Simulation-Based Receiver Selective Opening CCA Secure PKE from Standard Computational Assumptions

Keisuke Hara[1,2(✉)], Fuyuki Kitagawa[1,2], Takahiro Matsuda[2], Goichiro Hanaoka[2], and Keisuke Tanaka[1]

[1] Tokyo Institute of Technology, Tokyo, Japan
hara.k.am@m.titech.ac.jp, {kitagaw1,keisuke}@is.titech.ac.jp
[2] National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
{t-matsuda,hanaoka-goichiro}@aist.go.jp

Abstract. In the situation where there are one sender and multiple receivers, a receiver selective opening (RSO) attack for a public key encryption (PKE) scheme considers adversaries that can corrupt some of the receivers and get their secret keys and plaintexts. Security against RSO attacks for a PKE scheme ensures confidentiality of ciphertexts of uncorrupted receivers. Simulation-based RSO security against chosen ciphertext attacks (SIM-RSO-CCA) is the strongest security notion in all RSO attack scenarios. Jia, Lu, and Li (INDOCRYPT 2016) proposed the first SIM-RSO-CCA secure PKE scheme. However, their scheme used indistinguishability obfuscation, which is not known to be constructed from any standard computational assumption. In this paper, we propose two constructions of SIM-RSO-CCA secure PKE from standard computational assumptions. First, we propose a generic construction of SIM-RSO-CCA secure PKE using an IND-CPA secure PKE scheme and a non-interactive zero-knowledge proof system satisfying one-time simulation soundness. Second, we propose an efficient concrete construction of SIM-RSO-CCA secure PKE based on the decisional Diffie-Hellman assumption.

1 Introduction

1.1 Background and Motivation

In the context of public key encryption (PKE), the generally accepted security notions are IND-CPA and IND-CCA security [10,12]. However, Bellare, Hofheinz, and Yilek [4] pointed out that IND-CPA and IND-CCA security might not be strong enough when considering *Selective Opening (SO)* attacks in a multi-user scenario. Intuitively, SO attacks consider the corruptions of some fraction of users and the extortions of their secret information. Motivated by the

D. Catalano and R. De Prisco (Eds.): SCN 2018, LNCS 11035, pp. 140–159, 2018.
https://doi.org/10.1007/978-3-319-98113-0_8

above problem, they firstly introduced SO security for PKE. Even if an adversary can mount SO attacks, SO security can guarantee confidentiality of ciphertexts of uncorrupted users. In practice, considering secret communication among many users, we should take account of information leakage from some users. Therefore, SO security is an important security notion for PKE in practice. To date, two settings have been considered for SO security: *Sender Selective Opening (SSO) security* [4,5] and *Receiver Selective Opening (RSO) security* [3,15]. The main focus in this paper is on RSO security. In the situation where one sender and multiple receivers exist, RSO security guarantees confidentiality of uncorrupted ciphertexts even if an adversary can corrupt some fraction of receivers and get their plaintexts and secret keys. SO security is defined in both the chosen plaintext attack (CPA) and the chosen ciphertext attack (CCA) settings. In order to take active adversaries into account, we should consider CCA security for many situations.

Furthermore, there are two types of definitions for SO security: indistinguishability-based SO security and simulation-based SO security. The definition of indistinguishability-based SO security usually has a restriction for a plaintext distribution that an adversary can choose. More specifically, the definition of indistinguishability-based SO security usually requires the plaintext distribution to satisfy a notion called *efficient resamplability* [4]. Intuitively, efficient resamplability requires a plaintext distribution to be such that even if some plaintexts are fixed, the other plaintexts can be efficiently sampled. This requirement is somewhat artificial and limits application scenarios since plaintext distributions appearing in practice do not necessarily satisfy this requirement.

On the other hand, simulation-based SO security does not have such a restriction on the plaintext distribution. This security requires that the output of any adversary that is given the public keys, ciphertexts, and plaintexts and secret information of corrupted users, can be simulated by a simulator which only takes the corrupted plaintexts as its input. The secret information corresponds to randomnesses (used in encryptions) of the senders in the SSO setting and secret keys of the receivers in the RSO setting, respectively. Compared to indistinguishability-based SO security, simulation-based SO security can guarantee security even if an adversary chooses an arbitrary plaintext distribution. Since there is no restriction on the plaintext distributions, we can say that simulation-based SO security is preferable to indistinguishability-based SO security considering the utilization of PKE. Also, the previous works [3,15] showed that simulation-based SO security is stronger than indistinguishability-based SO security in the CPA setting. It seems that this implication also holds in the CCA setting.

From the above arguments, we aim to achieve simulation-based RSO security against chosen ciphertext attacks which we call SIM-RSO-CCA security for PKE. So far, the only construction of SIM-RSO-CCA secure PKE is of Jia, Lu, and Li [16], but their construction is based on a very strong cryptographic primitive, *indistinguishability obfuscation (iO)* [2,11]. This primitive is not known to be constructed from standard computational assumptions. Hence, in this paper, we

tackle the following question: *Is it possible to construct a SIM-RSO-CCA secure PKE scheme from standard computational assumptions?*

1.2 Our Contributions

Based on the above motivation, we give affirmative answers to the question. More specifically, our technical results consist of the following three parts.

SIM-RSO-CCA *Security Derived from* RNC-CCA *Security.* As our first technical result, we introduce a new security notion that we call RNC-CCA *security* for receiver non-committing encryption (RNCE) [6, Sect. 4], which is a variant of PKE with a special non-committing property. Then, we show that RNC-CCA secure RNCE implies SIM-RSO-CCA secure PKE. When considering SIM-RSO-CCA security for PKE, we must take into account information of multiple users, a simulator, and an adversary. Thus, if we try to prove SIM-RSO-CCA security directly from standard computational assumptions, security proofs could become very complex. The merit of considering RNCE with our new security notion is that the definition of RNC-CCA security involves only a single user, a single adversary, and no simulator. Hence, we can potentially avoid a complex security proof when proving RNC-CCA security from standard computational assumptions. We believe that this result gives us a guideline for constructing a new SIM-RSO-CCA secure PKE scheme, and in fact, our proposed SIM-RSO-CCA secure PKE schemes are obtained via this result, as explained below.

A Generic Construction of RNC-CCA *Secure RNCE.* As our second technical result, we show a generic construction of RNC-CCA secure RNCE using an IND-CPA secure PKE scheme and a non-interactive zero-knowledge (NIZK) proof system satisfying one-time simulation soundness. (In the following, we call this primitive an OTSS-NIZK for simplicity.) This primitive is slightly stronger than a normal NIZK proof system. However, the constructions of this primitive based on various standard assumptions are known [13,14,19]. Therefore, our second technical result shows that we can construct RNC-CCA secure RNCE schemes from various standard assumptions through our generic construction.

An Efficient Concrete Construction of RNC-CCA *Secure RNCE.* Although our generic construction of RNC-CCA secure RNCE can be instantiated based on standard computational assumptions, we require an NIZK proof system as a building block. In general, NIZK proof systems are not very efficient, and thus the above construction does not necessarily lead to an efficient construction. Thus, as our third technical result, we show an efficient concrete construction of RNC-CCA secure RNCE based on the decisional Diffie-Hellman (DDH) assumption. This scheme is a variant of the Cramer-Shoup encryption scheme [7], and thus we do not need general NIZK proof systems. (We note that this efficient concrete construction supports only a polynomial-sized plaintext space.)

In summary, combining our first and second technical results, we obtain the first generic construction of SIM-RSO-CCA secure PKE from an IND-CPA

secure PKE scheme and an OTSS-NIZK. This result enables us to construct SIM-RSO-CCA secure PKE from various standard computational assumptions. Moreover, combining our first and third technical results, we obtain the first efficient concrete construction of SIM-RSO-CCA secure PKE (with a polynomial-sized plaintext space) from the DDH assumption.

1.3 Technical Overview

As mentioned earlier, Jia et al. [16] proposed the first SIM-RSO-CCA secure PKE scheme using iO. They pointed out that there exist common features between an IND-CCA security proof and a SIM-RSO security proof. To date, there are three major techniques for constructing IND-CCA secure PKE schemes: the double encryption technique [26], the hash proof system (HPS) technique [8], and the all-but-one (ABO) technique [24,25]. Sahai and Waters [27] pointed out that the "punctured programming" paradigm is compatible with iO when constructing various cryptographic primitives, and they in particular constructed an IND-CCA secure PKE scheme based on iO. Jia et al.'s SIM-RSO-CCA secure PKE scheme is obtained from the Sahai-Waters PKE scheme. Since the ABO technique has some similarity to the punctured programming paradigm, in retrospect, Jia et al.'s PKE scheme can be viewed as constructed via the ABO technique.

In contrast to their approach, we take two different paths of constructing SIM-RSO-CCA secure PKE schemes, that is, the double encryption technique and the HPS technique. Somewhat surprisingly, our SIM-RSO-CCA secure PKE schemes only require underlying cryptographic primitives that were required to construct IND-CCA secure PKE schemes. In particular, our constructions do not need any other strong cryptographic primitives, such as iO, for achieving SIM-RSO-CCA security.

