Adversary-dependent Lossy Trapdoor Function from Hardness of Factoring Semi-smooth RSA Subgroup Moduli [⋆]

Takashi Yamakawa[1,2], Shota Yamada[2],
Goichiro Hanaoka[2], and Noboru Kunihiro[1]

The University of Tokyo, Chiba, Japan
yamakawa@it.k.u-tokyo.ac.jp, kunihiro@k.u-tokyo.ac.jp
National Institute of Advanced Industrial Science and Technology (AIST),
Tokyo, Japan
{yamada-shota, hanaoka-goichiro}@aist.go.jp

Abstract. Lossy trapdoor functions (LTDFs), proposed by Peikert and Waters (STOC'08), are known to have a number of applications in cryptography. They have been constructed based on various assumptions, which include the quadratic residuosity (QR) and decisional composite residuosity (DCR) assumptions, which are factoring-based *decision* assumptions. However, there is no known construction of an LTDF based on the factoring assumption or other factoring-related search assumptions. In this paper, we first define a notion of *adversary-dependent lossy trapdoor functions* (ad-LTDFs) that is a weaker variant of LTDFs. Then we construct an ad-LTDF based on the hardness of factorizing RSA moduli of a special form called semi-smooth RSA subgroup (SS) moduli proposed by Groth (TCC'05). Moreover, we show that ad-LTDFs can replace LTDFs in many applications. Especially, we obtain the first factoring-based deterministic encryption scheme that satisfies the security notion defined by Boldyreva et al. (CRYPTO'08) without relying on a decision assumption. Besides direct applications of ad-LTDFs, by a similar technique, we construct a chosen ciphertext secure public key encryption scheme whose ciphertext overhead is the shortest among existing schemes based on the factoring assumption w.r.t. SS moduli.

1 Introduction

1.1 Background

In modern cryptography, constructing provably secure cryptographic primitives is an important research topic. In this line of researches, Peikert and Waters [27] proposed *lossy trapdoor functions* (LTDFs) and constructed a number of cryptographic primitives such as a collision resistant hash function, a chosen plaintext (CPA) and chosen ciphertext (CCA) secure public key encryption (PKE) schemes and an oblivious transfer scheme based on LTDFs. Following the work,

[⋆] The first author is supported by a JSPS Fellowship for Young Scientists.

it is also shown that LTDFs can be used for constructing a deterministic encryption (DE) scheme [5] and a selective opening attack (SOA) secure PKE scheme [3]. As seen above, LTDFs have many applications, and therefore it is important to research concrete constructions of LTDFs.

As concrete constructions of LTDFs, Peikert and Waters [27] constructed schemes based on the decisional Diffie-Hellman (DDH) and learning with errors (LWE) assumptions. After that, many constructions of LTDFs have been proposed thus far. Among them, LTDFs related to the factoring are based on the quadratic residuosity (QR) [11], decisional composite residuosity (DCR) [11], Φ-hiding [20], or general class of subgroup decision assumptions [36], all of which are decision assumptions. On the other hand, there is no known construction of an LTDF based on the factoring assumption or a factoring-related search assumption. In general, search assumptions are rather weaker than decision assumptions. Thus it is important to research the possibility of constructing LTDFs based on a search assumption.

1.2 Our Result

In this paper, though we do not construct LTDFs based on the factoring assumption, we construct an *adversary dependent lossy trapdoor function* (ad-LTDF), which is a new notion we introduce, based on the factoring assumption w.r.t. semi-smooth RSA subgroup (SS) moduli, which are RSA moduli of a special form [13]. Then we show that ad-LTDFs can replace LTDFs in many applications. As a result, we immediately obtain factoring-based cryptographic primitives including a hash function, PKE scheme and DE scheme. Besides direct applications of ad-LTDFs, by using similar technique, we construct CCA secure PKE scheme with compact ciphertext based on the factoring assumption w.r.t. SS moduli. More details are given in the following.

Adversary-dependent lossy trapdoor function. We first reconsider the definition of LTDFs, and introduce a notion of an ad-LTDF, which is a weaker variant of an LTDF. Intuitively, an LTDF is a computationally indistinguishable pair of an injective and lossy functions. Here, the description of lossy functions should be fixed by the scheme. On the other hand, for ad-LTDFs, we allow a description of lossy function to depend on an adversary. That is, we only require that for any efficient adversary $\mathcal{A}$ there exists a lossy function that $\mathcal{A}$ cannot distinguish from an injective function. We observe that this significant relaxation does not harm the security of many LTDF-based cryptographic constructions. This is because in many LTDF-based schemes, lossy functions are used only in security proofs and they do not appear in the real scheme. This means that even if lossy functions depend on an adversary, we can still prove the security of the scheme. By this observation, we can see that ad-LTDFs can replace LTDFs in many applications.

