $\{b_i, x_i, y_i, d_i, e_i\}_{i \in U}$.

We can see that $\mathcal{B}$ perfectly simulates $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n)$ for $\mathcal{A}$, and $\mathcal{B}$ wins the amplified adaptive hardcore game whenever we have $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1$ and $U \cap T = \emptyset$ in the simulated experiment.

[11]Note that $x_{i,\beta}$ was defined only for $(i, \beta) \in \overline{S} \times \{0,1\}$ in $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n)$, but we naturally generalize it to all $(i, \beta) \in [4n] \times \{0,1\}$. We remark that this is well-defined since the challenger uses $\mathsf{Gen}_{\mathcal{F}}$ on all positions in $\widetilde{\mathsf{Exp}}^{\text{cut-and-choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n)$ unlike in $\mathsf{Exp}^{\text{cut-and-choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{A}}(\lambda, n)$.

We have $k = O(\log\lambda)$ by the definition of k and $n = \omega(\log\lambda)$ by the assumption. Therefore we have $k < \frac{n}{2}$ for sufficiently large λ. Then, for sufficiently large λ, we have

$$\begin{aligned}
&\Pr[U \cap T = \emptyset | \widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}] \\
&= \frac{\Pr[U \cap T = \emptyset \wedge \widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}]}{\Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}]} \\
&= \sum_{S^* \subseteq [4n] \text{ s.t. } |S^*| = 2n} \ \sum_{T^* \subseteq [4n] \text{ s.t. } |T^*| < k} \frac{\Pr[U \cap T^* = \emptyset \wedge \widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}} \wedge S = S^* \wedge T = T^*]}{\Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}]} \\
&= \sum_{S^* \subseteq [4n] \text{ s.t. } |S^*| = 2n} \ \sum_{T^* \subseteq [4n] \text{ s.t. } |T^*| < k} \frac{\Pr[U \cap T^* = \emptyset] \cdot \Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}} \wedge S = S^* \wedge T = T^*]}{\Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}]} \\
&> \sum_{S^* \subseteq [4n] \text{ s.t. } |S^*| = 2n} \ \sum_{T^* \subseteq [4n] \text{ s.t. } |T^*| < k} \left(\frac{n-k}{2n-k}\right)^k \cdot \frac{\Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}} \wedge S = S^* \wedge T = T^*]}{\Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}]} \\
&= \left(\frac{n-k}{2n-k}\right)^k \\
&\geq \frac{1}{\mathrm{poly}(\lambda)}
\end{aligned}$$

where the first equality follows from the definition of conditional probability, the second and fourth equalities follow from the fact that $|T| < k$ when $\overline{\mathsf{Bad}}$ occurs, the third equality follows from the fact that U is independent of the events $\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1$, $\overline{\mathsf{Bad}}$, and $T = T^*$ when we fix S, the first inequality follows from Corollary 5.5 and $|T| < k < n/2 < n$ for sufficiently large λ, and the second inequality follows from $k < \frac{n}{2}$ for sufficiently large λ, in which case we have $\frac{n-k}{2n-k} > \frac{1}{3}$, and $k = O(\log\lambda)$.

Therefore we have

$$\begin{aligned}
\Pr[\mathcal{B} \text{ wins}] &\geq \Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge U \cap T = \emptyset] \\
&\geq \Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge U \cap T = \emptyset \wedge \overline{\mathsf{Bad}}] \\
&= \Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}] \cdot \Pr[U \cap T = \emptyset | \widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1 \wedge \overline{\mathsf{Bad}}] \\
&\geq \frac{1}{\mathrm{poly}(\lambda)}
\end{aligned}$$

for infinitely many λ. This contradicts the amplified adaptive hardcore property (Lemma 2.29). Therefore, our assumption that $\Pr[\widetilde{\mathsf{Exp}}^{\mathsf{cut\text{-}and\text{-}choose}}_{\mathcal{F},\mathcal{A}}(\lambda, n) = 1]$ is non-negligible is false, which completes the proof of Lemma 5.6. $\square$

5.3 Construction

We construct a PKE scheme with certified deletion with classical communication $\Sigma_{\mathsf{cccd}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ with plaintext space $\mathcal{M} = \{0,1\}^\ell$ from an NTCF family $\mathcal{F}$ with the corresponding trapdoor injective family $\mathcal{G}$ for which we use notations given in Section 2.3, a public key NCE scheme $\Sigma_{\mathsf{nce}} = \mathsf{NCE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ with plaintext space $\{S \subseteq [4n] : |S| = 2n\}$ where n is a positive integer such that $n \leq \mathrm{poly}(\lambda)$ and $n = \omega(\log\lambda)$ and we just write S to mean the description of the set S by abuse of notation, a OW-CPA secure PKE scheme $\Sigma_{\mathsf{ow}} = \mathsf{OW}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with plaintext space $\{0,1\}^\lambda$, and a hash function H from $\{0,1\}^\lambda \times (\{0,1\} \times \mathcal{X})^{2n}$ to $\{0,1\}^\ell$ modeled as a quantumly-accessible random oracle.

$\mathsf{KeyGen}(1^\lambda)$**:**

- Generate $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^\lambda)$ and $(\mathsf{ow.pk}, \mathsf{ow.sk}) \leftarrow \mathsf{OW.KeyGen}(1^\lambda)$ and output $(\mathsf{pk}, \mathsf{sk}) := ((\mathsf{nce.pk}, \mathsf{ow.pk}), (\mathsf{nce.sk}, \mathsf{ow.sk}))$.

$\mathsf{Enc}\langle \mathcal{S}(\mathsf{pk}, m), \mathcal{R}\rangle$**:** This is an interactive protocol between a sender $\mathcal{S}$ with input (pk, m) and a receiver $\mathcal{R}$ without input that works as follows.

- $\mathcal{S}$ parses $\mathsf{pk} = (\mathsf{nce.pk}, \mathsf{ow.pk})$.
- $\mathcal{S}$ chooses a uniform subset $S \subseteq [4n]$ such that $|S| = 2n$, generates

$$(\mathsf{k}_i, \mathsf{td}_i) \leftarrow \begin{cases} \mathsf{Gen}_{\mathcal{G}}(1^\lambda) & i \in S \\ \mathsf{Gen}_{\mathcal{F}}(1^\lambda) & i \in \overline{S} \end{cases}$$

  for $i \in [4n]$, and sends $\{\mathsf{k}_i\}_{i \in [4n]}$ to $\mathcal{R}$.
- For $i \in [4n]$, $\mathcal{R}$ generates a quantum state

$$|\psi_i'\rangle = \begin{cases} \frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}, b \in \{0,1\}} \sqrt{(g_{\mathsf{k}_i,b}(x))(y)} |b, x\rangle |y\rangle & (i \in S) \\ \frac{1}{\sqrt{|\mathcal{X}|}} \sum_{x \in \mathcal{X}, y \in \mathcal{Y}, b \in \{0,1\}} \sqrt{(f'_{\mathsf{k}_i,b}(x))(y)} \, |b, x\rangle \, |y\rangle & (i \in \overline{S}) \end{cases}$$

  by using Samp, measure the last register to obtain $y_i \in \mathcal{Y}$, and let $|\phi_i'\rangle$ be the post-measurement state where the measured register is discarded. Note that this can be done without knowing S since $\mathsf{Samp}_{\mathcal{F}} = \mathsf{Samp}_{\mathcal{G}}$, which is just denoted by Samp, as required in Definition 2.32. By Definitions 2.28 and 2.31, we can see that for all $i \in [4n]$, $|\phi_i'\rangle$ has a negligible trace distance from the following state:

$$|\phi_i\rangle = \begin{cases} |b_i\rangle \, |x_i\rangle & (i \in S) \\ \frac{1}{\sqrt{2}} (|0\rangle \, |x_{i,0}\rangle + |1\rangle \, |x_{i,1}\rangle) & (i \in \overline{S}) \end{cases}$$

  where $(x_i, b_i) \leftarrow \mathsf{Inv}_{\mathcal{G}}(\mathsf{td}_i, y_i)$ for $i \in S$ and $x_{i,\beta} \leftarrow \mathsf{Inv}_{\mathcal{F}}(\mathsf{td}_i, \beta, y_i)$ for $(i, \beta) \in \overline{S} \times \{0,1\}$.[12] $\mathcal{R}$ sends $\{y_i\}_{i \in [4n]}$ to $\mathcal{S}$ and keeps $\{|\phi_i'\rangle\}_{i \in [4n]}$.
- $\mathcal{S}$ chooses $K \leftarrow \{0,1\}^\lambda$ and computes $(b_i, x_i) \leftarrow \mathsf{Inv}_{\mathcal{G}}(\mathsf{td}_i, y_i)$ for all $i \in S$. If $\mathsf{Chk}_{\mathcal{G}}(\mathsf{k}_i, b_i, x_i, y_i) = 0$ for some $i \in S$, $\mathcal{S}$ returns $\perp$ to $\mathcal{R}$. Otherwise, let $i_1, ..., i_{2n}$ be the elements of S in the ascending order. $\mathcal{S}$ sets $Z := (K, (b_{i_1}, x_{i_1}), (b_{i_2}, x_{i_2}), ..., (b_{i_{2n}}, x_{i_{2n}}))$, computes

$$\begin{aligned} &\mathsf{nce.CT} \leftarrow \mathsf{NCE.Enc}(\mathsf{nce.pk}, S), \\ &\mathsf{ow.CT} \leftarrow \mathsf{OW.Enc}(\mathsf{ow.pk}, K), \\ &\mathsf{CT}_{\mathsf{msg}} := m \oplus H(Z), \end{aligned}$$

  and sends $(\mathsf{nce.CT}, \mathsf{ow.CT}, \mathsf{CT}_{\mathsf{msg}})$ to $\mathcal{R}$.
- $\mathcal{S}$ outputs $\mathsf{vk} := \{\mathsf{td}_i, y_i\}_{i \in \overline{S}}$ and $\mathcal{R}$ outputs $\mathsf{CT} := (\{|\phi_i'\rangle\}_{i \in [4n]}, \mathsf{nce.CT}, \mathsf{ow.CT}, \mathsf{CT}_{\mathsf{msg}})$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$**:**

- Parse $\mathsf{sk} = (\mathsf{nce.sk}, \mathsf{ow.sk})$ and $\mathsf{CT} = (\{|\phi_i'\rangle\}_{i \in [4n]}, \mathsf{nce.CT}, \mathsf{ow.CT}, \mathsf{CT}_{\mathsf{msg}})$.
- Compute $S' \leftarrow \mathsf{NCE.Dec}(\mathsf{nce.sk}, \mathsf{nce.CT})$.
- Compute $K' \leftarrow \mathsf{OW.Dec}(\mathsf{ow.sk}, \mathsf{ow.CT})$.
- For all $i \in S'$, measure $|\phi_i'\rangle$ in the computational basis and let (b_i', x_i') be the outcome.
- Compute and output $m' := \mathsf{CT}_{\mathsf{msg}} \oplus H(K', (b'_{i_1}, x'_{i_1}), (b'_{i_2}, x'_{i_2}), ..., (b'_{i_{2n}}, x'_{i_{2n}}))$ where $i_1, ..., i_{2n}$ are the elements of S' in the ascending order.[13]

[12]Indeed, $|\psi_i'\rangle = |\psi_i\rangle$ for $i \in S$.
[13]If $S' = \perp$ or $K' = \perp$, output $\perp$.