In order to take the above approach, we adopt another strategy proposed by Hazay, Patra, and Warinschi [15], who pointed out that RNCE [6, Sect. 4] is an appropriate cryptographic primitive for achieving RSO security. Concretely, they showed that CPA secure RNCE implies SIM-RSO-CPA secure PKE. Inspired by their idea, we formalize a new security notion for RNCE which we call RNC-CCA security, and show that RNC-CCA secure RNCE implies SIM-RSO-CCA secure PKE. Then, we propose a generic construction and an efficient concrete construction of RNC-CCA secure RNCE based on the double encryption technique and the HPS technique, respectively.

The Features of RNCE. Here, we explain the features of RNCE. Informally, RNCE is special PKE having the following two algorithms, Fake and Open.[1] Fake is the fake encryption algorithm that takes a public key and a trapdoor information (generated at the key generation) as input, and outputs a *fake ciphertext* which has no information about a plaintext. Open is the opening algorithm that takes a public key, a trapdoor information, the fake ciphertext, and a certain

[1] In fact, our syntax of RNCE has additional algorithms FKG and FDec. These algorithms are needed for defining RNC-CCA security. See Sect. 3 for the details.

plaintext m as input, and outputs a *fake secret key* which decrypts the *fake ciphertext* to the plaintext m.

RNCE requires the following two security properties. The first one is that an adversary cannot distinguish a real ciphertext generated by the ordinary encryption algorithm and a fake ciphertext generated by Fake. The second one is that an adversary cannot distinguish a real secret key generated by the ordinary key generation algorithm and a fake secret key generated by Open. Canetti, Halevi, and Katz [6, Sect. 4.1] firstly introduced RNCE and a security notion for it considering only non-adaptive chosen ciphertext attacks (CCA1). We extend their security notion to RNC-CCA security considering adaptive chosen ciphertext attacks.

Sufficient Condition for SIM-RSO-CCA *Secure PKE.* We briefly review the security definition of RNCE. Informally, if only considering CPA, the security of RNCE is defined using an experiment that proceeds as follows.

1. An adversary is given a public key and chooses an arbitrary plaintext from the plaintext space.
2. The adversary is given either a real ciphertext or a fake ciphertext depending on the challenge bit chosen uniformly at random.
3. The adversary is given either a real secret key or a fake secret key depending on the above challenge bit.
4. The adversary guesses whether the given ciphertext and secret key are real or fake.

When defining RNC-CCA security for RNCE, it is natural to consider a definition in which an adversary is allowed to make a decryption query at any time in the above security experiment. If we define such a security experiment, an adversary can make a decryption query after he gets a secret key. Therefore, when we show that RNC-CCA secure RNCE implies SIM-RSO-CCA secure PKE, an adversary of RNC-CCA security can perfectly simulate the decryption oracle for a SIM-RSO-CCA adversary.

However, there is one technical problem if we adopt the above definition. The problem is that we cannot obtain an efficient concrete construction of RNCE from the HPS technique. More specifically, it seems hard to construct an RNCE scheme based on the Cramer-Shoup encryption scheme [7]. The critical problem is that when proving the CCA security of the Cramer-Shoup encryption scheme, we use the fact that the entropy of the secret key is sufficiently large. In the security experiment of RNCE, an adversary gets the secret key used in the experiment, and thus the entropy of the secret key is completely lost and the security proof fails if we adopt the above definition.

In order to circumvent the above problem, we define the security experiment for RNC-CCA security of an RNCE scheme so that an adversary is not allowed to make decryption queries after he gets the secret key. Adopting this security definition, we do not have to simulate the decryption oracle for the adversary after he gets the secret key, and we can complete the security proof of our RNCE scheme. See Sect. 5 for the details.

Here, one might have the following question: Can we show that RNC-CCA security implies SIM-RSO-CCA security when adopting the above modified definition for RNC-CCA security? We show an affirmative answer to this question. In a nutshell, we do not have to simulate the decryption queries which are relative to the secret keys of corrupted users in the definition of SIM-RSO-CCA security, and thus we can still show that RNC-CCA secure RNCE implies SIM-RSO-CCA secure PKE. See Sect. 3 for the details.

How to Derive RNC-CCA *Secure RNCE from the Double Encryption Technique.* Here, we give an overview of our generic construction of RNC-CCA secure RNCE derived from the classical double encryption technique [23,26]. One can see that our generic construction is an extension of a CPA secure RNCE scheme observed by Canetti et al. [6, Sect. 4.1]. Their RNCE scheme is inspired by the double encryption technique without considering CCA security. The trick for the non-committing property of their construction is that the secret key used in the decryption algorithm is chosen at random from the two underlying secret keys, and thus their scheme is very simple. In order to upgrade the CPA security of this RNCE scheme to CCA security, we focus on the work by Lindell [19] who constructed an IND-CCA secure PKE scheme based on an IND-CPA secure PKE scheme and an OTSS-NIZK using the double encryption technique. Applying a similar method to the above RNCE scheme, we obtain our generic construction of RNC-CCA secure RNCE. See Sect. 4 for the details.

We note that the technique for achieving the non-committing property, i.e., generating multiple secret keys and using only one of them for decryption, has been adopted in a number of works, e.g., in the construction of an adaptively and forward secure key-evolving encryption scheme [6, Sect. 3], and more recently in the construction of a tightly secure key encapsulation mechanism in the multi-user setting with corruption [1]. Furthermore, our construction shares an idea of binding two ciphertexts with an NIZK proof system with [6, Sect. 3] to resist against active behaviors of an adversary (e.g., decryption queries). However, one difference is that we require one-time simulation-soundness for the underlying NIZK proof system, while they require unbounded simulation-soundness.

How to Derive RNC-CCA *Secure RNCE from the HPS Technique.* Here, we explain an overview of our concrete construction of RNC-CCA secure RNCE derived from the HPS technique [7,8]. Our concrete construction is an extension of the CCA1 secure RNCE scheme proposed by Canetti et al. [6, Sect. 4.2]. Their RNCE scheme is a variant of the Cramer-Shoup-"lite" encryption scheme [7], which is an IND-CCA1 secure PKE scheme based on the DDH assumption. The only difference is that they encode a plaintext m by the group element g^m, where g is a generator of the underlying group. This encoding is essential for the opening algorithm Open of their proposed scheme, and the plaintext space of their scheme is of polynomial-size since they have to compute the discrete logarithm of g^m in the decryption procedure. We extend their scheme to a CCA secure RNCE scheme based on the "full"-Cramer-Shoup encryption scheme [7]. See Sect. 5 for the details.

1.4 Related Work

To date, SSO secure PKE schemes have been extensively studied, and several constructions of SIM-SSO-CCA secure PKE have been shown based on various standard computational assumptions [18,20–22]. On the other hand, RSO secure PKE schemes have been much less studied.

As mentioned above, the only existing construction of SIM-RSO-CCA secure PKE is the construction using iO proposed by Jia et al. [16]. Jia, Lu, and Li [17] proposed indistinguishability-based RSO-CCA (IND-RSO-CCA) secure PKE schemes based on standard computational assumptions. Concretely, they showed two generic constructions of IND-RSO-CCA secure PKE schemes. First, they gave a generic construction based on an IND-RSO-CPA secure PKE scheme, an IND-CCA secure PKE scheme, an NIZK proof system, and a strong one-time signature scheme. Second, they gave a generic construction based on universal HPS. It is not obvious whether their schemes (can be easily extended to) satisfy SIM-RSO-CCA security.

1.5 Organization

The rest of the paper is organized as follows: In Sect. 2, we review the notations and definitions of cryptographic primitives. In Sect. 3, we introduce RNC-CCA security for RNCE and show its implication to SIM-RSO-CCA security for PKE. In Sect. 4, we show a generic construction of RNC-CCA secure RNCE with a binary plaintext space, which is constructed from an IND-CPA secure PKE scheme and an OTSS-NIZK. In Sect. 5, we show a DDH-based concrete construction of RNC-CCA secure RNCE.

2 Preliminaries

In this section, we define some notations and cryptographic primitives.

2.1 Notations

In this paper, $x \leftarrow X$ denotes sampling an element from a finite set X uniformly at random. $y \leftarrow \mathcal{A}(x;r)$ denotes that a probabilistic algorithm $\mathcal{A}$ outputs y for an input x using a randomness r, and we simply denote $y \leftarrow \mathcal{A}(x)$ when we need not write an internal randomness explicitly. For strings x and y, $x\|y$ denotes the concatenation of x and y, and $x := y$ denotes the substitution y for x. In other cases, $x := y$ denotes that x is defined as y. λ denotes a security parameter. A function $f(\lambda)$ is a negligible function in λ, if $f(\lambda)$ tends to 0 faster than $\frac{1}{\lambda^c}$ for every constant $c > 0$. $\mathsf{negl}(\lambda)$ denotes an unspecified negligible function. PPT stands for probabilistic polynomial time. If n, a, b are integers such that $a \leq b$, $[n]$ denotes the set of integers $\{1, \cdots, n\}$ and $[a, b]$ denotes the set of integers $\{a, \cdots, b\}$. If $\mathbf{m} = (m_1, \cdots, m_n)$ is an n-dimensional vector, $\mathbf{m}_J$ denotes the subset $\{m_j\}_{j \in J}$ where $J \subseteq [n]$. If $\mathcal{O}$ is a function or an algorithm and $\mathcal{A}$ is an algorithm, $\mathcal{A}^{\mathcal{O}}$ denotes that $\mathcal{A}$ has oracle access to $\mathcal{O}$.