Moreover, we construct an ad-LTDF based on the factoring assumption w.r.t. SS moduli, which is introduced by Groth [13]. As a result, we can instantiate many LTDF-based constructions based on the factoring assumption w.r.t. SS moduli. The intuition of the construction of the ad-LTDF is given in Sec. 1.3.

Applications of ad-LTDFs. As stated above, ad-LTDFs can replace LTDFs in many applications, and we give a construction of an ad-LTDF under the factoring assumption w.r.t. SS moduli. Thus we immediately obtain new factoring-based constructions of many cryptographic primitives such as a collision resistant hash function, CPA secure PKE scheme and a DE scheme. Among them, the DE scheme obtained by this way is the first factoring-based scheme that satisfies the PRIV security for block-sources, which is defined in [5], without relying on any decision assumption.

Table 1. Comparison among CCA secure PKE schemes based on the factoring assumption: ℓ_N is the bit-length of an underlying composite number N, ℓ_{MAC} denotes the bit-length of a message authentication code, Factoring SS denotes the factoring assumption w.r.t. SS moduli, and we assume that an exponentiation with an exponent of length ℓ can be computed by 1.5ℓ multiplications.

Schemes	Ciphertext overhead (bit)	Public key size (bit)	Computational cost for encryption (mult)	Computational cost for decryption (mult)	Assumption
HK09 [18]	$2\ell_N$	$3\ell_N$	$3\ell_N + 3.5\lambda$	$1.5\ell_N + 10.5\lambda$	Factoring
MLLJ11 [25]	$2\ell_N$	$3\ell_N$	18.5λ	18λ	Factoring SS
Ours	$\ell_N + \ell_{MAC}$	$O(\lambda^2\ell_N/\log\lambda)$	$O(\lambda\ell_N^2/\log\lambda)$	$O(\lambda\ell_N^2/\log\lambda)$	Factoring SS

CCA secure PKE with short ciphertext. Besides direct applications of ad-LTDFs, we construct a CCA secure PKE scheme whose ciphertext overhead is the shortest among schemes based on the factoring assumption w.r.t. SS moduli. Table 1 shows the efficiency of CCA secure PKE schemes based on the factoring assumption. Among existing schemes, the scheme proposed by Hofheinz and Kiltz [18] is one of the best in regard to the ciphertext overhead, which consists of 2 elements of $\mathbb{Z}_N^*$. Mei et al. [25] improved the efficiency of the Hofheinz-Kiltz scheme [18] in regard to encryption and decryption costs by using SS moduli. However, they did not improve the ciphertext overhead. In contrast, the ciphertext overhead of our scheme consists of only 1 element of $\mathbb{Z}_N^*$ and a message authentication code (MAC), whose bit-length can be much smaller than that of N. By giving a concrete parameter, the ciphertext overhead of our scheme is 1360-bit for 80-bit security whereas that of [18] is 2048-bit. On the other hand, the public key size of our scheme is much larger than that of [18], and an encryption and decryption are much less efficient than those in [18]. We note that the reduction from the CCA security of our scheme to the factoring assumption w.r.t. SS moduli is quite loose, but all known CCA secure PKE scheme based on the factoring assumption (including [18, 25]) also require loose reductions because they require Blum-Blum-Shub pseudo-random number generator [4].

We note that there is a strong negative result for a CCA secure PKE scheme whose ciphertext overhead is less than 2 group elements in a prime order setting

[14]. Even in a composite order setting, there are only a few CCA secure PKE schemes whose ciphetext overhead is less than 2 group elements, all of which rely on a subgroup decision assumption [16, 21, 17] or an interactive assumption [19] stronger than the factoring assumption. Ours is the first scheme to overcome this bound based solely on the factoring assumption (though our assumption is the factoring assumption w.r.t. SS moduli, which may not be considered standard).

1.3 Our Technique

Difficulty of constructiing LTDFs based on a search assumption. Before explaining our technique, we first explain why it is difficult to construct LTDFs based on a search assumption. Recall that an LTDF is a computationally indistinguishable pair of injective and lossy functions. Apparently, the definition of LTDFs itself requires the hardness of a decision problem. Thus for constructing LTDFs based on a search assumption, we have to rely on some "search-to-decision" reduction. As a general technique for such a reduction, there is the Goldreich-Levin hardcore theorem [12], which enables us to extract "pseudorandomness" from hardness of any search problem. However, the Goldreich-Levin hardcore bit destroys algebraic structures of original problems. On the other hand, considering existing constructions of LTDFs, algebraic structures of underlying problems are crucial for constructing LTDFs. Thus, for constructing LTDFs based on search assumptions, we have to establish another "search-to-decision" reduction technique that does not hurt underlying algebraic structures. In the context of lattice problems, this has been already done. Namely, it is shown that search-LWE and decision-LWE assumptions are equivalent [33]. Thus LTDFs can be constructed based on the search-LWE assumption. However, there is no known such a reduction in the context of the factoring problem. Namely, we have no reduction from decision assumptions such as QR, DCR, or more general subgroup decision assumptions to the factoring assumption.