$\mathsf{Del}(\mathsf{CT})$:

- Parse $\mathsf{CT} = (\{|\phi'_i\rangle\}_{i\in[4n]}, \mathsf{nce.CT}, \mathsf{ow.CT}, \mathsf{CT}_{\mathsf{msg}})$.
- For all $i \in [4n]$, evaluate the function J on the second register of $|\phi'_i\rangle$. That is, apply an isometry that maps $|b,x\rangle$ to $|b,J(x)\rangle$ to $|\phi'_i\rangle$. (Note that this can be done efficiently since J is injective and efficiently invertible.) Let $|\phi''_i\rangle$ be the resulting state.
- For all $i \in [4n]$, measure $|\phi''_i\rangle$ in the Hadamard basis and let (e_i, d_i) be the outcome.
- Output $\mathsf{cert} := \{(e_i, d_i)\}_{i\in[4n]}$.

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

- Parse $\mathsf{vk} = \{\mathsf{td}_i, y_i\}_{i\in\overline{S}}$ and $\mathsf{cert} = \{(e_i, d_i)\}_{i\in[4n]}$.
- Compute $x_{i,\beta} \leftarrow \mathsf{Inv}_{\mathcal{F}}(\mathsf{td}_i, \beta, y_i)$ for all $(i,\beta) \in \overline{S} \times \{0,1\}$.
- Output $\top$ if $d_i \in G_{\mathsf{k}_i,0,x_{i,0}} \cap G_{\mathsf{k}_i,1,x_{i,1}}$ and $e_i = d_i \cdot (J(x_{i,0}) \oplus J(x_{i,1}))$ hold for all $i \in \overline{S}$ and output $\bot$ otherwise.

Correctness. As observed in the description, $|\phi'_i\rangle$ in the ciphertext has a negligible trace distance from $|\phi_i\rangle$. Therefore, it suffices to prove correctness assuming that $|\phi'_i\rangle$ is replaced with $|\phi_i\rangle$. After this replacement, decryption correctness clearly holds assuming correctness of Σ_{nce} and Σ_{ow}.

We prove verification correctness below. For $i \in \overline{S}$, if we apply J to the second register of $|\phi_i\rangle$ and then apply Hadamard transform for both registers as in Del, then the resulting state can be written as

$$\begin{aligned}&2^{-\frac{w+2}{2}} \sum_{d,b,e} (-1)^{d\cdot J(x_{i,b})\oplus eb} |e\rangle |d\rangle \\ &= 2^{-\frac{w}{2}} \sum_{d\in\{0,1\}^w} (-1)^{d\cdot J(x_{i,0})} |d\cdot(J(x_{i,0})\oplus J(x_{i,1}))\rangle |d\rangle .\end{aligned}$$

Therefore, the measurement result is (e_i, d_i) such that $e_i = d_i \cdot (J(x_{i,0}) \oplus J(x_{i,1}))$ for a uniform $d_i \leftarrow \{0,1\}^w$. By the first item of the adaptive hardcore property in Definition 2.28, it holds that $d_i \in G_{\mathsf{k}_i,0,x_{i,0}} \cap G_{\mathsf{k}_i,1,x_{i,1}}$ except for a negligible probability. Therefore, the certificate $\mathsf{cert} = \{(e_i, d_i)\}_{i\in[4n]}$ passes the verification by Vrfy with overwhelming probability.

Security. We prove the following theorem.

Theorem 5.7. *If Σ_{nce} is RNC secure, Σ_{ow} is OW-CPA secure, and $\mathcal{F}$ is an injective invariant NTCF family with the corresponding injective trapdoor family $\mathcal{G}$, Σ_{cccd} is IND-CPA-CD secure in the QROM where H is modeled as a quantumly-accessible random oracle.*

Proof. What we need to prove is that for any QPT adversary $\mathcal{A}$, it holds that

$$\mathsf{Adv}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda,0)=1] - \Pr[\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda,1)=1]| \le \mathsf{negl}(\lambda).$$

Let $q = \mathrm{poly}(\lambda)$ be the maximum number of $\mathcal{A}$'s random oracle queries. For clarity, we describe how $\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda, b)$ works below.

1. A uniformly random function H from $\{0,1\}^\lambda \times (\{0,1\} \times \mathcal{X})^{2n}$ to $\{0,1\}^\ell$ is chosen, and $\mathcal{A}$ can make arbitrarily many quantum queries to H at any time in the experiment.

2. The challenger generates $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^\lambda)$ and $(\mathsf{ow.pk}, \mathsf{ow.sk}) \leftarrow \mathsf{OW.KeyGen}(1^\lambda)$ and sends $\mathsf{pk} := (\mathsf{nce.pk}, \mathsf{ow.pk})$ to $\mathcal{A}$.

3. $\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M}^2$ to the challenger.

4. The challenger chooses a uniform subset $S \subseteq [4n]$ such that $|S| = 2n$, generates

$$(\mathsf{k}_i, \mathsf{td}_i) \leftarrow \begin{cases} \mathsf{Gen}_{\mathcal{G}}(1^\lambda) & i \in S \\ \mathsf{Gen}_{\mathcal{F}}(1^\lambda) & i \in \overline{S} \end{cases}$$

for $i \in [4n]$, and sends $\{\mathsf{k}_i\}_{i\in[4n]}$ to $\mathcal{A}$.

5. $\mathcal{A}$ sends $\{y_i\}_{i\in[4n]}$ to the challenger.

6. The challenger chooses $K \leftarrow \{0,1\}^\lambda$ and computes $(b_i, x_i) \leftarrow \mathsf{Inv}_{\mathcal{G}}(\mathsf{td}_i, y_i)$ for all $i \in S$. If $\mathsf{Chk}_{\mathcal{G}}(\mathsf{k}_i, b_i, x_i, y_i) = 0$ for some $i \in S$, the challenger sets $Z := \mathsf{null}$ and returns $\perp$ to $\mathcal{A}$ where null is a special symbol indicating that Z is undefined. Otherwise, let $i_1, ..., i_{2n}$ be the elements of S in the ascending order. The challenger sets $Z := (K, (b_{i_1}, x_{i_1}), (b_{i_2}, x_{i_2}), ..., (b_{i_{2n}}, x_{i_{2n}}))$, computes

$$\mathsf{nce.CT} \leftarrow \mathsf{NCE.Enc}(\mathsf{nce.pk}, S),$$
$$\mathsf{ow.CT} \leftarrow \mathsf{OW.Enc}(\mathsf{ow.pk}, K),$$
$$\mathsf{CT}_{\mathsf{msg}} := m_b \oplus H(Z),$$

and sends $(\mathsf{nce.CT}, \mathsf{ow.CT}, \mathsf{CT}_{\mathsf{msg}})$ to $\mathcal{A}$.

7. $\mathcal{A}$ sends $\mathsf{cert} = \{(e_i, d_i)\}_{i\in[4n]}$ to the challenger.

8. The challenger computes $x_{i,\beta} \leftarrow \mathsf{Inv}_{\mathcal{F}}(\mathsf{td}_i, \beta, y_i)$ for all $(i,\beta) \in \overline{S} \times \{0,1\}$. If $d_i \in G_{\mathsf{k}_i,0,x_{i,0}} \cap G_{\mathsf{k}_i,1,x_{i,1}}$ and $e_i = d_i \cdot (J(x_{i,0}) \oplus J(x_{i,1}))$ hold for all $i \in \overline{S}$, sends $\mathsf{sk} := (\mathsf{nce.sk}, \mathsf{ow.sk})$ to $\mathcal{A}$, and otherwise sends $\perp$ to $\mathcal{A}$.

9. $\mathcal{A}$ outputs b'. The output of the experiment is b'.

We define the following sequence of hybrids.

$\mathsf{Hyb}_1(b)$**:** Let $\mathtt{Reveal}_{\mathsf{sk}}$ be the event that the challenger sends sk in Step 8. $\mathsf{Hyb}_1(b)$ is identical to $\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda, b)$ except that K is chosen at the beginning and the oracle given to $\mathcal{A}$ before $\mathtt{Reveal}_{\mathsf{sk}}$ occurs is replaced with $H_{K\|*\to H'}$, which is H reprogrammed according to H' on inputs whose first entry is K where H' is another independent random function. More formally, $H_{K\|*\to H'}$ is defined by

$$H_{K\|*\to H'}(K', (b_1, x_1), ..., (b_{2n}, x_{2n})) := \begin{cases} H(K', (b_1, x_1), ..., (b_{2n}, x_{2n})) & (K' \neq K) \\ H'(K', (b_1, x_1), ..., (b_{2n}, x_{2n})) & (K' = K) \end{cases}.$$

We note that the challenger still uses H to generate $\mathsf{CT}_{\mathsf{msg}}$ and the oracle after $\mathtt{Reveal}_{\mathsf{sk}}$ occurs is still H similarly to the real experiment. On the other hand, if $\mathtt{Reveal}_{\mathsf{sk}}$ does not occur, the oracle $H_{K\|*\to H'}$ is used throughout the experiment except for the generation of $\mathsf{CT}_{\mathsf{msg}}$.

$\mathsf{Hyb}_2(b)$**:** This is identical to $\mathsf{Hyb}_1(b)$ except that $\mathsf{nce.CT}$ and $\mathsf{nce.sk}$ that may be sent to $\mathcal{A}$ in Step 6 and 8 are replaced by

$$\mathsf{nce}.\widetilde{\mathsf{CT}} \leftarrow \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}),$$
$$\mathsf{nce}.\widetilde{\mathsf{sk}} \leftarrow \mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce}.\widetilde{\mathsf{CT}}, S).$$

$\mathsf{Hyb}_3(b)$**:** This is identical to $\mathsf{Hyb}_2(b)$ except that the oracle given to $\mathcal{A}$ after $\mathtt{Reveal}_{\mathsf{sk}}$ occurs is replaced with $H_{Z\to r}$, which is H reprogrammed to output r on input $Z = (K, (b_{i_1}, x_{i_1}), ..., (b_{i_{2n}}, x_{i_{2n}}))$ where r is an independently random ℓ-bit string. More formally, $H_{Z\to r}$ is defined by

$$H_{Z\to r}(Z') := \begin{cases} H(Z') & (Z' \neq Z) \\ r & (Z' = Z) \end{cases}.$$

Note that we have $H_{Z\to r} = H$ if $Z = \mathsf{null}$, i.e., if $\mathsf{Chk}_{\mathcal{G}}(\mathsf{k}_i, b_i, x_i, y_i) = 0$ for some $i \in S$ in Step 6.

Proposition 5.8. *If Σ_{ow} is OW-CPA secure,* $|\Pr[\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda,b)=1]-\Pr[\mathsf{Hyb}_1(b)=1]|\leq \mathsf{negl}(\lambda)$.