2.2 Public Key Encryption

A public key encryption (PKE) scheme with a plaintext space $\mathcal{M}$ consists of a tuple of three PPT algorithms $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$. KG is the key generation algorithm that, given a security parameter 1^λ, outputs a public key pk and a secret key sk. Enc is the encryption algorithm that, given a public key pk and a plaintext $m \in \mathcal{M}$, outputs a ciphertext c. Dec is the (deterministic) decryption algorithm that, given a public key pk, a secret key sk, and a ciphertext c, outputs a plaintext $m \in \{\bot\} \cup \mathcal{M}$. As the correctness for Π, we require that $\mathsf{Dec}(pk, sk, \mathsf{Enc}(pk, m)) = m$ holds for all $\lambda \in \mathbb{N}$, $m \in \mathcal{M}$, and $(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$.

Next, we define IND-CPA and SIM-RSO-CCA security for a PKE scheme.

Definition 1 (IND-CPA **security).** *We say that $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is IND-CPA secure if for any PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi,\mathcal{A}}(\lambda) := 2\Big|\Pr[b \leftarrow \{0,1\}; (pk, sk) \leftarrow \mathsf{KG}(1^\lambda); (m_0^*, m_1^*, \mathsf{st}_1) \leftarrow \mathcal{A}_1(pk); \\ c^* \leftarrow \mathsf{Enc}(pk, m_b^*); b' \leftarrow \mathcal{A}_2(c^*, \mathsf{st}_1) : b = b'] - \frac{1}{2}\Big| = \mathsf{negl}(\lambda),$$

where it is required that $|m_0^*| = |m_1^*|$.

Definition 2 (SIM-RSO-CCA **security).** *Let n be the number of users. For a PKE scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$, an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$, and a simulator $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3)$, we define the following pair of experiments.*

$\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}real}}_{n,\Pi,\mathcal{A}}(\lambda)$:
$(\mathbf{pk}, \mathbf{sk}) := (pk_j, sk_j)_{j\in[n]} \leftarrow (\mathsf{KG}(1^\lambda))_{j\in[n]}$
$(\mathsf{Dist}, \mathsf{st}_1) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{Dec}}(\cdot,\cdot)}(\mathbf{pk})$
$\mathbf{m}^* := (m_j^*)_{j\in[n]} \leftarrow \mathsf{Dist}$
$\mathbf{c}^* := (c_j^*)_{j\in[n]} \leftarrow (\mathsf{Enc}(pk_j, m_j^*))_{j\in[n]}$
$(J, \mathsf{st}_2) \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{Dec}}(\cdot,\cdot)}(\mathbf{c}^*, \mathsf{st}_1)$
$\mathsf{out} \leftarrow \mathcal{A}_3^{\mathcal{O}_{\mathsf{Dec}}(\cdot,\cdot)}(\mathbf{sk}_J, \mathbf{m}_J^*, \mathsf{st}_2)$
Return $(\mathbf{m}^*, \mathsf{Dist}, J, \mathsf{out})$

$\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}sim}}_{n,\Pi,\mathcal{S}}(\lambda)$:
$(\mathsf{Dist}, \mathsf{st}_1) \leftarrow \mathcal{S}_1(1^\lambda)$
$\mathbf{m}^* := (m_j^*)_{j\in[n]} \leftarrow \mathsf{Dist}$
$(J, \mathsf{st}_2) \leftarrow \mathcal{S}_2(\mathsf{st}_1)$
$\mathsf{out} \leftarrow \mathcal{S}_3(\mathbf{m}_J^*, \mathsf{st}_2)$
Return $(\mathbf{m}^*, \mathsf{Dist}, J, \mathsf{out})$

In both of the experiments, we require that the distributions Dist *output by $\mathcal{A}$ and $\mathcal{S}$ be efficiently samplable. In $\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}real}}_{n,\Pi,\mathcal{A}}(\lambda)$, a decryption query (c, j) is answered by $\mathsf{Dec}(pk_j, sk_j, c)$. $\mathcal{A}_2$ and $\mathcal{A}_3$ are not allowed to make a decryption query (c_j^*, j) for any $j \in [n]$. Furthermore, $\mathcal{A}_3$ is not allowed to make a decryption query (c, j) satisfying $j \in J$. (This is without losing generality, since $\mathcal{A}_3$ can decrypt any ciphertext using the given secret keys.)*

We say that Π is SIM-RSO-CCA secure if for any PPT adversary $\mathcal{A}$ and any positive integer $n = n(\lambda)$, there exists a PPT simulator $\mathcal{S}$ such that for any PPT distinguisher $\mathcal{D}$,

$$\mathsf{Adv}^{\mathsf{rso\text{-}cca}}_{n,\Pi,\mathcal{A},\mathcal{S},\mathcal{D}}(\lambda) := |\Pr[\mathcal{D}(\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}real}}_{n,\Pi,\mathcal{A}}(\lambda)) = 1] \\ - \Pr[\mathcal{D}(\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}sim}}_{n,\Pi,\mathcal{S}}(\lambda)) = 1]| = \mathsf{negl}(\lambda).$$

Remark 1. For simplicity, we consider non-adaptive opening queries by an adversary in our experiments. That is, an adversary can make an opening query $J \subseteq [n]$ only at once. However, our constructions of SIM-RSO-CCA secure PKE remain secure even if we consider adaptive opening queries by an adversary.

Remark 2. In this paper, as in the previous works [16,17], we consider only the revelation of secret keys in the definition of SIM-RSO-CCA security. Namely, we assume that an adversary cannot obtain a random coin used for generating a secret key. Hazay, Patra, and Warinschi [15] considered the revelation of both secret keys and random coins used in the key generation algorithm in the RSO-CPA security. If we take into account corruptions of both secret keys and random coins, it seems that we need *key simulatability* [9,15] for building blocks.

2.3 Non-interactive Zero-Knowledge Proof System

Let $\mathcal{R}$ be a binary relation that is efficiently computable, and $\mathcal{L} := \{x | \exists w \text{ s.t. } (x, w) \in \mathcal{R}\}$. A non-interactive proof system for $\mathcal{L}$ consists of a tuple of the following five PPT algorithms $\Phi = (\mathsf{CRSGen}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{SimCRS}, \mathsf{SimPrv})$.

CRSGen: The common reference string (CRS) generation algorithm, given a security parameter 1^λ, outputs a CRS crs.
Prove: The proving algorithm, given a CRS crs, a statement $x \in \mathcal{L}$, and a witness w for the fact that $x \in \mathcal{L}$, outputs a proof π.
Verify: The verification algorithm, given a CRS crs, a statement x, and a proof π, outputs either 1 (meaning "accept") or 0 (meaning "reject").
SimCRS: The simulator's CRS generation algorithm, given a security parameter 1^λ, outputs a simulated CRS crs and a trapdoor key tk.
SimPrv: The simulator's proving algorithm, given a trapdoor key tk and a (possibly false) statement x, outputs a simulated proof π.

As the correctness for Φ, we require that $\mathsf{Verify}(crs, x, \mathsf{Prove}(crs, x, w)) = 1$ holds for all $\lambda \in \mathbb{N}$, all $crs \leftarrow \mathsf{CRSGen}(1^\lambda)$, all statements $x \in \mathcal{L}$, and all witnesses w for the fact that $x \in \mathcal{L}$.

Next, we define the security notions for a non-interactive proof system: *One-time simulation soundness* (OT-SS) and *zero-knowledge* (ZK).

Definition 3 (One-time simulation soundness). *We say that a non-interactive proof system* $\Phi = (\mathsf{CRSGen}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{SimCRS}, \mathsf{SimPrv})$ *satisfies one-time simulation soundness (OT-SS) if for any PPT adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{ot\text{-}ss}}_{\Phi,\mathcal{A}}(\lambda) := \Pr[(crs, tk) \leftarrow \mathsf{SimCRS}(1^\lambda); (x^*, \mathsf{st}_1) \leftarrow \mathcal{A}_1(crs);\\ \pi^* \leftarrow \mathsf{SimPrv}(tk, x^*); (x, \pi) \leftarrow \mathcal{A}_2(\pi^*, \mathsf{st}_1) :\\ (x \notin \mathcal{L}) \wedge (\mathsf{Verify}(crs, x, \pi) = 1) \wedge ((x, \pi) \neq (x^*, \pi^*))] = \mathsf{negl}(\lambda).\end{aligned}$$

Definition 4 (Zero-knowledge). *For a non-interactive proof system* $\Phi = (\mathsf{CRSGen}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{SimCRS}, \mathsf{SimPrv})$ *and an adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, *consider the following pair of experiments.*

$$\begin{array}{l|l}
\mathsf{Exp}^{\mathsf{zk\text{-}real}}_{\Phi,\mathcal{A}}(\lambda): & \mathsf{Exp}^{\mathsf{zk\text{-}sim}}_{\Phi,\mathcal{A}}(\lambda): \\
\quad crs \leftarrow \mathsf{CRSGen}(1^\lambda) & \quad (crs, tk) \leftarrow \mathsf{SimCRS}(1^\lambda) \\
\quad (x, w, \mathsf{st}_1) \leftarrow \mathcal{A}_1(crs) & \quad (x, w, \mathsf{st}_1) \leftarrow \mathcal{A}_1(crs) \\
\quad \pi \leftarrow \mathsf{Prove}(crs, x, w) & \quad \pi \leftarrow \mathsf{SimPrv}(tk, x) \\
\quad b' \leftarrow \mathcal{A}_2(\pi, \mathsf{st}_1) & \quad b' \leftarrow \mathcal{A}_2(\pi, \mathsf{st}_1) \\
\quad \text{Return } b' & \quad \text{Return } b'
\end{array}$$

In both of the experiments, it is required that $x \in \mathcal{L}$ and w is a witness for $x \in \mathcal{L}$. We say that Φ is zero-knowledge (ZK) if for any PPT adversary $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{zk}}_{\Phi,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{zk\text{-}real}}_{\Phi,\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{zk\text{-}sim}}_{\Phi,\mathcal{A}}(\lambda) = 1]| = \mathsf{negl}(\lambda).$$

In this paper, we call a non-interactive proof system satisfying both OT-SS and ZK property an *OTSS-NIZK*.