New search-to-decision reduction technique. The core of this work is to give a new search-to-decision reduction technique in the context of factoring w.r.t. SS moduli. Namely, we introduce a new decision assumption that we call the adversary-dependent decisional RSA subgroup (ad-DRSA) assumption, and reduce the ad-DRSA assumption to the factoring assumption w.r.t. SS moduli. In the following, we explain the technique in more detail.

We say that a composite number N is an SS modulus if it can be written as $N = PQ = (2pp'+1)(2qq'+1)$, where P and Q are primes with the same length, p and q are "smooth" numbers (i.e., products of distinct small primes) and p' and q' are relatively large primes. Then the group of quadratic residues $\mathbb{QR}_N$ is a cyclic group of order $pqp'q'$, and has many subgroups since pq is smooth. With respect to SS moduli, Groth [13] proposed the decisional RSA subgroup (DRSA) assumption , which claims that any PPT adversary cannot distinguish a random element of G from that of $\mathbb{QR}_N$ where G is the unique subgroup of $\mathbb{QR}_N$ of order $p'q'$.

Our first observation is that if there exists an algorithm that breaks the DRSA assumption, then one can find at least one small prime that divides $\Phi(N)$. This can be seen by the following argument: Assume that all prime factors of pq are of ℓ_B-bit length. (Since pq is smooth, ℓ_B is relatively small. Especially, we set $\ell_B = O(\log \lambda)$.) Recall that the DRSA assumption claims that any PPT algorithm cannot distinguish a random element of G from that of $\mathbb{QR}_N$. This is equivalent to that the distributions of $g^{p_1 \cdots p_M}$ and g are indistinguishable where $g \xleftarrow{\$} \mathbb{QR}_N$ and $p_1, \ldots, p_M$ are the all ℓ_B-bit primes (and thus M is the number of the all ℓ_B-bit primes). If there exists an algorithm $\mathcal{A}$ that breaks the DRSA assumption, then it distinguishes these two distributions. Thus, by the hybrid argument, there exists $j \in [M]$ such that $\mathcal{A}$ distinguishes the distribution of $g^{p_1 \cdots p_{j-1}}$ from $g^{p_1, \cdots p_j}$. By using $\mathcal{A}$, one can find this p_j by the exhaustive search since M is polynomial in the security parameter in our parameter setting. (See Sec. 2.4 for more detail.) For this p_j, we have $p_j | \Phi(N)$ (with overwhelming probability) since otherwise p_j-th power on $\mathbb{QR}_N$ is a permutation on the group and thus distributions of $g^{p_1 \cdots p_{j-1}}$ and $g^{p_1, \cdots p_j}$ are completely identical. The above argument proves that if there exists an algorithm that breaks the DRSA assumption, then one can find at least one small prime that divides $\Phi(N)$. However, this fact states nothing about the reduction from the DRSA assumption to the factoring assumption since even if one can find one small prime p that divides $\Phi(N)$, we do not know how to factorize N.

Here, we relax the DRSA assumption to define the *adversary-dependent decisional RSA subgroup* (ad-DRSA) assumption. Intuitively, the ad-DRSA assumption claims that for any PPT adversary $\mathcal{A}$, there exists a subgroup $S_{\mathcal{A}}$ of $\mathbb{QR}_N$ such that $\mathcal{A}$ does not distinguish a random element of $S_{\mathcal{A}}$ from that of $\mathbb{QR}_N$. More precisely, the ad-DRSA assumption is parametrized by an integer $m \leq M$, and m-ad-DRSA assumption claims that for any PPT algorithm $\mathcal{A}$, there exists at least one choice of $p_1, \ldots p_m$ out of all ℓ_B-bit primes such that $\mathcal{A}$ cannot distinguish $g^{p_1 \cdots p_m}$ from g where $g \xleftarrow{\$} \mathbb{QR}_N$. By this definition, if there exists a PPT algorithm $\mathcal{A}$ that breaks the m-ad-DRSA assumption, then $\mathcal{A}$ distinguishes $g^{p_1 \cdots p_m}$ from g for *all* choices of $p_1, \ldots, p_m$. If m is sufficiently smaller than M, then there exists "many" choices of $p_1, \ldots, p_m$ and thus one can find "many" primes that divides $\Phi(N)$: One can find at least one such prime for each choice of $p_1, \ldots, p_m$ by the similar method as in the case of the DRSA assumption. Then the product of these primes is a large divisor of $\Phi(N)$ and thus one can factorize N by using the Coppersmith theorem [6], which claims that if one is given a "large" divisor of $\Phi(N)$, then one can factorize N efficiently. Thus, the m-ad-DRSA assumption is reduced to the factoring assumption.