Proof. To show this, we assume that $|\Pr[\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda,b)=1]-\Pr[\mathsf{Hyb}_1(b)=1]|$ is non-negligible, and construct an adversary $\mathcal{B}_{\mathsf{ow}}$ that breaks the OW-CPA security of Σ_{ow}. For notational simplicity, we denote $\mathsf{Exp}^{\mathsf{ccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{cccd}},\mathcal{A}}(\lambda,b)$ by $\mathsf{Hyb}_0(b)$. We consider an algorithm $\widetilde{\mathcal{A}}$ that works as follows. $\widetilde{\mathcal{A}}$ is given an oracle $\mathcal{O}$, which is either H or $H_{K\|*\to H'}$, and an input z that consists of K and the whole truth table of H, where $K\leftarrow\{0,1\}^{\lambda}$, H and H' are uniformly random functions. $\widetilde{\mathcal{A}}$ runs $\mathsf{Hyb}_0(b)$ except that it uses its own oracle $\mathcal{O}$ to simulate $\mathcal{A}$'s random oracle queries before $\mathtt{Reveal}_{\mathsf{sk}}$ occurs. On the other hand, $\widetilde{\mathcal{A}}$ uses H to simulate $\mathsf{CT}_{\mathsf{msg}}$ and $\mathcal{A}$'s random oracle queries after $\mathtt{Reveal}_{\mathsf{sk}}$ occurs regardless of $\mathcal{O}$, which is possible because the truth table of H is included in the input z. By definition, we clearly have

$$\Pr[\mathsf{Hyb}_0(b)=1]=\Pr[\widetilde{\mathcal{A}}^{H}(K,H)=1]$$

and

$$\Pr[\mathsf{Hyb}_1(b)=1]=\Pr[\widetilde{\mathcal{A}}^{H_{K\|*\to H'}}(K,H)=1]$$

where H in the input means the truth table of H. We apply the one-way to hiding lemma (Lemma 2.34) to the above $\widetilde{\mathcal{A}}$. Note that $\widetilde{\mathcal{A}}$ is inefficient, but the one-way to hiding lemma is applicable to inefficient algorithms. Then if we let $\widetilde{\mathcal{B}}$ be the algorithm that measures uniformly chosen query of $\widetilde{\mathcal{A}}$ and $S_K\subseteq\{0,1\}^{\lambda}\times(\{0,1\}\times\mathcal{X})^{2n}$ be the subset of all elements whose first entry is K, we have

$$|\Pr[\widetilde{\mathcal{A}}^{H}(K,H)=1]-\Pr[\widetilde{\mathcal{A}}^{H_{K\|*\to H'}}(K,H)=1]|\leq 2q\sqrt{\Pr[\widetilde{\mathcal{B}}^{H_{K\|*\to H'}}(K,H)\in S_K]}.$$

By the assumption, the LHS is non-negligible, and thus $\Pr[\widetilde{\mathcal{B}}^{H_{K\|*\to H'}}(K,H)\in S_K]$ is non-negligible. We remark that $\widetilde{\mathcal{B}}$ uses the truth table of H only for generating $\mathsf{CT}_{\mathsf{msg}}:=m_b\oplus H(Z)$ since it halts before $\mathtt{Reveal}_{\mathsf{sk}}$ occurs, and thus it needs not to simulate the random oracle for $\mathcal{A}$ after $\mathtt{Reveal}_{\mathsf{sk}}$ occurs. Here, we observe that the oracle $H_{K\|*\to H'}$ reveals no information about $H(Z)$ since the first entry of Z is K, and thus $\mathsf{CT}_{\mathsf{msg}}$ is independently and uniformly random. Therefore, if we let $\widetilde{\mathcal{B}}'$ be the same as $\widetilde{\mathcal{B}}$ except that it does not take the truth table of H as input, and sets $\mathsf{CT}_{\mathsf{msg}}$ to be a uniformly random string instead of setting $\mathsf{CT}_{\mathsf{msg}}:=m_b\oplus H(Z)$, we have

$$\Pr[\widetilde{\mathcal{B}}^{H_{K\|*\to H'}}(K,H)\in S_K]=\Pr[\widetilde{\mathcal{B}'}^{H_{K\|*\to H'}}(K)\in S_K].$$

Moreover, for any fixed K, when H and H' are uniformly random, $H_{K\|*\to H'}$ is also a uniformly random function, and thus we have

$$\Pr[\widetilde{\mathcal{B}'}^{H_{K\|*\to H'}}(K)\in S_K]=\Pr[\widetilde{\mathcal{B}'}^{H}(K)\in S_K].$$

Since $\Pr[\widetilde{\mathcal{B}}^{H_{K\|*\to H'}}(K,H)\in S_K]$ is non-negligible, $\Pr[\widetilde{\mathcal{B}'}^{H}(K)\in S_K]$ is also non-negligible. Recall that $\widetilde{\mathcal{B}'}^{H}$ is an algorithm that simulates $\mathsf{Hyb}_0(b)$ with the modification that $\mathsf{CT}_{\mathsf{msg}}$ is set to be uniformly random and measures randomly chosen $\mathcal{A}$'s query before $\mathtt{Reveal}_{\mathsf{sk}}$ occurs. Then it is straightforward to construct an adversary $\mathcal{B}_{\mathsf{ow}}$ that breaks the OW-CPA security of Σ_{ow} by using $\widetilde{\mathcal{B}'}$. For clarity, we give the description of $\mathcal{B}_{\mathsf{ow}}$ below.

$\mathcal{B}_{\mathsf{ow}}$ is given $(\mathsf{ow.pk},\mathsf{ow.CT})$ from the challenger of $\mathsf{Exp}^{\mathsf{ow\text{-}cpa}}_{\Sigma,\mathcal{B}_{\mathsf{ow}}}(\lambda)$. $\mathcal{B}_{\mathsf{ow}}$ chooses $i\leftarrow[q]$ and runs $\mathsf{Hyb}_1(b)$ until $\mathcal{A}$ makes i-th random oracle query or $\mathtt{Reveal}_{\mathsf{sk}}$ occurs where $\mathcal{B}_{\mathsf{ow}}$ embeds the problem instance $(\mathsf{ow.pk},\mathsf{ow.CT})$ into those sent to $\mathcal{A}$ instead of generating them by itself. If $\mathtt{Reveal}_{\mathsf{sk}}$ occurs before $\mathcal{A}$ makes i-th random oracle query, $\mathcal{B}_{\mathsf{ow}}$ just aborts. Otherwise, $\mathcal{B}_{\mathsf{ow}}$ measures the i-th random oracle query by $\mathcal{A}$, and lets Z' be the measurement outcome. $\mathcal{B}_{\mathsf{ow}}$ outputs the first entry of Z'.

We note that $\mathcal{B}_{\mathsf{ow}}$ can efficiently simulate the random oracle H by Zhandry's compressed oracle technique [Zha19]. We also note that $\mathcal{B}_{\mathsf{ow}}$ needs not to know $\mathsf{ow.sk}$ since it aborts as soon as $\mathtt{Reveal}_{\mathsf{sk}}$ occurs. We can see that the probability that $Z'\in S_K$ where K is the underlying message behind $\mathsf{ow.CT}$ is exactly $\Pr[\widetilde{\mathcal{B}'}^{H}(K)\in S_K]$, which is non-negligible. When $Z'\in S_K$, $\mathcal{B}_{\mathsf{ow}}$ outputs K. Therefore, $\mathcal{B}_{\mathsf{ow}}$ succeeds in predicting K with non-negligible probability. This contradicts the OW-CPA security of Σ_{ow}. Therefore $|\Pr[\mathsf{Hyb}_0(b)]-\Pr[\mathsf{Hyb}_1(b)]|$ is negligible. □

Proposition 5.9. *If Σ_{nce} is RNC secure,* $|\Pr[\mathsf{Hyb}_1(b)=1]-\Pr[\mathsf{Hyb}_2(b)=1]| \leq \mathsf{negl}(\lambda)$.

Proof. The proof is very similar to Proposition 3.5, but for the convenience of readers, we give the proof here. To show the proposition, we assume that $|\Pr[\mathsf{Hyb}_1(b)=1]-\Pr[\mathsf{Hyb}_2(b)=1]|$ is non-negligible, and construct an adversary $\mathcal{B}_{\mathsf{nce}}$ that breaks the RNC security of Σ_{nce}. Let $\mathcal{A}$ be the distinguisher for $\mathsf{Hyb}_1(b)$ and $\mathsf{Hyb}_2(b)$. First, $\mathcal{B}_{\mathsf{nce}}$ receives $\mathsf{nce.pk}$ from the challenger of $\mathsf{Exp}^{\mathsf{rec\text{-}nc}}_{\Sigma,\mathcal{B}_{\mathsf{nce}}}(\lambda,b')$ where b' is 0 or 1. $\mathcal{B}_{\mathsf{nce}}$ generates $K \leftarrow \{0,1\}^{\lambda}$ and simulates the random oracle for $\mathcal{A}$ as in $\mathsf{Hyb}_1(b)$ and $\mathsf{Hyb}_2(b)$. That is, it uses $H_{K\|*\to H'}$ before $\mathtt{Reveal}_{\mathsf{sk}}$ occurs and H after $\mathtt{Reveal}_{\mathsf{sk}}$ occurs, where H and H' are random functions. Note that quantum access to random functions H and H' can be efficiently simulated by using Zhandry's compressed oracle technique [Zha19]. $\mathcal{B}_{\mathsf{nce}}$ generates $(\mathsf{ow.pk},\mathsf{ow.sk}) \leftarrow \mathsf{OW.KeyGen}(1^{\lambda})$ and sends $(\mathsf{nce.pk},\mathsf{ow.pk})$ to $\mathcal{A}$ and receives (m_0,m_1) from $\mathcal{A}$. $\mathcal{B}_{\mathsf{nce}}$ generates S and $\{\mathsf{k}_i,\mathsf{td}_i\}_{i\in[4n]}$ as in $\mathsf{Hyb}_1(b)$ and $\mathsf{Hyb}_2(b)$, sends $\{\mathsf{k}_i\}_{i\in[4n]}$ to $\mathcal{A}$, and receives $\{y_i\}_{i\in[4n]}$ from $\mathcal{A}$. $\mathcal{B}_{\mathsf{nce}}$ computes $\mathsf{ow.CT}$ and $\mathsf{CT}_{\mathsf{msg}}$ as in $\mathsf{Hyb}_1(b)$ and $\mathsf{Hyb}_2(b)$, sends S to the challenger of $\mathsf{Exp}^{\mathsf{rec\text{-}nc}}_{\Sigma,\mathcal{B}_{\mathsf{nce}}}(\lambda,b')$, and receives $(\mathsf{nce.CT}^*,\mathsf{nce.sk}^*)$ from the challenger. Here,

$$(\mathsf{nce.CT}^*,\mathsf{nce.sk}^*) = (\mathsf{NCE.Enc}(\mathsf{nce.pk},S),\mathsf{nce.sk})$$

if $b'=0$, and

$$(\mathsf{nce.CT}^*,\mathsf{nce.sk}^*) = (\mathsf{NCE.Fake}(\mathsf{nce.pk},\mathsf{nce.sk},\mathsf{nce.aux}),\mathsf{NCE.Reveal}(\mathsf{nce.pk},\mathsf{nce.sk},\mathsf{nce.aux},\mathsf{nce.CT}^*,S))$$

if $b'=1$. $\mathcal{B}_{\mathsf{nce}}$ sends $(\mathsf{nce.CT}^*,\mathsf{ow.CT},\mathsf{CT}_{\mathsf{msg}})$ to $\mathcal{A}$ and receives cert from $\mathcal{A}$. If it is valid one (i.e., $\mathsf{Vrfy}(\mathsf{vk},\mathsf{cert})=\top$), $\mathcal{B}_{\mathsf{nce}}$ sends $(\mathsf{nce.sk}^*,\mathsf{ow.sk})$ to $\mathcal{A}$. Otherwise, it sends $\bot$ to $\mathcal{A}$. It is easy to see that $\mathcal{B}_{\mathsf{nce}}$ perfectly simulates $\mathsf{Hyb}_1(b)$ if $b'=0$ and $\mathsf{Hyb}_2(b)$ otherwise. By assumption, $\mathcal{A}$ can distinguish $\mathsf{Hyb}_1(b)$ and $\mathsf{Hyb}_2(b)$, and therefore $\mathcal{B}_{\mathsf{nce}}$ can break the RNC security of Σ_{nce}.