2.4 "+1"-Decisional Diffie-Hellman (DDH) Assumption

Here, we define the "+1"-DDH assumption. It is straightforward to see this assumption is implied by the standard DDH assumption. This assumption is used to simplify the security proof of our concrete construction in Sect. 5.

Definition 5 ("+1"-DDH assumption). *Let p be a prime number such that $p = \Theta(2^\lambda)$, $\mathbb{G}$ be a multiplicative cyclic group of order p, and $\mathbb{Z}_p$ be the set of integers modulo p. We say that the "+1"-DDH assumption holds in $\mathbb{G}$ if for any PPT adversary $\mathcal{A}$,*

$$\begin{aligned}\mathsf{Adv}^{+1\text{-}\mathsf{ddh}}_{\mathbb{G},\mathcal{A}}(\lambda) := |\Pr[g \leftarrow \mathbb{G}; a \leftarrow \mathbb{Z}_p^*; b \leftarrow \mathbb{Z}_p : \mathcal{A}(g, g^a, g^b, g^{ab}) = 1] \\ - \Pr[g \leftarrow \mathbb{G}; a \leftarrow \mathbb{Z}_p^*; b \leftarrow \mathbb{Z}_p : \mathcal{A}(g, g^a, g^b, g^{ab+1}) = 1]| = \mathsf{negl}(\lambda).\end{aligned}$$

2.5 Collision-Resistant Hash Function

In this section, we recall the definition of a collision-resistant hash function. A hash function consists of a pair of PPT algorithms $\Lambda = (\mathsf{HKG}, \mathsf{Hash})$. HKG is the hash key generation algorithm that, given a security parameter 1^λ, outputs a hash key hk. Hash is the (deterministic) hashing algorithm that, given a hash key hk and a string $x \in \{0,1\}^*$, outputs a hash value $h \in \{0,1\}^\lambda$.

Definition 6 (Collision-resistance). *We say that $\Lambda = (\mathsf{HKG}, \mathsf{Hash})$ is a collision-resistant hash function if for any PPT adversary $\mathcal{A}$,*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{cr}}_{\Lambda,\mathcal{A}}(\lambda) := \Pr[hk \leftarrow \mathsf{HKG}(1^\lambda); (x, x^*) \leftarrow \mathcal{A}(hk) : \\ (\mathsf{Hash}(hk, x) = \mathsf{Hash}(hk, x^*)) \wedge (x \neq x^*)] = \mathsf{negl}(\lambda).\end{aligned}$$

3 CCA Security for Receiver Non-commiting Encryption

In this section, we introduce a new security notion that we call RNC-CCA security for receiver non-commiting encryption (RNCE). Next, we show that RNC-CCA secure RNCE implies SIM-RSO-CCA secure PKE.

3.1 Receiver Non-commiting Encryption

Here, we give definitions of RNCE and RNC-CCA security for this primitive. Informally, RNCE is PKE having the property that it can generate a fake ciphertext which can be later opened to any plaintext (by showing an appropriate secret key). Canetti, Halevi, and Katz [6, Sect. 4.1] gave a definition of RNCE considering security against non-adaptive chosen ciphertext attacks (CCA1). We extend their definition to one considering security against adaptive CCA.

Informally, an RNCE scheme Π consists of the seven PPT algorithms $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FKG}, \mathsf{Fake}, \mathsf{Open}, \mathsf{FDec})$. $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ are the same algorithms as those of a PKE scheme. The remaining four algorithms $(\mathsf{FKG}, \mathsf{Fake}, \mathsf{Open}, \mathsf{FDec})$ are used for defining the security notion of this primitive. Therefore, these algorithms are not used when using this scheme in practice. We note that the definition of RNCE in [6, Sect. 4.1] does not contain FKG and FDec, but they are necessary for our formalization of RNC-CCA security. The formal definition is as follows.

Definition 7 (Receiver non-commiting encryption). *An RNCE scheme Π with a plaintext space $\mathcal{M}$ consists of the following seven PPT algorithms $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FKG}, \mathsf{Fake}, \mathsf{Open}, \mathsf{FDec})$. $(\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ are the same algorithms as those of a PKE scheme. $(\mathsf{FKG}, \mathsf{Fake}, \mathsf{Open}, \mathsf{FDec})$ are defined as follows.*

FKG**:** *The fake key generation algorithm, given a security parameter 1^λ, outputs a public key pk and a trapdoor td.*

Fake**:** *The fake encryption algorithm, given a public key pk and a trapdoor td, outputs a fake ciphertext $\widetilde{c}$.*

Open**:** *The opening algorithm, given a public key pk, a trapdoor td, a fake ciphertext $\widetilde{c}$, and a plaintext m, outputs a fake secret key $\widetilde{sk}$.*

FDec**:** *The fake decryption algorithm, given a public key pk, a trapdoor td, and a ciphertext c, outputs $m \in \{\bot\} \cup \mathcal{M}$.*

Next, we define RNC-CCA security for RNCE as follows.

Definition 8 (RNC-CCA **security).** *For an RNCE scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FKG}, \mathsf{Fake}, \mathsf{Open}, \mathsf{FDec})$ and an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$, consider the following pair of experiments.*

$\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi,\mathcal{A}}(\lambda)$:
$(pk, sk) \leftarrow \mathsf{KG}(1^\lambda)$
$(m^*, \mathsf{st}_1) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{Dec}}(\cdot)}(pk)$
$c^* \leftarrow \mathsf{Enc}(pk, m^*)$
$\mathsf{st}_2 \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{Dec}}(\cdot)}(c^*, \mathsf{st}_1)$
$sk^* := sk$
Return $b' \leftarrow \mathcal{A}_3(sk^*, \mathsf{st}_2)$

$\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi,\mathcal{A}}(\lambda)$:
$(pk, td) \leftarrow \mathsf{FKG}(1^\lambda)$
$(m^*, \mathsf{st}_1) \leftarrow \mathcal{A}_1^{\mathcal{O}_{\mathsf{Dec}}(\cdot)}(pk)$
$c^* \leftarrow \mathsf{Fake}(pk, td)$
$\mathsf{st}_2 \leftarrow \mathcal{A}_2^{\mathcal{O}_{\mathsf{Dec}}(\cdot)}(c^*, \mathsf{st}_1)$
$sk^* \leftarrow \mathsf{Open}(pk, td, c^*, m^*)$
Return $b' \leftarrow \mathcal{A}_3(sk^*, \mathsf{st}_2)$

In $\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi,\mathcal{A}}(\lambda)$, a decryption query c is answered by $\mathsf{Dec}(pk, sk, c)$. On the other hand, in $\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi,\mathcal{A}}(\lambda)$, a decryption query c is answered by $\mathsf{FDec}(pk, td, c)$. In both of the experiments, $\mathcal{A}_2$ is not allowed to make a decryption query $c = c^*$ and $\mathcal{A}_3$ is not allowed to make any decryption query.

We say that Π is RNC-CCA secure if for any PPT adversary $\mathcal{A}$,

$$\mathsf{Adv}^{\mathsf{rnc\text{-}cca}}_{\Pi,\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi,\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi,\mathcal{A}}(\lambda) = 1]| = \mathsf{negl}(\lambda).$$

3.2 RNC-CCA Secure RNCE Implies SIM-RSO-CCA Secure PKE

In this section, we show that an RNC-CCA secure RNCE scheme implies a SIM-RSO-CCA secure PKE scheme. Specifically, we show the following theorem.

Theorem 1. *If an RNCE scheme $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FKG}, \mathsf{Fake}, \mathsf{Open}, \mathsf{FDec})$ is RNC-CCA secure, then $\Pi_{\mathsf{rso}} := (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ is a SIM-RSO-CCA secure PKE scheme.*

Here we describe an intuition of the proof. (Due to the space limitation, the formal proof of Theorem 1 is given in the full version of this paper.) Let n be the number of key pairs and $\mathcal{A}$ be an adversary against the SIM-RSO-CCA security of Π_{rso} in security experiments. In the proof, we firstly construct a PPT simulator $\mathcal{S}$ in $\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}sim}}_{n,\Pi_{\mathsf{rso}},\mathcal{S}}(\lambda)$. Specifically, $\mathcal{S}$ computes fake ciphertexts $(\widetilde{c_j})_{j\in[n]}$ using Fake and fake secret keys $(\widetilde{sk_j})_{j\in J}$ using Open, where J is the set of corrupted indices. Here, $\mathcal{S}$ can perfectly simulate the decryption oracle for $\mathcal{A}$ using the trapdoors $(td_j)_{j\in[n]}$ generated by $\mathcal{S}$.