Remark 1 *We remark that if m is so small that there exists a choice of $p_1, \ldots, p_m$, all of which are coprime to $\Phi(N)$, then the m-ad-DRSA assumption is trivial since in that case g and $g^{p_1 \cdots p_m}$ are distributed identically. We show that there exists a parameter choice such that m-ad-DRSA assumption is non-trivial and it can be reduced to the factoring assumption simultaneously.*

How to use the ad-DRSA assumption. As explained above, we show a reduction from the ad-DRSA assumption, which is a certain type of a subgroup assumption, to the factoring assumption. However, the ad-DRSA assumption is not an ordinary subgroup decision assumption: Roughly speaking, it only claims that for any PPT adversary $\mathcal{A}$, there exists a subgroup $S_{\mathcal{A}} \in \mathbb{QR}_N$ such that $\mathcal{A}$ cannot distinguish random elements of $S_{\mathcal{A}}$ from $\mathbb{QR}_N$. Therefore, it cannot be used for constructions where elements of a subgroup are used in the real descriptions of the scheme. On the other hand, if elements of a subgroup are used only in the security proof, the ad-DRSA assumption suffices. We give two examples of such cases.

One is ad-LTDFs. As explained in Sec. 1.2, ad-LTDFs is a relaxation of LTDFs such that descriptions of lossy functions can depend on an adversary. For constructing ad-LTDFs based on the ad-DRSA assumption, we simply imitate the construction by Xue et al. [36], who constructed LTDFs based on the (standard) DRSA assumption. We observe that in their construction, the descriptions of injective functions consist only of elements of $\mathbb{QR}_N$, and elements of its subgroup are used only in the descriptions of lossy functions. Therefore even if we replace the DRSA assumption with the ad-DRSA assumption, only lossy functions depend on an adversary. This meets the definition of the ad-LTDFs.

The other is the hash-proof system-based CCA secure public key encryption. Hofheniz and Kiltz [16] introduced the concept of constraind CCA (CCCA) security, and showed efficient constructions of CCA secure public key encryption schemes based on a hash proof system, which can be constructed from any subgroup decision assumption [8]. Though elements of a subgroup are used in the real protocol of their original construction, it is easy to see that even if elements of a subgroup are replaced with those of a larger group, the scheme is still secure because they are indistinguishable by the assumption. Thus that scheme can be instantiated based on the ad-DRSA assumption.

1.4 Discussion

Plausibility of the factoring assumption w.r.t. SS moduli. Here, we discuss the plausibility of the assumption we used. SS moduli was first introduced by Groth [13] in 2005 and they have been used in some works [25, 36, 37]. All of these works assume the factoring assumption w.r.t. SS moduli (or more stronger assumptions). On the other hand, in 2011, Coron et.al. [7] gave a cryptanalysis against the Groth's work [13]. However, they did not improve attacks against SS moduli. Thus, we can say that SS moduli has attracted a certain amount of attention in the sense of both constructions and cryptanalysis, but no fatal attack is found thus far. Therefore we believe that the hardness of factoring SS moduli is rather reliable.

Interpretation of our result. In this paper, we constructed a weaker variant of LTDF (ad-LTDF) based on the factoring assumption w.r.t. SS moduli. One may wonder how meaningful our result is since an SS modulus is not an RSA modulus of a standard form. We believe that our result is meaningful in terms of that we constructed an "LTDF-like primitive" (ad-LTDF), which can replace LTDFs in many applications, based on a search assumption (factoring w.r.t. SS moduli) rather than a decision assumption. Although the application given in this paper is limited to the case of SS moduli, we hope that our new search-to-decision reduction technique can be extended to other general settings.

Limitation of ad-LTDFs. Though ad-LTDFs can replace LTDFs in many cases, there exist some LTDF-based primitives that cannot be obtained from ad-LTDFs. A typical example is the oblivious transfer protocol proposed by Peikert and Waters [27]. The reason why we cannot construct the scheme based on ad-LTDFs is that in the scheme, a lossy function is explicitly required. Specifically, a receiver sends a pair of injective and lossy functions to a sender. Since we cannot specify a lossy function before fixing an adversary, we cannot instantiate this scheme based on ad-LTDFs.