□

Proposition 5.10. *If $\mathcal{F}$ and $\mathcal{G}$ satisfy the cut-and-choose adaptive hardcore property (Lemma 5.6),* $|\Pr[\mathsf{Hyb}_2(b)=1]-\Pr[\mathsf{Hyb}_3(b)=1]| \leq \mathsf{negl}(\lambda)$.

Proof. To show this, we assume that $|\Pr[\mathsf{Hyb}_2(b)=1]-\Pr[\mathsf{Hyb}_3(b)=1]|$ is non-negligible, and show that the cut-and-choose adaptive hardcore property (Lemma 5.6) for $\mathcal{F}$ and $\mathcal{G}$ is broken. For simplicity, we first prove this assuming that $|\Pr[\mathsf{Hyb}_2(b)=1]-\Pr[\mathsf{Hyb}_3(b)=1]|$ is *noticeable*, i.e., there exists a polynomial p such that $|\Pr[\mathsf{Hyb}_2(b)=1]-\Pr[\mathsf{Hyb}_3(b)=1]| \geq 1/p(\lambda)$, and then we explain how to extend it to the non-negligible case.[14] First, we remark that $\mathsf{Hyb}_2(b)$ and $\mathsf{Hyb}_3(b)$ are identical unless $\mathtt{Reveal}_{\mathsf{sk}}$ occurs. Therefore, we have

$$\begin{aligned}|\Pr[\mathsf{Hyb}_2(b)=1]-\Pr[\mathsf{Hyb}_3(b)=1]| &= |\Pr[\mathsf{Hyb}_2(b)=1\wedge \mathtt{Reveal}_{\mathsf{sk}}]-\Pr[\mathsf{Hyb}_3(b)=1\wedge \mathtt{Reveal}_{\mathsf{sk}}]|\\ &= \Pr[\mathtt{Reveal}_{\mathsf{sk}}]\cdot|\Pr[\mathsf{Hyb}_2(b)=1\mid \mathtt{Reveal}_{\mathsf{sk}}]-\Pr[\mathsf{Hyb}_3(b)=1\mid \mathtt{Reveal}_{\mathsf{sk}}]|\end{aligned}$$

where we remark that $\Pr[\mathtt{Reveal}_{\mathsf{sk}}]$ is identical in $\mathsf{Hyb}_2(b)$ and $\mathsf{Hyb}_3(b)$ and thus we need not specify which hybrid is considered when we write $\Pr[\mathtt{Reveal}_{\mathsf{sk}}]$. Therefore, both $\Pr[\mathtt{Reveal}_{\mathsf{sk}}]$ and $|\Pr[\mathsf{Hyb}_2(b)=1|\mathtt{Reveal}_{\mathsf{sk}}]-\Pr[\mathsf{Hyb}_3(b)=1|\mathtt{Reveal}_{\mathsf{sk}}]|$ are noticeable.

We consider an algorithm $\widetilde{\mathcal{A}}$ that works as follows. $\widetilde{\mathcal{A}}$ is given an oracle $\mathcal{O}$, which is either H or $H_{Z\to r}$ and an input $z := (\{\mathsf{k}_i,\mathsf{td}_i\}_{i\in[4n]},\{y_i,e_i,d_i\}_{i\in[4n]},\rho_{\mathcal{A}})$ that are sampled according to the distribution $\mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$ defined below. Let $\mathcal{D}$ be the distribution of $(H,H_{Z\to r},(\{\mathsf{k}_i,\mathsf{td}_i\}_{i\in[4n]},\{y_i,e_i,d_i\}_{i\in[4n]},\rho_{\mathcal{A}}))$ sampled as in $\mathsf{Hyb}_3(b)$ where $\rho_{\mathcal{A}}$ is $\mathcal{A}$'s internal state just after sending cert in Step 7. We assume that $\mathcal{A}$ does nothing after sending cert until it receives sk or $\bot$ in Step 8 without loss of generality. Therefore, $\rho_{\mathcal{A}}$ is also the state just before receiving sk or $\bot$ in Step 8. $\mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$ is defined to be the conditional distribution of $\mathcal{D}$ conditioned on that $\mathtt{Reveal}_{\mathsf{sk}}$ occurs. Given an oracle $\mathcal{O}$, which is either H or $H_{Z\to r}$ and an input $z := (\{\mathsf{k}_i,\mathsf{td}_i\}_{i\in[4n]},\{y_i,e_i,d_i\}_{i\in[4n]},\rho_{\mathcal{A}})$ sampled from $\mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$, $\widetilde{\mathcal{A}}$ runs $\mathsf{Hyb}_2(b)$ starting from Step 8 where the internal state of $\mathcal{A}$ is initialized to be $\rho_{\mathcal{A}}$ and $\widetilde{\mathcal{A}}$ uses its own oracle $\mathcal{O}$ to simulate $\mathcal{A}$'s random oracle queries. By definition, we clearly have

$$\Pr[\mathsf{Hyb}_2(b)=1\mid \mathtt{Reveal}_{\mathsf{sk}}] = \Pr[\widetilde{\mathcal{A}}^{H}(z)=1]$$

[14]Looking ahead, we first consider the noticeable case because the fact that a product of two noticeable functions is also noticeable simplifies the proof. Note that a similar statement for non-negligible functions does not hold in general.

and

$$\Pr[\mathsf{Hyb}_3(b) = 1 \mid \mathtt{Reveal}_{\mathsf{sk}}] = \Pr[\widetilde{\mathcal{A}}^{H_{Z \to r}}(z) = 1]$$

where $(H, H_{Z \to r}, z) \leftarrow \mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$. We apply the one-way to hiding lemma (Lemma 2.34) to the above $\widetilde{\mathcal{A}}$. Let $\widetilde{\mathcal{B}}$ be the algorithm that measures uniformly chosen query of $\widetilde{\mathcal{A}}$. Then we have

$$|\Pr[\widetilde{\mathcal{A}}^{H}(z) = 1] - \Pr[\widetilde{\mathcal{A}}^{H_{Z \to r}}(z) = 1]| \leq 2q\sqrt{\Pr[\widetilde{\mathcal{B}}^{H_{Z \to r}}(z) = Z]}$$

where $(H, H_{Z \to r}, z) \leftarrow \mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$ and when $Z = \mathsf{null}$, we regard that $\widetilde{\mathcal{B}}^{H_{Z \to r}}(z) = Z$ does not occur regardless of the output of $\widetilde{\mathcal{B}}$. The LHS is equal to $|\Pr[\mathsf{Hyb}_2(b) = 1|\mathtt{Reveal}_{\mathsf{sk}}] - \Pr[\mathsf{Hyb}_3(b) = 1|\mathtt{Reveal}_{\mathsf{sk}}]|$, which is noticeable. Therefore $\Pr_{(H, H_{Z \to r}, z) \leftarrow \mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^{H_{Z \to r}}(z) = Z]$ is noticeable. Here, we observe that $H(Z)$ is used only for generating $\mathsf{CT}_{\mathsf{msg}} := m_b \oplus H(Z)$ in the execution of $(H, H_{Z \to r}, z) \leftarrow \mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$ and $\widetilde{\mathcal{B}}^{H_{Z \to r}}(z)$. Therefore, the output distribution of $\widetilde{\mathcal{B}}$ does not change even if we set $\mathsf{CT}_{\mathsf{msg}}$ to be a uniformly random string in the sampling procedure of $\mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$. Moreover, for any fixed Z, the joint distribution of $(H_{K\|* \to H'}, H_{Z \to r})$ is identical to the joint distribution of $(H_{K\|* \to H'}, H)$ when H, H', and r are uniformly random. Then, if we denote by $\mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}$ the distribution that is the same as $\mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}$ except that $\mathsf{CT}_{\mathsf{msg}}$ is set to be a uniformly random string in the sampling procedure and the second output $H_{Z \to r}$ is omitted, we have

$$\Pr_{(H, H_{Z \to r}, z) \leftarrow \mathcal{D}_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^{H_{Z \to r}}(z) = Z] = \Pr_{(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^{H}(z) = Z].$$

Since the LHS is noticeable, the RHS is noticeable. We note that the sequential execution of $(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}$ and $\widetilde{\mathcal{B}}^{H}(z)$ is the same as executing $\mathsf{Hyb}_2(b)$ with the modification that $\mathsf{CT}_{\mathsf{msg}}$ is set to be uniformly random and measuring a uniformly chosen $\mathcal{A}$'s query after Step 8 conditioned on that $\mathtt{Reveal}_{\mathsf{sk}}$ occurs. This naturally gives an adversary $\mathcal{B}_{\mathsf{cac}}$ that breaks the cut-and-choose adaptive hardcore property (Lemma 5.6) for $\mathcal{F}$ and $\mathcal{G}$, which embeds the problem instance of $\mathsf{Exp}^{\mathsf{cut\text{-}and\text{-}choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{B}_{\mathsf{cac}}}(\lambda, n)$ into the execution of $(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}$ and $\widetilde{\mathcal{B}}^{H}(z)$. For clarity, we give the description of $\mathcal{B}_{\mathsf{cac}}$ below.