Next, in order to move from the real experiment $\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}real}}_{n,\Pi_{\mathsf{rso}},\mathcal{A}}(\lambda)$ to the simulated experiment $\mathsf{Exp}^{\mathsf{rso\text{-}cca\text{-}sim}}_{n,\Pi_{\mathsf{rso}},\mathcal{S}}(\lambda)$, we change, step by step, n real challenge ciphertexts $(c^*_j)_{j\in[n]}$ to n fake ciphertexts $(\widetilde{c_j})_{j\in[n]}$ and n real secret keys $(sk_j)_{j\in[n]}$ to n fake secret keys $(\widetilde{sk_j})_{j\in[n]}$ which are given to $\mathcal{A}$, respectively. We can show this by the RNC-CCA security of Π using a hybrid argument. Here, we have to deal with some technically subtle point when simulating the decryption oracle for $\mathcal{A}$. Namely, we have to program the behavior of an adversary $\mathcal{B}$ against the RNC-CCA security of Π depending on whether the index i is contained in the corrupted set J output by $\mathcal{A}_2$, where i is the position that $\mathcal{B}$ embeds his own challenge instance into the challenge instances of $\mathcal{A}$. See the full version of this paper for the details.

4 Our Generic Construction of RNC-CCA Secure RNCE

In this section, we show our generic construction of an RNC-CCA secure RNCE scheme with the plaintext space $\{0,1\}$. First, in Sect. 4.1, we describe our generic construction. Then, in Sect. 4.2, we give a proof of RNC-CCA security for our generic construction.

4.1 The Description of Our Generic Construction

Here, we formally describe our generic construction of an RNC-CCA secure RNCE scheme with the plaintext space $\{0,1\}$. Let $\Pi = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$ be a

$\mathsf{KG}'(1^\lambda)$:	$\mathsf{Enc}'(pk, m)$:	$\mathsf{Dec}'(pk, sk, c)$:
$\alpha \leftarrow \{0,1\}$	$(r_0, r_1) \leftarrow \mathcal{R}_{\Pi}^2$	$x := (pk_0, pk_1, c_0, c_1)$
$(pk_0, sk_0) \leftarrow \mathsf{KG}(1^\lambda)$	$c_0 \leftarrow \mathsf{Enc}(pk_0, m; r_0)$	If $\mathsf{Verify}(crs, x, \pi) = 1$
$(pk_1, sk_1) \leftarrow \mathsf{KG}(1^\lambda)$	$c_1 \leftarrow \mathsf{Enc}(pk_1, m; r_1)$	then
$crs \leftarrow \mathsf{CRSGen}(1^\lambda)$	$x := (pk_0, pk_1, c_0, c_1)$	$m \leftarrow \mathsf{Dec}(pk_\alpha, sk_\alpha, c_\alpha)$
$pk := (pk_0, pk_1, crs)$	$w := (m, r_0, r_1)$	Return m
$sk := (\alpha, sk_\alpha)$	$\pi \leftarrow \mathsf{Prove}(crs, x, w)$	else Return $\perp$
Return (pk, sk)	Return $c := (c_0, c_1, \pi)$	
$\mathsf{FKG}'(1^\lambda)$:	$\mathsf{Fake}'(pk, td)$:	$\mathsf{FDec}'(pk, td, c)$:
$\alpha \leftarrow \{0,1\}$	$c_\alpha \leftarrow \mathsf{Enc}(pk_\alpha, 0)$	$x := (pk_0, pk_1, c_0, c_1)$
$(pk_0, sk_0) \leftarrow \mathsf{KG}(1^\lambda)$	$c_{1\oplus\alpha} \leftarrow \mathsf{Enc}(pk_{1\oplus\alpha}, 1)$	If $\mathsf{Verify}(crs, x, \pi) = 1$
$(pk_1, sk_1) \leftarrow \mathsf{KG}(1^\lambda)$	$x := (pk_0, pk_1, c_0, c_1)$	then
$(crs, tk) \leftarrow \mathsf{SimCRS}(1^\lambda)$	$\pi \leftarrow \mathsf{SimPrv}(tk, x)$	$m \leftarrow \mathsf{Dec}(pk_0, sk_0, c_0)$
$pk := (pk_0, pk_1, crs)$	Return $\widetilde{c} := (c_0, c_1, \pi)$	Return m
$td := (\alpha, sk_0, sk_1, tk)$	$\mathsf{Open}'(pk, td, \widetilde{c}, m)$:	else Return $\perp$
Return (pk, td)	$\widetilde{sk} := (\alpha \oplus m, sk_{\alpha\oplus m})$	
	Return $\widetilde{sk}$	

Fig. 1. Our generic construction of RNC-CCA secure RNCE Π'.

PKE scheme with the plaintext space $\{0,1\}$ and $\mathcal{R}_\Pi$ be a randomness space for the encryption algorithm Enc. Let $\Phi = (\mathsf{CRSGen}, \mathsf{Prove}, \mathsf{Verify}, \mathsf{SimCRS}, \mathsf{SimPrv})$ be a non-interactive proof system for L_{eq}, where

$$L_{eq} := \Big\{ (pk_0, pk_1, c_0, c_1) \mid \exists (m, r_0, r_1) \text{ s.t. } (c_0 = \mathsf{Enc}(pk_0, m; r_0)) \wedge (c_1 = \mathsf{Enc}(pk_1, m; r_1)) \Big\}.$$

Then, we construct an RNCE scheme $\Pi' = (\mathsf{KG}', \mathsf{Enc}', \mathsf{Dec}', \mathsf{FKG}', \mathsf{Fake}', \mathsf{Open}', \mathsf{FDec}')$ with the plaintext space $\{0,1\}$ as described in Fig. 1. We note that, considering a real ciphertext c and a real secret key sk, the correctness of the decryption of Π' is straightforward due to the correctness of Π and Φ.

How to Expand the Plaintext Space of Our Generic Construction. In the above, we only give the construction whose plaintext space is $\{0,1\}$. However, we can expand the plaintext space by using our single-bit construction in a parallel way except for the generation of a proof of an OTSS-NIZK. More concretely, if we encrypt an ℓ-bit plaintext $m = m_1 \| \cdots \| m_\ell$, the procedure is as follows. Firstly, we generate a public key $pk = ((pk_0^i, pk_1^i)_{i\in[\ell]}, crs)$ and a secret key $sk = (\alpha_i, sk_{\alpha_i}^i)_{i\in[\ell]}$, where $\alpha_1, \cdots, \alpha_\ell \leftarrow \{0,1\}$, $(pk_v^i, sk_v^i) \leftarrow \mathsf{KG}(1^\lambda)$ for all $(i,v) \in [\ell] \times \{0,1\}$, and crs denotes a CRS of an OTSS-NIZK. Next, we compute a ciphertext $c = ((c_0^i)_{i\in[\ell]}, (c_1^i)_{i\in[\ell]}, \pi)$, where $c_v^i \leftarrow \mathsf{Enc}(pk_v^i, m_i)$ for all $(i,v) \in [\ell] \times \{0,1\}$ and π is a proof proving that, for each $i \in [\ell]$, the ciphertexts c_0^i and c_1^i encrypt the same plaintext $m_i \in \{0,1\}$. Similarly, for the other procedures, we execute one-bit version algorithms in a parallel way for all $i \in [\ell]$ except for the procedure of the OTSS-NIZK. See the full version of the paper for the details.

4.2 Security Proof

In this section, we show the following theorem.

Theorem 2. *If Π is an IND-CPA secure PKE scheme and Φ is an OTSS-NIZK, then Π' is RNC-CCA secure.*

Before describing the formal proof, we highlight the flow of the proof. We change $\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi',\mathcal{A}}(\lambda)$ to $\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi',\mathcal{A}}(\lambda)$ step by step, where $\mathcal{A}$ is an adversary that attacks the RNC-CCA security of Π'. Although the main part of our proof is similar to that of the original double encryption paradigm [23,26], we have the following three remarkable changes.

First, toward transforming the challenge ciphertext to a fake ciphertext, we make the challenge ciphertext component $c^*_{1\oplus\alpha}$ encrypts $1 \oplus m^*$. Second, in order to eliminate the information of the bit α from the decryption oracle, when answering a decryption query $c = (c_0, c_1, \pi)$ made by $\mathcal{A}$, we use the components (pk_0, sk_0, c_0) corresponding to the position 0 instead of the components $(pk_\alpha, sk_\alpha, c_\alpha)$ corresponding to the position α. Third, we use $\alpha \oplus m^*$ instead of α in order to make the challenge ciphertext c^* and the secret key sk be independent of the challenge plaintext m^*. Due to these changes, the challenge ciphertext c^* and the real secret key sk are respectively switched to the fake ciphertext $\widetilde{c}$ and the fake secret key $\widetilde{sk}$. The proof is as follows.

Proof of Theorem 2. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ be any PPT adversary that attacks the RNC-CCA security of Π'. We introduce the following experiments $\{\mathsf{Exp}_i\}_{i=0}^{5}$.

Exp_0 : Exp_0 is the same as $\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi',\mathcal{A}}(\lambda)$. The detailed description is as follows.