1.5 Related Work

Deterministic encryption. Bellare et al. [1] initiated the study of DE, defined the security notion of DE called the PRIV security, and gave constructions of PRIV secure DE schemes in the random oracle model. Boldyreva et al. [5] slightly weakened the PRIV security to what they call the PRIV security for block-sources, and constructed DE schemes with this security in the standard model based on LTDFs. Bellare et al. [2] showed that DE scheme with a weaker security notion (where messages are uniformly random) can be constructed from any one-way trapdoor permutation. In this paper, we only consider the PRIV security for block-sources as defined in [5].

Factoring based CCA secure PKE schemes. In 2009, Hofheinz and Kiltz [18] proposed the first practical CCA secure PKE scheme under the factoring assumption in the standard model. After that, many variants of the scheme are proposed thus far [25, 24, 23, 37]. However, none of them improve the ciphertext overhead of the scheme. On the other hand, the ciphretext overhead of our proposed scheme is shorter than those of them.

2 Preliminaries

Here we review some basic notations and definitions.

2.1 Notations

We use $\mathbb{N}$ to denote the set of all natural numbers and $[n]$ to denote the set $\{1, \dots n\}$ for $n \in \mathbb{N}$. If S is a finite set, then we use $x \overset{\$}{\leftarrow} S$ to denote that x is chosen uniformly at random from S. If $\mathcal{A}$ is an algorithm, we use $x \leftarrow \mathcal{A}(y)$ to denote that x is output by $\mathcal{A}$ whose input is y. For a finite set S, $|S|$ denotes the cardinality of S. For a real number x, $\lceil x \rceil$ denotes the smallest integer not smaller than x and $\lfloor x \rfloor$ denotes the largest integer not larger than x. For a bit string a, ℓ_a denotes the length of a. For a function f in λ, we often denote f to mean $f(\lambda)$ for notational simplicity. We say that a function $f(\cdot) : \mathbb{N} \to [0,1]$ is negligible if for all polynomials $p(\cdot)$ and all sufficiently large $\lambda \in \mathbb{N}$, we have $f(\lambda) < |1/p(\lambda)|$. We say f is overwhelming if $1-f$ is negligible. We say that a function $f(\cdot) : \mathbb{N} \to [0,1]$ is noticeable if there exists a polynomial p such that for all sufficiently large λ, we have $f(\lambda) > |1/p(\lambda)|$. We say that an algorithm $\mathcal{A}$ is probabilistic polynomial time (PPT) if there exists a polynomial p such that running time of $\mathcal{A}$ with input length λ is less than $p(\lambda)$. We use $a|b$ to mean that a is a divisor of b. For a natural number N, $\varPhi(N)$ denote the number of natural numbers smaller than N that are coprime to N. For random variables X and Y, $\varDelta(X,Y)$ denote the statistical distance between them. We use the fact that for any (probabilistic) function f, $\varDelta(f(X), f(Y)) \leq \varDelta(X,Y)$ holds, and that $\varDelta((X_1, Z), (Y_1, Z)) = \mathbb{E}_Z[\varDelta(X_1, Y_1)]$ where $\mathbb{E}$ denotes the expected value. For random variables X and Y, we define min-entropy of X as $H_\infty(X) := -\log(\max_x \Pr[X = x])$ and average min-entropy of X given Y as $\tilde{H}_\infty(X|Y) := -\log(\varSigma_y \Pr[Y = y] \max_x \Pr[X = x|Y = y])$. We use λ to denote the security parameter.

2.2 Syntax and Security Notions

Here, we review definitions of cryptographic primitives.

Pairwise independent hash function. We say that a family $\mathcal{H}$ of hash functions from $\{0,1\}^n$ to $\{0,1\}^m$ is pairwise independent if for any $x_1 \neq x_2 \in \{0,1\}^n$ and $y_1, y_2 \in \{0,1\}^m$, $\Pr[H(x_1) = y_1 \wedge H(x_2) = y_2 : H \xleftarrow{\$} \mathcal{H}] = 2^{-2m}$ holds.

Collision resistant hash function. We formalize a collision resistant hash function as a pair of PPT algorithms $\varPi = (\mathsf{Gen}, \mathsf{Eval})$. Gen takes the security parameter 1^λ as input and outputs a function description h. Eval is a deterministic algorithm that takes a function description h and x as input and outputs $h(x)$. We require that for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{CR}}_{\mathcal{A},\Pi}(\lambda) := \Pr[h(x) = h(x'), x \neq x' : h \leftarrow \mathsf{Gen}(1^\lambda); (x, x') \leftarrow \mathcal{A}(h)]$ is negligible.