$\mathcal{B}_{\mathsf{cac}}$ receives $\{\mathsf{k}_i\}_{i \in [4n]}$ from the challenger of $\mathsf{Exp}^{\mathsf{cut\text{-}and\text{-}choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{B}_{\mathsf{cac}}}(\lambda, n)$. $\mathcal{B}_{\mathsf{cac}}$ chooses $K \leftarrow \{0,1\}^{\lambda}$ and simulates $\mathcal{A}$'s random oracle queries before $\mathtt{Reveal}_{\mathsf{sk}}$ occurs by $H_{K\|* \to H'}$ and those after $\mathtt{Reveal}_{\mathsf{sk}}$ occurs by H, which can be done efficiently by Zhandry's compressed oracle technique [Zha19]. $\mathcal{B}_{\mathsf{cac}}$ generates $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^{\lambda})$ and $(\mathsf{ow.pk}, \mathsf{ow.sk}) \leftarrow \mathsf{OW.KeyGen}(1^{\lambda})$, sends $\mathsf{pk} := (\mathsf{nce.pk}, \mathsf{ow.pk})$ to $\mathcal{A}$, and receives (m_0, m_1) from $\mathcal{A}$. $\mathcal{B}_{\mathsf{cac}}$ sends $\{\mathsf{k}_i\}_{i \in [4n]}$ to $\mathcal{A}$ and receives $\{y_i\}_{i \in [4n]}$ from $\mathcal{A}$. $\mathcal{B}_{\mathsf{cac}}$ generates $\mathsf{nce}.\widetilde{\mathsf{CT}} \leftarrow \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux})$, $\mathsf{ow.CT} \leftarrow \mathsf{OW.Enc}(\mathsf{ow.pk}, K)$, and $\mathsf{CT}_{\mathsf{msg}} \leftarrow \{0,1\}^{\ell}$, sends $(\mathsf{nce}.\widetilde{\mathsf{CT}}, \mathsf{ow.CT}, \mathsf{CT}_{\mathsf{msg}})$ to $\mathcal{A}$, and receives $\mathsf{cert} = \{(e_i, d_i)\}_{i \in [4n]}$ from $\mathcal{A}$. $\mathcal{B}_{\mathsf{cac}}$ sends $\{(y_i, e_i, d_i)\}_{i \in [4n]}$ to the challenger of $\mathsf{Exp}^{\mathsf{cut\text{-}and\text{-}choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{B}_{\mathsf{cac}}}(\lambda, n)$. If the challenger aborts, $\mathcal{B}_{\mathsf{cac}}$ aborts. Otherwise, $\mathcal{B}_{\mathsf{cac}}$ receives S from the challenger, generates $\mathsf{nce}.\widetilde{\mathsf{sk}} \leftarrow \mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce}.\widetilde{\mathsf{CT}}, S)$, and sends $\mathsf{sk} := (\mathsf{nce}.\widetilde{\mathsf{sk}}, \mathsf{ow.sk})$ to $\mathcal{A}$. $\mathcal{B}_{\mathsf{cac}}$ chooses $i \leftarrow [q]$, measures $\mathcal{A}$'s i-th query after Step 8, and lets $Z' = (K', (b'_{i_1}, x'_{i_1}), (b'_{i_2}, x'_{i_2}), ..., (b'_{i_{2n}}, x'_{i_{2n}}))$ be the measurement outcome where $i_1, ..., i_{2n}$ are the elements of S in the ascending order. $\mathcal{B}_{\mathsf{cac}}$ sends $\{b'_i, x'_i\}_{i \in S}$ to the challenger.

We can see that $\mathcal{B}_{\mathsf{cac}}$ perfectly simulates the sequential execution of $(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}$ and $\widetilde{\mathcal{B}}^{H}(z)$ conditioned on that it does not abort, which corresponds to the event $\mathtt{Reveal}_{\mathsf{sk}}$ in the simulated experiment for $\mathcal{A}$. Moreover, when $Z' = Z \neq \mathsf{null}$ where Z is defined as in $\mathsf{Hyb}_3(b)$, we have $\mathsf{Chk}_{\mathcal{G}}(\mathsf{k}_i, b'_i, x'_i, y_i) = 1$ for all $i \in S$ by the definition of Z. Therefore, we have

$$\Pr[\mathsf{Exp}^{\mathsf{cut\text{-}and\text{-}choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{B}_{\mathsf{cac}}}(\lambda, n) = 1] \geq \Pr[\mathtt{Reveal}_{\mathsf{sk}}] \cdot \Pr_{(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^{H}(z) = Z].$$

Since both $\Pr[\mathtt{Reveal}_{\mathsf{sk}}]$ and $\Pr_{(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^{H}(z) = Z]$ are noticeable as already proven, $\Pr[\mathsf{Exp}^{\mathsf{cut\text{-}and\text{-}choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{B}_{\mathsf{cac}}}(\lambda, n) = 1]$ is noticeable, which means that $\mathcal{B}_{\mathsf{cac}}$ breaks the cut-and-choose adaptive hardcore property. This completes the proof for the case where $|\Pr[\mathsf{Hyb}_2(b) = 1] - \Pr[\mathsf{Hyb}_3(b) = 1]|$ is noticeable.

When it is non-negligible rather than noticeable, the reason that the above proof does not immediately works is that $\Pr[\mathtt{Reveal}_{\mathsf{sk}}] \cdot \Pr_{(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^{H}(z) = Z]$ may be negligible even if both $\Pr[\mathtt{Reveal}_{\mathsf{sk}}]$ and $\Pr_{(H, z) \leftarrow \mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^{H}(z) =$

$Z]$ are non-negligible in general. Intuitively, we overcome this by observing that $\Pr[\mathtt{Reveal}_{\mathsf{sk}}]$ and $\Pr_{(H,z)\leftarrow\mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^H(z) = Z]$ take "noticeable values" on the same infinite subset of security parameters, in which case the product also takes "noticeable values" on the same subset. More precisely, since we assume that $|\Pr[\mathsf{Hyb}_2(b) = 1] - \Pr[\mathsf{Hyb}_3(b) = 1]|$ is non-negligible, there exists an infinite subset $I \subseteq \mathbb{N}$ and a polynomial p such that $|\Pr[\mathsf{Hyb}_2(b) = 1] - \Pr[\mathsf{Hyb}_3(b) = 1]| \geq 1/p(\lambda)$ for all $\lambda \in I$. By similar arguments as above, we can show that there exists a polynomial p' such that $\Pr[\mathtt{Reveal}_{\mathsf{sk}}] \geq 1/p'(\lambda)$ and $\Pr_{(H,z)\leftarrow\mathcal{D}'_{\mathtt{Reveal}_{\mathsf{sk}}}}[\widetilde{\mathcal{B}}^H(z) = Z] \geq 1/p'(\lambda)$ for all $\lambda \in I$. Then we have $\Pr[\mathsf{Exp}^{\mathsf{cut\text{-}and\text{-}choose}}_{(\mathcal{F},\mathcal{G}),\mathcal{B}_{\mathsf{cac}}}(\lambda, n) = 1] \geq (1/p'(\lambda))^2$ for all $\lambda \in I$, in which case it is non-negligible. This completes the proof. ☐

Proposition 5.11. *It holds that* $\Pr[\mathsf{Hyb}_3(0) = 1] = \Pr[\mathsf{Hyb}_3(1) = 1]$.

Proof. In Hyb_3, the challenger queries H while the adversary queries $H_{K\|*\rightarrow H'}$ or $H_{Z\rightarrow r}$. Therefore, $H(Z)$ is used only for generating $\mathsf{CT}_{\mathsf{msg}}$ in Hyb_3 and thus $\mathsf{CT}_{\mathsf{msg}}$ is an independently uniform string regardless of b from the view of the adversary. Therefore Proposition 5.11 holds. ☐

By combining Propositions 5.8 to 5.11 Theorem 5.7 is proven. ☐

6 PKE with Publicly Verifiable Certified Deletion and Classical Communication

In this section, we define the notion of public key encryption with publicly verifiable certified deletion with classical communication, and construct it from the witness encryption and the one-shot signature. In Section 6.1, we present the definition of the public key encryption with publicly verifiable certified deletion with classical communication. In Section 6.2, we give a construction and show its security.

6.1 Definition of PKE with Publicly Verifiable Certified Deletion with Classical Communication

In this section, we consider a PKE with publicly verifiable certified deletion with classical communication. It is a publicly verifiable version of the one given in (Definition 5.1). The construction of Section 5.3 is not publicly verifiable, because the verification key vk (which is the trapdoor $\{\mathsf{td}_i\}_i$) should be secret to the adversary. On the other hand, in Section 6.2 we construct a publicly verifiable one, which means that the security is kept even if the verification key vk (which is the public key $\mathsf{oss.pk}$ of the one-shot signature in our construction) is given to the adversary.

The definition (syntax) is the same as that of the non-publicly-verifiable one (Definition 5.1). Furthermore, its correctness, i.e., the decryption correctness and the verification correctness, are also the same as those of the non-publicly-verifiable one (Definition 5.2). Regarding the certified deletion security, it is the same as that of the non-publicly-verifiable one (Definition 5.3) except that the challenger sends vk to the adversary (which is $\mathsf{oss.pk}$ in our construction).

6.2 Construction

We construct a PKE scheme with publicly verifiable certified deletion with classical communication $\Sigma_{\mathsf{pvcccd}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ from a public key NCE scheme $\Sigma_{\mathsf{nce}} = \mathsf{NCE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$, the witness encryption scheme $\Sigma_{\mathsf{we}} = \mathsf{WE}.(\mathsf{Enc}, \mathsf{Dec})$, and the one-shot signature scheme $\Sigma_{\mathsf{oss}} = \mathsf{OSS}.(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Vrfy})$.

$\mathsf{KeyGen}(1^\lambda)$**:**

- Generate $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE}.\mathsf{KeyGen}(1^\lambda)$.
- Output $(\mathsf{pk}, \mathsf{sk}) = (\mathsf{nce.pk}, \mathsf{nce.sk})$.

$\mathsf{Enc}\langle\mathcal{S}(\mathsf{pk}, m), \mathcal{R}\rangle$**:** This is an interactive protocol between a sender $\mathcal{S}$ with input (pk, m) and a receiver $\mathcal{R}$ without input that works as follows.

- $\mathcal{S}$ parses $\mathsf{pk} = \mathsf{nce.pk}$.
- $\mathcal{S}$ generates $\mathsf{crs} \leftarrow \mathsf{OSS.Setup}(1^\lambda)$, and sends crs to $\mathcal{R}$.
- $\mathcal{R}$ generates $(\mathsf{oss.pk}, \mathsf{oss.sk}) \leftarrow \mathsf{OSS.KeyGen}(\mathsf{crs})$, sends $\mathsf{oss.pk}$ to $\mathcal{S}$, and keeps $\mathsf{oss.sk}$.
- $\mathcal{S}$ computes $\mathsf{we.CT} \leftarrow \mathsf{WE.Enc}(1^\lambda, x, m)$ with the statement x that "$\exists\sigma$ s.t. $\mathsf{OSS.Vrfy}(\mathsf{crs}, \mathsf{oss.pk}, \sigma, 0) = \top$".
- $\mathcal{S}$ computes $\mathsf{nce.CT} \leftarrow \mathsf{NCE.Enc}(\mathsf{nce.pk}, \mathsf{we.CT})$, and sends $\mathsf{nce.CT}$ to $\mathcal{R}$.
- $\mathcal{S}$ outputs $\mathsf{vk} = (\mathsf{crs}, \mathsf{oss.pk})$. $\mathcal{R}$ outputs $\mathsf{CT} = (\mathsf{nce.CT}, \mathsf{oss.sk})$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$**:**

- Parse $\mathsf{sk} = \mathsf{nce.sk}$ and $\mathsf{CT} = (\mathsf{nce.CT}, \mathsf{oss.sk})$.
- Compute $\sigma \leftarrow \mathsf{OSS.Sign}(\mathsf{oss.sk}, 0)$.
- Compute $m' \leftarrow \mathsf{NCE.Dec}(\mathsf{nce.sk}, \mathsf{nce.CT})$
- Compute $m \leftarrow \mathsf{WE.Dec}(m', \sigma)$.
- Output m.