1. First, Exp_0 samples $\alpha \leftarrow \{0,1\}$ and computes $(pk_0, sk_0) \leftarrow \mathsf{KG}(1^\lambda)$, $(pk_1, sk_1) \leftarrow \mathsf{KG}(1^\lambda)$, and $crs \leftarrow \mathsf{CRSGen}(1^\lambda)$. Next, Exp_0 sets $pk := (pk_0, pk_1, crs)$ and $sk := (\alpha, sk_\alpha)$ and runs $\mathcal{A}_1(pk)$. When $\mathcal{A}_1$ makes a decryption query $c = (c_0, c_1, \pi)$, Exp_0 checks whether $\mathsf{Verify}(crs, (pk_0, pk_1, c_0, c_1), \pi) = 1$ holds. If this holds, Exp_0 computes $m \leftarrow \mathsf{Dec}(pk_\alpha, sk_\alpha, c_\alpha)$, and returns m to $\mathcal{A}_1$. Otherwise, Exp_0 returns $\perp$ to $\mathcal{A}_1$.
2. When $\mathcal{A}_1$ outputs (m^*, st_1) and terminates, Exp_0 computes the challenge ciphertext c^* as follows. First, Exp_0 samples $(r^*_0, r^*_1) \leftarrow \mathcal{R}^2_\Pi$, computes $c^*_0 \leftarrow \mathsf{Enc}(pk_0, m^*; r^*_0)$, $c^*_1 \leftarrow \mathsf{Enc}(pk_1, m^*; r^*_1)$, and $\pi^* \leftarrow \mathsf{Prove}(crs, (pk_0, pk_1, c^*_0, c^*_1), (m^*, r^*_0, r^*_1))$. Next, Exp_0 sets $c^* = (c^*_0, c^*_1, \pi^*)$, and runs $\mathcal{A}_2(c^*, \mathsf{st}_1)$. When $\mathcal{A}_2$ makes a decryption query c, Exp_0 answers in the same way as above.
3. When $\mathcal{A}_2$ outputs state information st_2 and terminates, Exp_0 runs $\mathcal{A}_3(sk, \mathsf{st}_2)$. When $\mathcal{A}_3$ outputs a bit b' and terminates, Exp_0 outputs b'.

Exp_1 : Exp_1 is identical to Exp_0 except for the following change. The common reference string crs is generated by executing $(crs, tk) \leftarrow \mathsf{SimCRS}(1^\lambda)$, and Exp_1 generates a simulated proof $\pi^* \leftarrow \mathsf{SimPrv}(tk, (pk_0, pk_1, c^*_0, c^*_1))$ when computing the challenge ciphertext c^*.

Exp_2 : Exp_2 is identical to Exp_1 except that when computing the challenge ciphertext c^*, Exp_2 computes $c^*_{1\oplus\alpha} \leftarrow \mathsf{Enc}(pk_{1\oplus\alpha}, 1 \oplus m^*)$ instead of $c^*_{1\oplus\alpha} \leftarrow \mathsf{Enc}(pk_{1\oplus\alpha}, m^*)$.

Exp_3 : Exp_3 is identical to Exp_2 except that when responding to a decryption query $c = (c_0, c_1, \pi)$, Exp_3 answers $m \leftarrow \mathsf{Dec}(pk_0, sk_0, c_0)$ instead of $m \leftarrow \mathsf{Dec}(pk_\alpha, sk_\alpha, c_\alpha)$, if $\mathsf{Verify}(crs, (pk_0, pk_1, c_0, c_1), \pi) = 1$ holds. Note that the decryption procedure in Exp_3 is exactly the same as FDec'.

Exp_4 : Exp_4 is identical to Exp_3 except that $\alpha \oplus m^*$ is used instead of α. That is, when computing the challenge ciphertext c^*, Exp_4 computes c^*_0 and c^*_1 by $c^*_{\alpha\oplus m^*} \leftarrow \mathsf{Enc}(pk_{\alpha\oplus m^*}, m^*)$ and $c^*_{\alpha\oplus(1\oplus m^*)} \leftarrow \mathsf{Enc}(pk_{\alpha\oplus(1\oplus m^*)}, 1 \oplus m^*)$. Moreover, Exp_4 gives the secret key $sk = (\alpha \oplus m^*, sk_{\alpha\oplus m^*})$ to $\mathcal{A}_3$ instead of $sk = (\alpha, sk_\alpha)$.

Exp_5 : Exp_5 is exactly the same as $\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi',\mathcal{A}}(\lambda)$.

We let $p_i := \Pr[\mathsf{Exp}_i(\lambda) = 1]$ for all $i \in [0, 5]$. Then, we have $\mathsf{Adv}^{\mathsf{rnc\text{-}cca}}_{\Pi',\mathcal{A}}(\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi',\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi',\mathcal{A}}(\lambda) = 1]| = |p_0 - p_5| \leq \sum_{i=0}^{4} |p_i - p_{i+1}|$. It remains to show how each $|p_i - p_{i+1}|$ is upper-bounded. Due to the space limitation, we will show them formally in the full version of the paper. There, we will show that there exist an adversary $\mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$ against the ZK property of Φ such that $|p_0 - p_1| = \mathsf{Adv}^{\mathsf{zk}}_{\Phi,\mathcal{E}}(\lambda)$, an adversary $\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2)$ against the IND-CPA security of Π such that $|p_1 - p_2| = \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi,\mathcal{F}}(\lambda)$, and an adversary $\mathcal{G} = (\mathcal{G}_1, \mathcal{G}_2)$ against the OT-SS of Φ such that $|p_2 - p_3| \leq \mathsf{Adv}^{\mathsf{ot\text{-}ss}}_{\Phi,\mathcal{G}}(\lambda)$. Then, we will show that $|p_3 - p_4| = 0$ holds. The main reason why this is true, is because since α is chosen uniformly at random, $\alpha \oplus m^*$ is also distributed uniformly at random. Finally, we will show that $|p_4 - p_5| = 0$ holds, by showing that Exp_4 and Exp_5 are identical.

Putting everything together, we obtain $\mathsf{Adv}^{\mathsf{rnc\text{-}cca}}_{\Pi',\mathcal{A}}(\lambda) = |p_0 - p_5| \leq \sum_{i=0}^{4} |p_i - p_{i+1}| \leq \mathsf{Adv}^{\mathsf{zk}}_{\Phi,\mathcal{E}}(\lambda) + \mathsf{Adv}^{\mathsf{ind\text{-}cpa}}_{\Pi,\mathcal{F}}(\lambda) + \mathsf{Adv}^{\mathsf{ot\text{-}ss}}_{\Phi,\mathcal{G}}(\lambda)$. Since Π is IND-CPA secure and Φ is an OTSS-NIZK, for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{rnc\text{-}cca}}_{\Pi',\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$ holds. Therefore, Π' satisfies RNC-CCA security. □ (**Theorem** 2)

5 Our DDH-Based Concrete Construction of RNC-CCA Secure RNCE

In this section, we show our concrete construction of RNC-CCA secure RNCE with a polynomial-sized plaintext space, based on the DDH assumption and a collision-resistant hash function. First, in Sect. 5.1, we describe our DDH-based concrete construction. Then, in Sect. 5.2, we give a proof of RNC-CCA security for our DDH-based construction.

5.1 The Description of Our Concrete Construction

Here, we give the formal description of our DDH-based construction of RNC-CCA secure RNCE with a polynomial-sized plaintext space. One can see that