Public key encryption. A PKE scheme consists of three algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Gen takes the security parameter 1^λ as input and outputs (PK, SK), where PK is a public key and SK is a secret key. Enc takes a public key PK and a message msg as input and outputs a ciphertext C. Dec takes a secret key SK and a ciphertext C as input and outputs a massage msg. We require that for all (PK, SK) output by Gen, all msg and all C output by $\mathsf{Enc}(PK, msg)$, we have $\mathsf{Dec}(SK, C) = msg$.

For ATK $\in \{\text{CPA}, \text{CCA}\}$, a public key encryption scheme $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is ATK secure if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\mathsf{Adv}^{\mathsf{ATK}}_{\mathcal{A},\mathsf{PKE}}(\lambda) := |\Pr[b = b' : (PK, SK) \leftarrow \mathsf{Gen}(1^\lambda); (msg_0, msg_1, st) \leftarrow \mathcal{A}_1^{\mathcal{O}_1}(PK); b \xleftarrow{\$} \{0,1\}; C^* \leftarrow \mathsf{Enc}(PK, msg_b); b \leftarrow \mathcal{A}_2^{\mathcal{O}_2}(PK, C^*, st)] - 1/2|$ is negligible where if ATK=CPA, then $\mathcal{O}_i$ $(i = 1, 2)$ is an oracle that always returns $\perp$, and if ATK=CCA, then $\mathcal{O}_i$ $(i = 1, 2)$ is a decryption oracle that is given a ciphertext C and returns

$\mathsf{Dec}(SK, C)$ if $i = 1$ or $C \neq C^*$ and otherwise $\perp$.

Key encapsulation mechanism. Here, we review the definition of key encapsulation mechanism (KEM) and its security. It is shown that a CCA secure PKE scheme is obtained by combining a constrained CCA (CCCA) secure KEM and a CCA secure authenticated symmetric key encryption scheme [16]. In the following, we recall the definitions of KEM and its CCCA security.

A KEM consists of three algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Gen takes a security parameter 1^λ as input and outputs (PK, SK), where PK is a public key and SK is a secret key. Enc takes a public key PK as input and outputs (C, K), where C is a ciphertext and K is a symmetric key. Dec takes a secret key SK and a ciphertext C as input and outputs a key K with length ℓ_K or $\perp$. We require that for all (PK, SK) output by Gen and all (C, K) output by $\mathsf{Enc}(PK)$, we have $\mathsf{Dec}(SK, C) = K$.

To define the CCCA security of $\mathsf{KEM} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$, we consider the following game between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. First, $\mathcal{C}$ generates $(PK, SK) \leftarrow \mathsf{Gen}(1^\lambda)$ and $(C^*, K) \leftarrow \mathsf{Enc}(PK)$, chooses a random bit $b \xleftarrow{\$} \{0,1\}$, and sets $K^* := K$ if $b = 1$ and otherwise $K^* \xleftarrow{\$} \{0,1\}^{\ell_K}$. Then (PK, C^*, K^*) is given to the adversary $\mathcal{A}$. In the game, $\mathcal{A}$ can query pairs of ciphertexts and predicates any number of times. When $\mathcal{A}$ queries (C, pred), $\mathcal{C}$ computes $K \leftarrow \mathsf{Dec}(SK, C)$ and returns K to $\mathcal{A}$ if $C \neq C^*$ and $\mathsf{pred}(K) = 1$, and otherwise $\perp$. Finally, $\mathcal{A}$ outputs a bit b'. We define the CCCA advantage of $\mathcal{A}$ as $\mathsf{Adv}^{\mathsf{CCCA}}_{\mathcal{A},\mathsf{KEM}}(\lambda) := |\Pr[b = b'] - 1/2|$. We say that KEM is CCCA secure if $\mathsf{Adv}^{\mathsf{CCCA}}_{\mathcal{A},\mathsf{KEM}}(\lambda)$ is negligible for any PPT *valid* adversary $\mathcal{A}$, where "valid" is defined below.

Before defining "valid" , we prepare two definitions. We say that a predicate pred is *non-trivial* if $\Pr[\mathsf{pred}(K) = 1 : K \xleftarrow{\$} \{0,1\}^{\ell_K}]$ is negligible. We say that an algorithm $\mathcal{C}'$ is an *alternative challenger* if it has the same syntax as the real challenger $\mathcal{C}$. We say that an adversary $\mathcal{A}$ is valid if for any PPT alternative challenger $\mathcal{C}'$, all predicates pred queried by $\mathcal{A}$ in the game between $\mathcal{A}$ and $\mathcal{C}'$ are non-trivial.

Though the above definition of the CCCA security slightly differs from the original definition given in [16], we can easily prove that our definition still yields the "hybrid encryption theorem" that a CCA secure PKE scheme can be obtained by a CCCA secure KEM and authenticated symmetric key encryption.