$\mathsf{Del}(\mathsf{CT})$**:**

- Parse $\mathsf{CT} = (\mathsf{nce.CT}, \mathsf{oss.sk})$.
- Compute $\sigma \leftarrow \mathsf{OSS.Sign}(\mathsf{oss.sk}, 1)$.
- Output $\mathsf{cert} = \sigma$.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert})$**:**

- Parse $\mathsf{vk} = (\mathsf{crs}, \mathsf{oss.pk})$ and $\mathsf{cert} = \sigma$.
- Compute $b \leftarrow \mathsf{OSS.Vrfy}(\mathsf{crs}, \mathsf{oss.pk}, \sigma, 1)$.
- Output b.

Correctness. The decryption and verification correctness easily follow from the correctness of Σ_{we} and Σ_{oss}.

Security. We show the following theorem.

Theorem 6.1. *If Σ_{nce} is RNC secure, Σ_{we} has the extractable security and Σ_{oss} is secure, then Σ_{pvcccd} is IND-CPA-CD secure.*

Proof. For clarity, we describe how $\mathsf{Exp}^{\mathsf{pvccpk\text{-}cert\text{-}del}}_{\Sigma_{\mathsf{pvcccd}},\mathcal{A}}(\lambda, b)$ (which we call $\mathsf{Hyb}_0(b)$ for simplicity) works below.

1. The challenger generates $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^\lambda)$, and sends $\mathsf{nce.pk}$ to $\mathcal{A}$.
2. $\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M}^2$ to the challenger.
3. The challenger generates $\mathsf{crs} \leftarrow \mathsf{OSS.Setup}(1^\lambda)$, and sends crs to $\mathcal{A}$.
4. $\mathcal{A}$ sends $\mathsf{oss.pk}$ to the challenger.
5. The challenger computes $\mathsf{we.CT} \leftarrow \mathsf{WE.Enc}(1^\lambda, x, m_b)$, where x is the statement that "$\exists\sigma$ s.t. $\mathsf{OSS.Vrfy}(\mathsf{crs}, \mathsf{oss.pk}, \sigma, 0) = \top$. The challenger computes $\mathsf{nce.CT} \leftarrow \mathsf{NCE.Enc}(\mathsf{nce.pk}, \mathsf{we.CT})$. The challenger sends $\mathsf{nce.CT}$ to $\mathcal{A}$.
6. $\mathcal{A}$ sends $\mathsf{cert} = \sigma$ to the challenger.
7. The challenger computes $b \leftarrow \mathsf{OSS.Vrfy}(\mathsf{crs}, \mathsf{oss.pk}, \sigma, 1)$. If the output is $b = \bot$, the challenger sends $\bot$ to $\mathcal{A}$. If the output is $b = \top$, the challenger sends $\mathsf{nce.sk}$ to $\mathcal{A}$.

8. $\mathcal{A}$ outputs b'. The output of the experiment is b'.

We define the following hybrid.

$\mathsf{Hyb}_1(b)$**:** This is identical to $\mathsf{Hyb}_0(b)$ except that nce.CT and nce.sk are generated as

$$\begin{aligned}\mathsf{nce.CT} &\leftarrow \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}),\\ \mathsf{nce.sk} &\leftarrow \mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce.CT}, \mathsf{we.CT}).\end{aligned}$$

Proposition 6.2. *If Σ_{nce} is RNC secure,* $|\Pr[\mathsf{Hyb}_0(b) = 1] - \Pr[\mathsf{Hyb}_1(b) = 1]| \le \mathsf{negl}(\lambda)$.

Proof. To show this, we assume that $|\Pr[\mathsf{Hyb}_0(b) = 1] - \Pr[\mathsf{Hyb}_1(b) = 1]|$ is non-negligible, and construct an adversary $\mathcal{B}_{\mathsf{nce}}$ that breaks the RNC security of Σ_{nce}. Let $\mathcal{A}$ be the distinguisher for $\mathsf{Hyb}_0(b)$ and $\mathsf{Hyb}_1(b)$. First, $\mathcal{B}_{\mathsf{nce}}$ receives nce.pk from the challenger of the RNC security game. $\mathcal{B}_{\mathsf{nce}}$ then sends nce.pk to $\mathcal{A}$. $\mathcal{B}_{\mathsf{nce}}$ receives (m_0, m_1) from $\mathcal{A}$. $\mathcal{B}_{\mathsf{nce}}$ generates $\mathsf{crs} \leftarrow \mathsf{OSS.Setup}(1^\lambda)$ and sends crs to $\mathcal{A}$. $\mathcal{B}_{\mathsf{nce}}$ receives oss.pk from $\mathcal{A}$. $\mathcal{B}_{\mathsf{nce}}$ generates $\mathsf{we.CT} \leftarrow \mathsf{WE.Enc}(1^\lambda, x, m_b)$. $\mathcal{B}_{\mathsf{nce}}$ sends we.CT to the challenger of the RNC security game, and receives $(\mathsf{nce.CT}^*, \mathsf{nce.sk}^*)$ from the challenger of the RNC security game. Here,

$$(\mathsf{nce.CT}^*, \mathsf{nce.sk}^*) = (\mathsf{NCE.Enc}(\mathsf{nce.pk}, \mathsf{we.CT}), \mathsf{nce.sk})$$

if the challenger's bit is 0, and

$$(\mathsf{nce.CT}^*, \mathsf{nce.sk}^*) = (\mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}), \mathsf{NCE.Reveal}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}, \mathsf{nce.CT}^*, \mathsf{we.CT}))$$

if the challenger's bit is 1. $\mathcal{B}_{\mathsf{nce}}$ sends $\mathsf{nce.CT}^*$ to $\mathcal{A}$. $\mathcal{B}_{\mathsf{nce}}$ receives $\mathsf{cert} = \sigma$ from $\mathcal{A}$. If $\mathsf{OSS.Vrfy}(\mathsf{crs}, \mathsf{oss.pk}, \sigma, 1) = \bot$, $\mathcal{B}_{\mathsf{nce}}$ sends $\bot$ to $\mathcal{A}$, and otherwise sends $\mathsf{nce.sk}^*$ to $\mathcal{A}$. By assumption, $\mathcal{A}$ can distinguish $\mathsf{Hyb}_0(b)$ and $\mathsf{Hyb}_1(b)$, and therefore $\mathcal{B}_{\mathsf{nce}}$ can output the bit of the challenger of the RNC security game with non-negligible probability, which breaks the RNC security of Σ_{nce}. □

Proposition 6.3. *If Σ_{oss} is secure and Σ_{we} has the extractable security,* $|\Pr[\mathsf{Hyb}_1(0) = 1] - \Pr[\mathsf{Hyb}_1(1) = 1]| \le \mathsf{negl}(\lambda)$.

Proof. Let good (resp. bad) be the event that the adversary in $\mathsf{Hyb}_1(b)$ sends a valid (resp. invalid) cert. Then,

$$\begin{aligned}|\Pr[\mathsf{Hyb}_1(0) = 1] - \Pr[\mathsf{Hyb}_1(1) = 1]| &= |\Pr[\mathsf{Hyb}_1(0) = 1 \land \mathsf{good}] + \Pr[\mathsf{Hyb}_1(0) = 1 \land \mathsf{bad}]\\ &\quad - \Pr[\mathsf{Hyb}_1(1) = 1 \land \mathsf{good}] - \Pr[\mathsf{Hyb}_1(1) = 1 \land \mathsf{bad}]|\\ &\le |\Pr[\mathsf{Hyb}_1(0) = 1 \land \mathsf{good}] - \Pr[\mathsf{Hyb}_1(1) = 1 \land \mathsf{good}]|\\ &\quad + |\Pr[\mathsf{Hyb}_1(0) = 1 \land \mathsf{bad}] - \Pr[\mathsf{Hyb}_1(1) = 1 \land \mathsf{bad}]|\\ &= |\Pr[\mathsf{Hyb}_1(0) = 1 \land \mathsf{good}] - \Pr[\mathsf{Hyb}_1(1) = 1 \land \mathsf{good}]|\end{aligned}$$

Assume that $|\Pr[\mathsf{Hyb}_1(0) = 1] - \Pr[\mathsf{Hyb}_1(1) = 1]|$ is non-negligible, which means that there exists an infinite subset $I \subseteq \mathbb{N}$ and a polynomial p such that $|\Pr[\mathsf{Hyb}_1(0) = 1 \land \mathsf{good}] - \Pr[\mathsf{Hyb}_1(1) = 1 \land \mathsf{good}]| \ge 1/p(\lambda)$ for all $\lambda \in I$. Because

$$|\Pr[\mathsf{Hyb}_1(0) = 1 \land \mathsf{good}] - \Pr[\mathsf{Hyb}_1(1) = 1 \land \mathsf{good}]| = |\Pr[\mathsf{Hyb}_1(0) = 1|\mathsf{good}] - \Pr[\mathsf{Hyb}_1(1) = 1|\mathsf{good}]|\Pr[\mathsf{good}],$$

It means $|\Pr[\mathsf{Hyb}_1(0) = 1|\mathsf{good}] - \Pr[\mathsf{Hyb}_1(1) = 1|\mathsf{good}]| \ge 1/p(\lambda)$ and $\Pr[\mathsf{good}] \ge 1/p(\lambda)$ for all $\lambda \in I$. Let aux be the adversary's internal state after outputting cert conditioned on good. Then, due to the extractability of the witness encryption, there is a QPT extractor $\mathcal{E}$ and polynomial q such that the probability that $\mathcal{E}(1^\lambda, x, \mathsf{aux})$ outputs a valid σ' such that $\mathsf{OSS.Vrfy}(\mathsf{crs}, \mathsf{oss.pk}, \sigma', 0) = \top$ is at least $1/q(\lambda)$ for all $\lambda \in I$.