$\mathsf{KG}(1^\lambda)$:	$\mathsf{Enc}(pk, m)$:	$\mathsf{Dec}(pk, sk, c)$:
$g_1 \leftarrow \mathbb{G};\ w \leftarrow \mathbb{Z}_p^*;\ g_2 := g_1^w$ $x_1, x_2, y_1, y_2, z_1, z_2 \leftarrow \mathbb{Z}_p$ $k := g_1^{x_1} g_2^{x_2}$ $s := g_1^{y_1} g_2^{y_2}$ $t := g_1^{z_1} g_2^{z_2}$ $hk \leftarrow \mathsf{HKG}(1^\lambda)$ $pk := (g_1, g_2, k, s, t, hk)$ $sk := (x_1, x_2, y_1, y_2, z_1, z_2)$ Return (pk, sk)	$r \leftarrow \mathbb{Z}_p$ $u_1 := g_1^r$ $u_2 := g_2^r$ $e := k^r \cdot g_1^m$ $\mu \leftarrow \mathsf{Hash}(hk, u_1 \| u_2 \| e)$ $v := s^r t^{r\mu}$ Return $c := (u_1, u_2, e, v)$	Parse $c = (u_1, u_2, e, v)$ $\mu \leftarrow \mathsf{Hash}(hk, u_1 \| u_2 \| e)$ If $u_1^{y_1+z_1\mu} u_2^{y_2+z_2\mu} = v$ then $m := \log_{g_1}(\frac{e}{(u_1^{x_1} u_2^{x_2})})$ Return m else Return $\perp$
$\mathsf{FKG}(1^\lambda)$:	$\mathsf{Fake}(pk, td)$:	$\mathsf{FDec}(pk, td, c)$:
$g_1 \leftarrow \mathbb{G};\ \ w \leftarrow \mathbb{Z}_p^*$ $g_2 := g_1^w$ $x_1, x_2, y_1, y_2, z_1, z_2 \leftarrow \mathbb{Z}_p$ $k := g_1^{x_1} g_2^{x_2}$ $s := g_1^{y_1} g_2^{y_2}$ $t := g_1^{z_1} g_2^{z_2}$ $hk \leftarrow \mathsf{HKG}(1^\lambda)$ $pk := (g_1, g_2, k, s, t, hk)$ $\widetilde{r} \leftarrow \mathbb{Z}_p$ $td := (\widetilde{r}, w, x_1, x_2,$ $y_1, y_2, z_1, z_2)$ Return (pk, td)	$\widetilde{u_1} := g_1^{\widetilde{r}}$ $\widetilde{u_2} := g_1 g_2^{\widetilde{r}}$ $\widetilde{e} := g_1^{x_2} k^{\widetilde{r}}$ $\widetilde{\mu} \leftarrow \mathsf{Hash}(hk, \widetilde{u_1} \| \widetilde{u_2} \| \widetilde{e})$ $\widetilde{v} := g_1^{y_2+z_2\mu} s^{\widetilde{r}} t^{\widetilde{r}\widetilde{\mu}}$ Return $\widetilde{c} := (\widetilde{u_1}, \widetilde{u_2}, \widetilde{e}, \widetilde{v})$ $\mathsf{Open}(pk, td, \widetilde{c}, m)$: $x_1' := x_1 + mw \pmod p$ $x_2' := x_2 - m \pmod p$ $\widetilde{sk} := (x_1', x_2', y_1, y_2, z_1, z_2)$ Return $\widetilde{sk}$	Parse $c = (u_1, u_2, e, v)$ $\mu \leftarrow \mathsf{Hash}(hk, u_1 \| u_2 \| e)$ $\widetilde{u_1} := g_1^{\widetilde{r}}$ $\widetilde{u_2} := g_1 g_2^{\widetilde{r}}$ $\widetilde{e} := g_1^{x_2} k^{\widetilde{r}}$ $\widetilde{\mu} \leftarrow \mathsf{Hash}(hk, \widetilde{u_1} \| \widetilde{u_2} \| \widetilde{e})$ If $(\mu \neq \widetilde{\mu}) \wedge (u_1^w = u_2)$ $\wedge (u_1^{y_1+z_1\mu} u_2^{y_2+z_2\mu} = v)$ then $m := \log_{g_1}(\frac{e}{(u_1^{x_1} u_2^{x_2})})$ Return m else Return $\perp$

Fig. 2. Our DDH-based construction of RNC-CCA secure RNCE Π_{ddh}.

our scheme is a variant of the Cramer-Shoup encryption scheme [7]. The only difference is that we encode a plaintext m by the group element g_1^m, where g_1 is a generator of the underlying group. This encoding is essential for the opening algorithm Open of our proposed scheme. The plaintext space of our scheme needs to be of polynomial-size since we need to compute the discrete logarithm of g_1^m for the decryption procedure.

Formally, we let $\Lambda = (\mathsf{HKG}, \mathsf{Hash})$ be a hash function. Let $\mathbb{G}$ be a multiplicative cyclic group of prime order $p = \Theta(2^\lambda)$. We naturally encode an element in $\{0,1\}^\lambda$ as one in $\mathbb{Z}_p$. Then, we construct our RNCE scheme $\Pi_{\mathsf{ddh}} = (\mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{FKG}, \mathsf{Fake}, \mathsf{Open}, \mathsf{FDec})$ as described in Fig. 2. We note that the correctness of the decryption of Π_{ddh} is straightforward due to the correctness of the original Cramer-Shoup encryption scheme.

5.2 Security Proof

In this section, we show the following theorem.

Theorem 3. *If the "+1"-DDH assumption holds in $\mathbb{G}$, and $\Lambda = (\mathsf{HKG}, \mathsf{Hash})$ is a collision-resistant hash function, then Π_{ddh} is RNC-CCA secure.*

Since the "+1"-DDH assumption is implied by the ordinary DDH assumption, Theorem 3 implies that our construction Π_{ddh} is indeed RNC-CCA secure under the ordinary DDH assumption.

Before describing the proof, we highlight the flow of the proof. We change $\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda)$ to $\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda)$ step by step, where $\mathcal{A}$ is an adversary that attacks the RNC-CCA security of Π_{ddh}. Although the main part of our proof is similar to that of the original HPS technique [7,8], we have the following two remarkable changes in order to change the challenge ciphertext c^* to a fake ciphertext $\widetilde{c}$.

First, toward transforming the challenge ciphertext to a fake ciphertext, we change the challenge ciphertext component $u_2^* = g_2^{r^*}$ to $u_2^* = g_1 g_2^{r^*}$. Second, we change the real secret key component (x_1, x_2) to the fake secret key component (x_1', x_2') computed by Open described in Fig. 2. Due to these changes, the challenge ciphertext component e^* is changed to the fake ciphertext component $\widetilde{e}$ and the real secret key sk is changed to the fake secret key $\widetilde{sk}$. The proof is as follows.

Proof of Theorem 3. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$ be any PPT adversary that attacks the RNC-CCA security of Π_{ddh} and makes $Q_{dec} > 0$ decryption queries. We introduce the following experiments $\{\mathsf{Exp}_i\}_{i=0}^{7}$.

Exp_0 : Exp_0 is the same as $\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda)$. The detailed description is as follows.

1. First, Exp_0 samples $g_1 \leftarrow \mathbb{G}$ and $w \leftarrow \mathbb{Z}_p^*$ and sets $g_2 := g_1^w$. Next, Exp_0 samples $x_1, x_2, y_1, y_2, z_1, z_2 \leftarrow \mathbb{Z}_p$ and sets $k := g_1^{x_1} g_2^{x_2}$, $s := g_1^{y_1} g_2^{y_2}$, and $t := g_1^{z_1} g_2^{z_2}$. Then, it samples $hk \leftarrow \mathsf{HKG}(1^\lambda)$, sets $pk := (g_1, g_2, k, s, t, hk)$ and $sk := (x_1, x_2, y_1, y_2, z_1, z_2)$, and runs $\mathcal{A}_1(pk)$. When $\mathcal{A}_1$ makes a decryption query $c = (u_1, u_2, e, v)$, Exp_0 computes $\mu \leftarrow \mathsf{Hash}(hk, u_1 \| u_2 \| e)$ and checks whether $u_1^{y_1 + z_1 \mu} u_2^{y_2 + z_2 \mu} = v$ holds. If this holds, Exp_0 returns $m = \log_{g_1}(e \cdot (u_1^{x_1} u_2^{x_2})^{-1})$ to $\mathcal{A}_1$. Otherwise, Exp_0 returns $\perp$ to $\mathcal{A}_1$.
2. When $\mathcal{A}_1$ outputs (m^*, st_1) and terminates, Exp_0 computes the challenge ciphertext c^* as follows. First, Exp_0 samples $r^* \leftarrow \mathbb{Z}_p$ and sets $u_1^* := g_1^{r^*}$, $u_2^* := g_2^{r^*}$, and $e^* := k^{r^*} \cdot g_1^{m^*}$. Next, Exp_0 computes $\mu^* \leftarrow \mathsf{Hash}(hk, u_1^* \| u_2^* \| e^*)$, sets $v^* := s^{r^*} t^{r^* \mu^*}$ and $c^* := (u_1^*, u_2^*, e^*, v^*)$, and runs $\mathcal{A}_2(c^*, \mathsf{st}_1)$. When $\mathcal{A}_2$ makes a decryption query $c = (u_1, u_2, e, v)$, Exp_0 answers the query from $\mathcal{A}_2$ in the same way as above.
3. When $\mathcal{A}_2$ outputs state information st_2 and terminates, Exp_0 runs $\mathcal{A}_3(sk, \mathsf{st}_2)$. When $\mathcal{A}_3$ outputs b' and terminates, Exp_0 outputs b'.

Exp_1 : Exp_1 is identical to Exp_0 except for the following change. When computing the challenge ciphertext $c^* = (u_1^*, u_2^*, e^*, v^*)$, Exp_1 computes e^* and v^* by $e^* := (u_1^*)^{x_1} (u_2^*)^{x_2} \cdot g_1^{m^*}$ and $v^* := (u_1^*)^{y_1} (u_2^*)^{y_2} ((u_1^*)^{z_1} (u_2^*)^{z_2})^{\mu^*}$, respectively.

Exp_2 : Exp_2 is identical to Exp_1 except that when computing the challenge ciphertext $c^* = (u_1^*, u_2^*, e^*, v^*)$, Exp_2 computes u_2^* by $u_2^* := g_1^{w r^* + 1}$.

Exp_3 : Exp_3 is identical to Exp_2 except that when responding to a decryption query $c = (u_1, u_2, e, v)$ made by $\mathcal{A}_2$, Exp_3 answers $\perp$ if $\mathsf{Hash}(hk, u_1^* \| u_2^* \| e^*) = \mathsf{Hash}(hk, u_1 \| u_2 \| e)$ holds.

Exp_4 : Exp_4 is identical to Exp_3 except that when responding to a decryption query $c = (u_1, u_2, e, v)$ made by $\mathcal{A}_1$ or $\mathcal{A}_2$, Exp_4 answers $\perp$ if $u_1^w \neq u_2$ holds.