Deterministic encryption. A deterministic encryption scheme consists of three algorithms $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$. Gen takes a security parameter 1^λ as input and outputs (PK, SK), where PK is a public key and SK is a secret key. Enc is a deterministic algorithm that takes a public key PK and a message msg as input and outputs a ciphertext C. Dec takes a secret key SK and a ciphertext C as input and outputs a message msg or $\perp$. We require that for all msg, (PK, SK) output by Gen and C output by $\mathsf{Enc}(PK, msg)$, we have $\mathsf{Dec}(SK, C) = msg$.

We recall security notions for deterministic encryption following [5]. In [5], the authors considered three security notions called PRIV, PRIV1 and PRIV1-

IND, and proved all of them are equivalent. Therefore we consider only the simplest security definition PRIV1-IND in this paper. A random variable X over $\{0,1\}^n$ is called a (u,n)-source if $H_\infty(X) \geq u$. For ATK $\in \{\mathrm{CPA}, \mathrm{CCA}\}$, a deterministic encryption scheme $\mathsf{DE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ for ℓ-bit message is PRIV1-IND-ATK secure for (t,n)-sources if for any (t,n)-sources M_0 and M_1 and all PPT adversaries $\mathcal{A}$, $\mathsf{Adv}^{\mathsf{PRIV1-IND-ATK}}_{\mathcal{A},M_0,M_1,\mathsf{DE}}(\lambda) := |\Pr[b = b' : (PK, SK) \leftarrow \mathsf{Gen}(1^\lambda); b \xleftarrow{\$} \{0,1\}; msg^* \xleftarrow{\$} M_b; C^* \leftarrow \mathsf{Enc}(PK, msg^*); b' \leftarrow \mathcal{A}^{\mathcal{O}}(PK, C^*)] - 1/2|$ is negligible where if ATK=CPA, then $\mathcal{O}$ is an oracle that always returns $\perp$, and if ATK=CCA, then $\mathcal{O}$ is an decryption oracle that is given a ciphertext C and returns $\mathsf{Dec}(SK, C)$ if $C \neq C^*$ and otherwise $\perp$.

2.3 Known Lemmas

Here, we review three known lemmas used in this paper. First, we review a simple variant of the Hoeffding inequality [15].

Lemma 1 *(Hoeffding inequality) Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be probability distributions over $\{0,1\}$. Let $X_1, \ldots, X_K$ be K independent random variables with the distribution $\mathcal{D}_1$ and $Y_1, \ldots, Y_K$ be K independent random variables with the distribution $\mathcal{D}_2$. If we define $\epsilon := |\Pr[X = 1 : X \xleftarrow{\$} \mathcal{D}_1] - \Pr[Y = 1 : Y \xleftarrow{\$} \mathcal{D}_2]|$, then $\Pr[|\frac{|\Sigma_{k=1}^K X_k - \Sigma_{i=k}^K Y_k|}{K} - \epsilon| \geq \delta] \leq 4e^{-\delta^2 K/2}$ holds.*

The following is the generalized leftover hash lemma [10].

Lemma 2 *(Generalized leftover hash lemma) Let $X \in \{0,1\}^{n_1}$ and Y be random variables. Let $\mathcal{H}$ be a family of pairwise independent hash function from $\{0,1\}^{n_1}$ to $\{0,1\}^{n_2}$. Then we have $\Delta((H(X), H, Y), (U, H, Y)) \leq \delta$ where $H \xleftarrow{\$} \mathcal{H}$ as long as $\tilde{H}_\infty(X|Y) \geq n_2 + 2\log(1/\delta)$.*

The following is the "crooked version" of the above lemma proven by Boldyreva et al. [5].

Lemma 3 *(Generalized crooked leftover hash lemma [5, Lemma7.1]) Let $X \in \{0,1\}^n$ and Y be random variables. Let $\mathcal{H}$ be a family of pairwise independent hash function from $\{0,1\}^n$ to R and f be a function from R to S. Then for $H \xleftarrow{\$} \mathcal{H}$, we have $\Delta((f(H(X)), H, Y), (f(U), H, Y)) \leq \delta$ as long as $\tilde{H}_\infty(X|Y) \geq \log|S| + 2\log(1/\delta) - 2$.*

Finally, we review the Coppersmith theorem about bivariate integer equations. The following lemma is a special case of [6, Theorem 3].