We can construct an adversary $\mathcal{B}_{\mathsf{oss}}$ that breaks the security of the one-shot signature scheme as follows. $\mathcal{B}_{\mathsf{oss}}$ receives $\mathsf{oss.crs} \leftarrow \mathsf{OSS.Setup}(\lambda)$. It generates $(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^\lambda)$, and sends nce.pk to the adversary of $\mathsf{Hyb}_1(b)$. $\mathcal{B}_{\mathsf{oss}}$ receives m_0, m_1 from the adversary of $\mathsf{Hyb}_1(b)$, and returns oss.crs. $\mathcal{B}_{\mathsf{oss}}$ receives

$\mathsf{oss.pk}$ from the adversary of $\mathsf{Hyb}_1(b)$. $\mathcal{B}_{\mathsf{oss}}$ sends $\mathsf{nce.CT}^* \leftarrow \mathsf{NCE.Fake}(\mathsf{nce.pk}, \mathsf{nce.sk}, \mathsf{nce.aux})$ to the adversary of $\mathsf{Hyb}_1(b)$, and receives cert. $\mathcal{B}_{\mathsf{oss}}$ simulates $\mathcal{E}$ and gets its output σ'. $\mathcal{B}_{\mathsf{oss}}$ outputs $(\mathsf{oss.pk}, 1, \mathsf{cert}, 0, \sigma')$. Because

$$\begin{aligned}&\Pr[m_0 \neq m_1 \wedge \mathsf{Vrfy}(\mathsf{oss.crs}, \mathsf{oss.pk}, 0, \sigma') = \top \wedge \mathsf{Vrfy}(\mathsf{oss.crs}, \mathsf{oss.pk}, 1, \mathsf{cert}) = \top]\\ &= \Pr[\mathsf{good}]\Pr[\mathcal{E}(1^\lambda, x, \mathsf{aux}) \text{ outputs valid } \sigma'] \geq \frac{1}{p(\lambda)}\frac{1}{q(\lambda)}\end{aligned}$$

for all $\lambda \in I$, $\mathcal{B}_{\mathsf{oss}}$ breaks the security of the one-shot signature scheme.

□

By Proposition 6.2 and Proposition 6.3, we obtain Theorem 6.1.

□

Acknowledgement

TM is supported by the MEXT Q-LEAP, JST FOREST, JST PRESTO No.JPMJPR176A, and the Grant-in-Aid for Scientific Research (B) No.JP19H04066 of JSPS.

References

[ABSV15] Prabhanjan Ananth, Zvika Brakerski, Gil Segev, and Vinod Vaikuntanathan. From selective to adaptive security in functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 657–677. Springer, Heidelberg, August 2015. (Cited on page 11.)

[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 1, 2, 8, 14.)

[AHU19] Andris Ambainis, Mike Hamburg, and Dominique Unruh. Quantum security proofs using semi-classical oracles. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 269–295. Springer, Heidelberg, August 2019. (Cited on page 7, 17.)

[AK21] Prabhanjan Ananth and Fatih Kaleoglu. Uncloneable encryption, revisited. *IACR Cryptol. ePrint Arch.*, 2021:412, 2021. (Cited on page 3, 5.)

[AP20] Shweta Agrawal and Alice Pellet-Mary. Indistinguishability obfuscation without maps: Attacks and fixes for noisy linear FE. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part I*, volume 12105 of *LNCS*, pages 110–140. Springer, Heidelberg, May 2020. (Cited on page 10.)

[BB84] Charles H. Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. In *IEEE International Conference on Computers Systems and Signal Processing*, pages 175–179. IEEE, 1984. (Cited on page 1, 3.)

[BCM$^+$18] Zvika Brakerski, Paul Christiano, Urmila Mahadev, Umesh V. Vazirani, and Thomas Vidick. A cryptographic test of quantumness and certifiable randomness from a single quantum device. In Mikkel Thorup, editor, *59th FOCS*, pages 320–331. IEEE Computer Society Press, October 2018. (Cited on page 2, 3, 6, 7, 8, 15.)

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 41–69. Springer, Heidelberg, December 2011. (Cited on page 2, 3, 17.)

[BDGM20] Zvika Brakerski, Nico Döttling, Sanjam Garg, and Giulio Malavolta. Factoring and pairings are not necessary for iO: Circular-secure LWE suffices. Cryptology ePrint Archive, Report 2020/1024, 2020. https://eprint.iacr.org/2020/1024. (Cited on page 10.)

[BGI$^+$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 2, 10.)

[BGMZ18] 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. (Cited on page 10.)

[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 1, 2, 3, 4, 5, 11, 12.)

[BL20] Anne Broadbent and Sébastien Lord. Uncloneable quantum encryption via oracles. In Steven T. Flammia, editor, *15th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2020, June 9-12, 2020, Riga, Latvia*, volume 158 of *LIPIcs*, pages 4:1–4:22. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. (Cited on page 3.)

[BP15] Nir Bitansky and Omer Paneth. ZAPs and non-interactive witness indistinguishability from indistinguishability obfuscation. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 401–427. Springer, Heidelberg, March 2015. (Cited on page 14, 28.)

[BR97] Mihir Bellare and Phillip Rogaway. Collision-resistant hashing: Towards making UOWHFs practical. In Burton S. Kaliski Jr., editor, *CRYPTO'97*, volume 1294 of *LNCS*, pages 470–484. Springer, Heidelberg, August 1997. (Cited on page 3.)

[CCKW19] Alexandru Cojocaru, Léo Colisson, Elham Kashefi, and Petros Wallden. QFactory: Classically-instructed remote secret qubits preparation. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 615–645. Springer, Heidelberg, December 2019. (Cited on page 1, 3.)

[CFGN96] Ran Canetti, Uriel Feige, Oded Goldreich, and Moni Naor. Adaptively secure multi-party computation. In *28th ACM STOC*, pages 639–648. ACM Press, May 1996. (Cited on page 2, 13.)

[CHK05] Ran Canetti, Shai Halevi, and Jonathan Katz. Adaptively-secure, non-interactive public-key encryption. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 150–168. Springer, Heidelberg, February 2005. (Cited on page 2, 4, 5, 13.)

[CHVW19] Yilei Chen, Minki Hhan, Vinod Vaikuntanathan, and Hoeteck Wee. Matrix PRFs: Constructions, attacks, and applications to obfuscation. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part I*, volume 11891 of *LNCS*, pages 55–80. Springer, Heidelberg, December 2019. (Cited on page 10.)

[CMP20] Andrea Coladangelo, Christian Majenz, and Alexander Poremba. Quantum copy-protection of compute-and-compare programs in the quantum random oracle model. *arXiv*, 2009.13865, 2020. (Cited on page 3, 7.)

[CRW19] Xavier Coiteux-Roy and Stefan Wolf. Proving erasure. *2019 IEEE International Symposium on Information Theory (ISIT)*, Jul 2019. (Cited on page 3.)

[FM18] Honghao Fu and Carl A. Miller. Local randomness: Examples and application. *Physical Review A*, 97(3), Mar 2018. (Cited on page 3.)

[GDP16] Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46 (general data protection regulation). *Official Journal of the European Union (OJ)*, pages 1–88, 2016. (Cited on page 1.)

[GGH$^+$16] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. *SIAM J. Comput.*, 45(3):882–929, 2016. (Cited on page 6.)

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986. (Cited on page 20.)

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 467–476. ACM Press, June 2013. (Cited on page 2, 7, 8, 15.)

[GKP$^+$13] Shafi Goldwasser, Yael Tauman Kalai, Raluca A. Popa, Vinod Vaikuntanathan, and Nickolai 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 2, 7, 8, 15.)

[GP21] Romain Gay and Rafael Pass. Indistinguishability obfuscation from circular security. In *STOC 2021 (to appear)*, 2021. (Cited on page 10.)

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 2006*, pages 89–98. ACM Press, October / November 2006. Available as Cryptology ePrint Archive Report 2006/309. (Cited on page 1, 10.)

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008. (Cited on page 10.)

[GVW15] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. *J. ACM*, 62(6):45:1–45:33, 2015. (Cited on page 11.)

[GZ20] Marios Georgiou and Mark Zhandry. Unclonable decryption keys. Cryptology ePrint Archive, Report 2020/877, 2020. `https://eprint.iacr.org/2020/877`. (Cited on page 3, 8.)

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. (Cited on page 20.)

[HXY19] Minki Hhan, Keita Xagawa, and Takashi Yamakawa. Quantum random oracle model with auxiliary input. In Steven D. Galbraith and Shiho Moriai, editors, *ASIACRYPT 2019, Part I*, volume 11921 of *LNCS*, pages 584–614. Springer, Heidelberg, December 2019. (Cited on page 17.)

[JL00] Stanislaw Jarecki and Anna Lysyanskaya. Adaptively secure threshold cryptography: Introducing concurrency, removing erasures. In Bart Preneel, editor, *EUROCRYPT 2000*, volume 1807 of *LNCS*, pages 221–242. Springer, Heidelberg, May 2000. (Cited on page 2, 4, 13.)

[KNTY19] Fuyuki Kitagawa, Ryo Nishimaki, Keisuke Tanaka, and Takashi Yamakawa. Adaptively secure and succinct functional encryption: Improving security and efficiency, simultaneously. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 521–551. Springer, Heidelberg, August 2019. (Cited on page 2, 5, 13.)

[KNY20] Fuyuki Kitagawa, Ryo Nishimaki, and Takashi Yamakawa. Secure software leasing from standard assumptions. Cryptology ePrint Archive, Report 2020/1314, 2020. https://eprint.iacr.org/2020/1314. (Cited on page 1, 3, 7, 16.)

[KT20] Srijita Kundu and Ernest Tan. Composably secure device-independent encryption with certified deletion. *arXiv*, 2011.12704, 2020. (Cited on page 3.)

[Mah18] Urmila Mahadev. Classical verification of quantum computations. In Mikkel Thorup, editor, *59th FOCS*, pages 259–267. IEEE Computer Society Press, October 2018. (Cited on page 1, 2, 3, 6, 7, 8, 15, 17.)

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *22nd ACM STOC*, pages 427–437. ACM Press, May 1990. (Cited on page 5.)

[NY21] Ryo Nishimaki and Takashi Yamakawa. Quantum encryption with certified deletion: Public key and attribute-based. *IACR Cryptol. ePrint Arch.*, 2021:394, 2021. (Cited on page 1, 6.)

[PS19] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Heidelberg, August 2019. (Cited on page 14.)

[Reg09] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):34:1–34:40, 2009. (Cited on page 10.)

[RS19] Roy Radian and Or Sattath. Semi-quantum money. *arXiv*, abs/1908.08889, 2019. (Cited on page 1, 3, 7, 16.)

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005. (Cited on page 1, 10.)

[Unr15] Dominique Unruh. Revocable quantum timed-release encryption. *J. ACM*, 62(6):49:1–49:76, 2015. (Cited on page 3, 7, 17.)

[Wat15] Brent Waters. A punctured programming approach to adaptively secure functional encryption. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 678–697. Springer, Heidelberg, August 2015. (Cited on page 11.)

[Wie83] Stephen Wiesner. Conjugate coding. *SIGACT News*, 15(1):78–88, 1983. (Cited on page 1, 3.)

[WW21] Hoeteck Wee and Daniel Wichs. Candidate obfuscation via oblivious LWE sampling. In *Eurocrypt 2021 (to appear)*, 2021. (Cited on page 10.)

[Zha19] Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 239–268. Springer, Heidelberg, August 2019. (Cited on page 17, 37, 38, 39.)

A Reusable SKE with Certified Deletion

We present the definition and construction of reusable SKE with certified deletion in this section. First, we recall the definition of secret key NCE.

Definition A.1 (Secret Key NCE (Syntax)). *A secret key NCE scheme is a tuple of PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ *with plaintext space* $\mathcal{M}$.