Exp_5 : Exp_5 is identical to Exp_4 except that $x'_1 := x_1 + wm^* \pmod p$ and $x'_2 := x_2 - m^* \pmod p$ are used instead of x_1 and x_2, respectively. That is, when computing the challenge ciphertext $c^* := (u_1^*, u_2^*, e^*, v^*)$, Exp_5 computes e^* by $e^* := (u_1^*)^{x'_1}(u_2^*)^{x'_2} \cdot g_1^{m^*}$ instead of $(u_1^*)^{x_1}(u_2^*)^{x_2} \cdot g_1^{m^*}$. Note that $e^* = (u_1^*)^{x'_1}(u_2^*)^{x'_2} \cdot g_1^{m^*} = (g_1^{r^*})^{(x_1+wm^*)}(g_1^{(wr^*+1)})^{(x_2-m^*)} \cdot g_1^{m^*} = g_1^{x_2}(g_1^{x_1}g_2^{x_2})^{r^*}$ holds. Furthermore, Exp_5 gives the secret key $sk' := (x'_1, x'_2, y_1, y_2, z_1, z_2)$ to $\mathcal{A}_3$ instead of $sk := (x_1, x_2, y_1, y_2, z_1, z_2)$.
Since e^* in Exp_5 is independent of the challenge message m^*, without loss of generality we generate it before $\mathcal{A}_1$ is run.

Exp_6 : Exp_6 is identical to Exp_5 except that when responding to a decryption query $c = (u_1, u_2, e, v)$ made by $\mathcal{A}_1$, Exp_6 answers $\perp$ if $\mathsf{Hash}(hk, u_1^* \| u_2^* \| e^*) = \mathsf{Hash}(hk, u_1 \| u_2 \| e)$ holds. Note that the procedure of the decryption oracle in Exp_6 is exactly the same as that of $\mathsf{FDec}(pk, td, c)$.

Exp_7 : Exp_7 is exactly the same as $\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda)$.

We let $p_i := \Pr[\mathsf{Exp}_i(\lambda) = 1]$ for all $i \in [0, 7]$. Then, we have $\mathsf{Adv}^{\mathsf{rnc\text{-}cca}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda) = |\Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}real}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda) = 1] - \Pr[\mathsf{Exp}^{\mathsf{rnc\text{-}sim}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda) = 1]| = |p_0 - p_7| \leq \sum_{i=0}^{6} |p_i - p_{i+1}|$. It remains to show how each $|p_i - p_{i+1}|$ is upper-bounded. Due to the space limitation, we will show them formally in the full version of the paper. There, we will show that $|p_0 - p_1| = 0$ holds since the difference between Exp_0 and Exp_1 is only conceptual. Then, we will show that there exist a PPT adversary $\mathcal{E}$ against the "+1"-DDH assumption in $\mathbb{G}$ such that $|p_1 - p_2| = \mathsf{Adv}^{\text{+1-ddh}}_{\mathbb{G},\mathcal{E}}(\lambda)$, and a PPT adversary $\mathcal{F}$ against the collision-resistance of Λ such that $|p_2 - p_3| \leq \mathsf{Adv}^{\mathsf{cr}}_{\Lambda,\mathcal{F}}(\lambda)$. Next, we will show that $|p_3 - p_4| \leq \frac{Q_{dec}}{p}$ holds by showing that the probability that each of $\mathcal{A}$'s valid queries is rejected in Exp_5 but not in Exp_4, is at most $\frac{1}{p}$. Then, we will show that $|p_4 - p_5| = 0$ holds since (x_1, x_2) and (x'_1, x'_2) are information-theoretically indistinguishable from $\mathcal{A}$. Next, we will show that there exists a PPT adversary $\mathcal{G}$ against the collision-resistance of Λ such that $|p_5 - p_6| \leq \mathsf{Adv}^{\mathsf{cr}}_{\Lambda,\mathcal{G}}(\lambda) + \frac{Q_{dec}}{p}$. Finally, we will show that $|p_6 - p_7| = 0$ holds, by showing that Exp_6 and Exp_7 are identical.

Putting everything together, we obtain $\mathsf{Adv}^{\mathsf{rnc\text{-}cca}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda) = |p_0 - p_7| \leq \sum_{i=0}^{6} |p_i - p_{i+1}| \leq \mathsf{Adv}^{\text{+1-ddh}}_{\mathbb{G},\mathcal{E}}(\lambda) + \mathsf{Adv}^{\mathsf{cr}}_{\Lambda,\mathcal{F}}(\lambda) + \mathsf{Adv}^{\mathsf{cr}}_{\Lambda,\mathcal{G}}(\lambda) + \frac{2Q_{dec}}{p}$. Since the "+1"-DDH assumption holds in $\mathbb{G}$, Λ is a collision-resistant hash function, Q_{dec} is a polynomial of λ, and $p = \Theta(2^\lambda)$, for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{rnc\text{-}cca}}_{\Pi_{\mathsf{ddh}},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$ holds. Therefore, Π_{ddh} satisfies RNC-CCA security. □ (**Theorem 3**)

References

1. Bader, C., Hofheinz, D., Jager, T., Kiltz, E., Li, Y.: Tightly-secure authenticated key exchange. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9014, pp. 629–658. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46494-6_26
2. Barak, B., et al.: On the (im)possibility of obfuscating programs. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 1–18. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44647-8_1

3. Bellare, M., Dowsley, R., Waters, B., Yilek, S.: Standard security does not imply security against selective-opening. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 645–662. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_38
4. Bellare, M., Hofheinz, D., Yilek, S.: Possibility and impossibility results for encryption and commitment secure under selective opening. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 1–35. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01001-9_1
5. Bellare, M., Yilek, S.: Encryption schemes secure under selective opening attack. IACR Cryptology ePrint Archive, 2009:101
6. Canetti, R., Halevi, S., Katz, J.: Adaptively-secure, non-interactive public-key encryption. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 150–168. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-30576-7_9
7. Cramer, R., Shoup, V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 13–25. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0055717
8. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 45–64. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-46035-7_4
9. Damgård, I., Nielsen, J.B.: Improved non-committing encryption schemes based on a general complexity assumption. In: Bellare, M. (ed.) CRYPTO 2000. LNCS, vol. 1880, pp. 432–450. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_27
10. Dolev, D., Dwork, C., Naor, M.: Non-malleable cryptography (extended abstract). In: STOC 1991, pp. 542–552 (1991)
11. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. IACR Cryptology ePrint Archive, 2013:451
12. Goldwasser, S., Micali, S.: Probabilistic encryption. J. Comput. Syst. Sci. **28**(2), 270–299 (1984)
13. Groth, J., Ostrovsky, R., Sahai, A.: Perfect non-interactive zero knowledge for NP. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 339–358. Springer, Heidelberg (2006). https://doi.org/10.1007/11761679_21
14. Groth, J., Sahai, A.: Efficient non-interactive proof systems for bilinear groups. In: Smart, N. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 415–432. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78967-3_24
15. Hazay, C., Patra, A., Warinschi, B.: Selective opening security for receivers. In: Iwata, T., Cheon, J.H. (eds.) ASIACRYPT 2015. LNCS, vol. 9452, pp. 443–469. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48797-6_19
16. Jia, D., Lu, X., Li, B.: Receiver selective opening security from indistinguishability obfuscation. In: Dunkelman, O., Sanadhya, S.K. (eds.) INDOCRYPT 2016. LNCS, vol. 10095, pp. 393–410. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49890-4_22
17. Jia, D., Lu, X., Li, B.: Constructions secure against receiver selective opening and chosen ciphertext attacks. In: Handschuh, H. (ed.) CT-RSA 2017. LNCS, vol. 10159, pp. 417–431. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-52153-4_24

18. Libert, B., Sakzad, A., Stehlé, D., Steinfeld, R.: All-but-many lossy trapdoor functions and selective opening chosen-ciphertext security from LWE. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017. LNCS, vol. 10403, pp. 332–364. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63697-9_12
19. Lindell, Y.: A simpler construction of CCA2-secure public-key encryption under general assumptions. In: Biham, E. (ed.) EUROCRYPT 2003. LNCS, vol. 2656, pp. 241–254. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-39200-9_15
20. Liu, S., Paterson, K.G.: Simulation-based selective opening CCA security for PKE from key encapsulation mechanisms. In: Katz, J. (ed.) PKC 2015. LNCS, vol. 9020, pp. 3–26. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46447-2_1
21. Liu, S., Zhang, F., Chen, K.: Selective opening chosen ciphertext security directly from the DDH assumption. In: Xu, L., Bertino, E., Mu, Y. (eds.) NSS 2012. LNCS, vol. 7645, pp. 100–112. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34601-9_8
22. Lyu, L., Liu, S., Han, S., Gu, D.: Tightly SIM-SO-CCA secure public key encryption from standard assumptions. In: Abdalla, M., Dahab, R. (eds.) PKC 2018. LNCS, vol. 10769, pp. 62–92. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76578-5_3
23. Naor, M., Yung, M.: Public-key cryptosystems provably secure against chosen ciphertext attacks. In: STOC 1990, pp. 427–437 (1990)
24. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. In: STOC 2008, pp. 187–196 (2008)
25. Rosen, A., Segev, G.: Chosen-ciphertext security via correlated products. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 419–436. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00457-5_25
26. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: FOCS 1999, pp. 543–553 (1999)
27. Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: STOC 2014, pp. 475–484 (2014)