$\mathsf{KeyGen}(1^\lambda) \rightarrow (\mathsf{ek}, \mathsf{dk}, \mathsf{aux})$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a key pair* $(\mathsf{ek}, \mathsf{dk})$ *and an auxiliary information* aux.

$\mathsf{Enc}(\mathsf{ek}, m) \to \mathsf{CT}$**:** *The encryption algorithm takes as input* ek *and a plaintext* $m \in \mathcal{M}$ *and outputs a ciphertext* CT.

$\mathsf{Dec}(\mathsf{dk}, \mathsf{CT}) \to m'$ **or** $\bot$**:** *The decryption algorithm takes as input* dk *and* CT *and outputs a plaintext* m' *or* $\bot$.

$\mathsf{Fake}(\mathsf{ek}, \mathsf{aux}) \to \widetilde{\mathsf{CT}}$**:** *The fake encryption algorithm takes* dk *and* aux*, and outputs a fake ciphertext* $\widetilde{\mathsf{CT}}$.

$\mathsf{Reveal}(\mathsf{ek}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m) \to \widetilde{\mathsf{dk}}$**:** *The reveal algorithm takes* $\mathsf{ek}, \mathsf{aux}, \widetilde{\mathsf{CT}}$ *and* m*, and outputs a fake secret key* $\widetilde{\mathsf{dk}}$.

Correctness is similar to that of PKE, so we omit it.

Definition A.2 (Receiver Non-Committing (RNC) Security for SKE). *A secret key NCE scheme is RNC secure if it satisfies the following. Let* $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ *be a secret key NCE scheme. We consider the following security experiment* $\mathsf{Exp}^{\mathsf{sk\text{-}rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda, b)$.

1. *The challenger computes* $(\mathsf{ek}, \mathsf{dk}, \mathsf{aux}) \leftarrow \mathsf{KeyGen}(1^\lambda)$ *and sends* 1^λ *to the adversary* $\mathcal{A}$.
2. $\mathcal{A}$ *sends an encryption query* m *to the challenger. The challenger computes and returns* $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{ek}, m)$ *to* $\mathcal{A}$*. This process can be repeated polynomially many times.*
3. $\mathcal{A}$ *sends a query* $m \in \mathcal{M}$ *to the challenger.*
4. *The challenger does the following.*
   - *If* $b = 0$*, the challenger generates* $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{ek}, m)$ *and returns* $(\mathsf{CT}, \mathsf{dk})$ *to* $\mathcal{A}$.
   - *If* $b = 1$*, the challenger generates* $\widetilde{\mathsf{CT}} \leftarrow \mathsf{Fake}(\mathsf{ek}, \mathsf{aux})$ *and* $\widetilde{\mathsf{dk}} \leftarrow \mathsf{Reveal}(\mathsf{ek}, \mathsf{aux}, \widetilde{\mathsf{CT}}, m)$ *and returns* $(\widetilde{\mathsf{CT}}, \widetilde{\mathsf{dk}})$ *to* $\mathcal{A}$.
5. *Again* $\mathcal{A}$ *can send encryption queries.*
6. $\mathcal{A}$ *outputs* $b' \in \{0, 1\}$.

Let $\mathsf{Adv}^{\mathsf{sk\text{-}rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda)$ *be the advantage of the experiment above. We say that the* Σ *is RNC secure if for any QPT adversary, it holds that*

$$\mathsf{Adv}^{\mathsf{sk\text{-}rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{sk\text{-}rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{sk\text{-}rec\text{-}nc}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \leq \mathsf{negl}(\lambda).$$

Definition A.3 (Reusable SKE with Certified Deletion (Syntax)). *A secret key encryption scheme with certified deletion is a tuple of quantum algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ *with plaintext space* $\mathcal{M}$ *and key space* $\mathcal{K}$.

$\mathsf{KeyGen}(1^\lambda) \to \mathsf{sk}$**:** *The key generation algorithm takes as input the security parameter* 1^λ *and outputs a secret key* $\mathsf{sk} \in \mathcal{K}$.

$\mathsf{Enc}(\mathsf{sk}, m) \to (\mathsf{vk}, \mathsf{CT})$**:** *The encryption algorithm takes as input* sk *and a plaintext* $m \in \mathcal{M}$ *and outputs a verification key* vk *and a ciphertext* CT.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT}) \to m'$**:** *The decryption algorithm takes as input* sk *and* CT *and outputs a plaintext* $m' \in \mathcal{M}$ *or* $\bot$.

$\mathsf{Del}(\mathsf{CT}) \to \mathsf{cert}$**:** *The deletion algorithm takes as input* CT *and outputs a certification* cert.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert}) \to \top$ **or** $\bot$**:** *The verification algorithm takes* vk *and* cert *and outputs* $\top$ *or* $\bot$.

A difference the definition by Broadbent and Islam and ours is that Enc outputs not only CT but also vk, and vk is used in Vrfy istead of sk.

Definition A.4 (Correctness for reusable SKE with Certified Deletion). *There are two types of correctness. One is decryption correctness and the other is verification correctness.*

Decryption correctness: *For any $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$,*

$$\Pr\left[\mathsf{Dec}(\mathsf{sk},\mathsf{CT}) \neq m \;\middle|\; \begin{array}{l} \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (\mathsf{vk},\mathsf{CT}) \leftarrow \mathsf{Enc}(\mathsf{sk}, m) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Verification correctness: *For any $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$,*

$$\Pr\left[\mathsf{Vrfy}(\mathsf{vk},\mathsf{cert}) = \bot \;\middle|\; \begin{array}{l} \mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda) \\ (\mathsf{vk},\mathsf{CT}) \leftarrow \mathsf{Enc}(\mathsf{sk}, m) \\ \mathsf{cert} \leftarrow \mathsf{Del}(\mathsf{CT}) \end{array}\right] \leq \mathsf{negl}(\lambda).$$

Definition A.5 (Certified Deletion Security for Reusable SKE). *Let $\Sigma = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ be a secret key encryption with certified deletion. We consider the following security experiment $\mathsf{Exp}^{\mathsf{sk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, b)$.*

1. *The challenger computes $\mathsf{sk} \leftarrow \mathsf{KeyGen}(1^\lambda)$.*
2. *$\mathcal{A}$ sends an encryption query m to the challenger. The challenger computes $(\mathsf{vk},\mathsf{CT}) \leftarrow \mathsf{Enc}(\mathsf{sk}, m)$ to $\mathcal{A}$ and returns $(\mathsf{vk},\mathsf{CT})$ to $\mathcal{A}$. This process can be repeated polynomially many times.*
3. *$\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M}^2$ to the challenger.*
4. *The challenger computes $(\mathsf{vk}_b, \mathsf{CT}_b) \leftarrow \mathsf{Enc}(\mathsf{sk}, m_b)$ and sends CT_b to $\mathcal{A}$.*
5. *Again, $\mathcal{A}$ can send encryption queries.*
6. *At some point, $\mathcal{A}$ sends cert to the challenger.*
7. *The challenger computes $\mathsf{Vrfy}(\mathsf{vk}_b, \mathsf{cert})$. If the output is $\bot$, the challenger sends $\bot$ to $\mathcal{A}$. If the output is $\top$, the challenger sends sk to $\mathcal{A}$.*
8. *If the challenger sends $\bot$ in the previous item, $\mathcal{A}$ can send encryption queries again.*
9. *$\mathcal{A}$ outputs $b' \in \{0,1\}$.*

Let $\mathsf{Adv}^{\mathsf{sk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda)$ be the advantage of the experiment above. We say that the Σ is IND-CPA-CD secure if for any QPT $\mathcal{A}$, it holds that

$$\mathsf{Adv}^{\mathsf{sk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda) \coloneqq |\Pr[\mathsf{Exp}^{\mathsf{sk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{sk\text{-}cert\text{-}del}}_{\Sigma,\mathcal{A}}(\lambda, 1) = 1]| \leq \mathsf{negl}(\lambda).$$

Our reusable SKE scheme. We construct $\Sigma_{\mathsf{rskcd}} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ with plaintext space $\mathcal{M}$ from an one-time SKE with certified deletion scheme $\Sigma_{\mathsf{oskcd}} = \mathsf{SKE}.(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Del}, \mathsf{Vrfy})$ with plaintext space $\mathcal{M}$ and key space $\mathcal{K}$ and a secret key NCE scheme $\Sigma_{\mathsf{nce}} = \mathsf{NCE}.(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fake}, \mathsf{Reveal})$ with plaintext space $\mathcal{K}$.

$\mathsf{KeyGen}(1^\lambda)$**:**

- Generate $(\mathsf{nce.ek}, \mathsf{nce.dk}, \mathsf{nce.aux}) \leftarrow \mathsf{NCE.KeyGen}(1^\lambda)$ and output $\mathsf{sk} \coloneqq (\mathsf{nce.ek}, \mathsf{nce.dk})$.

$\mathsf{Enc}(\mathsf{sk}, m)$**:**

- Parse $\mathsf{sk} = (\mathsf{nce.ek}, \mathsf{nce.dk})$.
- Generate $\mathsf{oske.sk} \leftarrow \mathsf{SKE.Gen}(1^\lambda)$.
- Compute $\mathsf{nce.CT} \leftarrow \mathsf{NCE.Enc}(\mathsf{nce.ek}, \mathsf{oske.sk})$ and $\mathsf{oske.CT} \leftarrow \mathsf{SKE.Enc}(\mathsf{oske.sk}, m)$.
- Output $\mathsf{CT} \coloneqq (\mathsf{nce.CT}, \mathsf{oske.CT})$ and $\mathsf{vk} \coloneqq \mathsf{oske.sk}$.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{CT})$**:**

- Parse $\mathsf{sk} = (\mathsf{nce.ek}, \mathsf{nce.dk})$ and $\mathsf{CT} = (\mathsf{nce.CT}, \mathsf{oske.CT})$.
- Compute $\mathsf{sk}' \leftarrow \mathsf{NCE.Dec}(\mathsf{nce.dk}, \mathsf{nce.CT})$.
- Compute and output $m' \leftarrow \mathsf{SKE.Dec}(\mathsf{sk}', \mathsf{oske.CT})$.

$\mathsf{Del}(\mathsf{CT})$**:**

- Parse $\mathsf{CT} = (\mathsf{nce.CT}, \mathsf{oske.CT})$.
- Generate $\mathsf{oske.cert} \leftarrow \mathsf{SKE.Del}(\mathsf{oske.CT})$.
- Output $\mathsf{cert} := \mathsf{oske.cert}$.

$\mathsf{Vrfy}(\mathsf{vk}, \mathsf{cert})$**:**

- Parse $\mathsf{vk} = \mathsf{oske.sk}$ and $\mathsf{cert} = \mathsf{oske.cert}$.
- Output $b \leftarrow \mathsf{SKE.Vrfy}(\mathsf{oske.sk}, \mathsf{oske.cert})$.

Theorem A.6. *If Σ_{nce} is RNC secure and Σ_{oskcd} is OT-CD secure, Σ_{rskcd} is IND-CPA-CD secure.*

We omit the proof of this theorem since it is almost the same as the proof of Theorem 3.4.