Lemma 4 *Let $p(x,y) = a + bx + cy$ be a polynomial over $\mathbb{Z}$. For positive integers X, Y and $W = \max\{a, bX, cY\}$, if $XY < 2^{-8} \cdot W$ holds, then one can find all solutions (x_0, y_0) such that $p(x_0, y_0) = 0$, $|x_0| < X$ and $|y_0| < Y$ in time polynomial in $\log_2 W$.*

2.4 Semi-smooth RSA subgroup modulus

For integers ℓ_B, t_p and t_q, We say that $N = PQ = (2pp' + 1)(2qq' + 1)$ is an (ℓ_B, t_p, t_q)-semi-smooth RSA subgroup ((ℓ_B, t_p, t_q)-SS) modulus if the following conditions hold.

- P and Q are distinct prime numbers with the same length that satisfy $\gcd(P - 1, Q - 1) = 2$.
- p' and q' are distinct primes larger than 2^{ℓ_B}.
- p and q are products of t_p and t_q distinct ℓ_B-bit primes. Here, an ℓ_B-bit prime means a prime number between $2^{\ell_B - 1}$ and 2^{ℓ_B}. We note that we have $\gcd(p, q) = 1$ since we have $\gcd(P - 1, Q - 1) = 2$.

We define $t := t_p + t_q$. Let $\mathcal{P}_{\ell_B}$ be the set of all ℓ_B-bit primes, and $M_{\ell_B} := |\mathcal{P}_{\ell_B}|$. We define the group of quadratic residues as $\mathbb{QR}_N := \{u^2 : u \in \mathbb{Z}_N^*\}$. This is a subgroup of $\mathbb{Z}_N^*$, and a cyclic group of order $pqp'q'$. Then there exists unique subgroups of order $p'q'$ and pq, and we denote them by G and $G^\perp$ respectively. Then we have $\mathbb{QR}_N = G \times G^\perp$. That is, for any element $g \in \mathbb{QR}_N$, we can uniquely represent $g = g(G)g(G^\perp)$ by using $g(G) \in G$ and $g(G^\perp) \in G^\perp$. Moreover, if the factorization of N is given, then we can compute $g(G)$ and $g(G^\perp)$ from g efficiently.

When N is an SS modulus, we cannot say that a random element g of $\mathbb{QR}_N$ is a generator (i.e., $\mathrm{ord}(g) = pqp'q'$) with overwhelming probability. However, we can prove that g has an order larger than a certain value with overwhelming probability.

Lemma 5 *([13, Lemma2]) Let N be an (ℓ_B, t_p, t_q)-SS modulus. For any integer $d < t$ if $\frac{(t2^{1-\ell_B})^{d+1}}{(1-t2^{1-\ell_B})(d+1)!}$ is negligible, then $\Pr[\mathrm{ord}(g) \geq p'q'2^{(t-d)(\ell_B-1)} : g \xleftarrow{\$} \mathbb{QR}_N]$ is overwhelming. Especially, $\Pr[\mathrm{ord}(g(G^\perp)) \geq 2^{(t-d)(\ell_B-1)} : g \xleftarrow{\$} \mathbb{QR}_N]$ is overwhelming.*

When ℓ_B is small, $\mathrm{ord}(G^\perp)$ is smooth, and therefore the discrete logarithm on the group can be solved efficiently by the Pohlig-Hellman algorithm [28].

Lemma 6 *([13]) If $\ell_B = O(\log \lambda)$, then the discrete logarithm problem on $G^\perp$ can be solved efficiently. More precisely, there exists a PPT algorithm that, given an (ℓ_B, t_p, t_q)-SS modulus N, $g \in G^\perp$ and g^x, outputs $x \mod \mathrm{ord}(g)$.*

By combining the above lemmas, we obtain the following lemma.

Lemma 7 *Let N be an (ℓ_B, t_p, t_q)-SS modulus and we assume $\ell_B = O(\log(\lambda))$. If $\frac{(t2^{1-\ell_B})^{d+1}}{(1-t2^{1-\ell_B})(d+1)!}$ is negligible and $x \leq 2^{(t-d)(\ell_B-1)}$ holds, then there exists a PPT algorithm PLog that, given P,Q, g, g^x, outputs x with overwhelming probability where $g \xleftarrow{\$} \mathbb{QR}_N$.*

Hardness assumptions. Here, we give definitions of two hardness assumptions. Let IGen be an algorithm that is given the security parameter 1^λ and outputs an (ℓ_B, t_p, t_q)-SS modulus with its factorization. We first define the factoring assumption.

Definition 1 *We say that the factoring assumption holds with respect to* IGen *if for any PPT algorithm* $\mathcal{A}$*,* $\Pr[\mathcal{A}(N) \in \{P, Q\} : (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda)]$ *is negligible.*

Next, we define the *decisional RSA subgroup* (DRSA) assumption proposed by Groth [13]. This assumption claims that any PPT algorithm cannot distinguish a random element of G from that of $\mathbb{QR}_N$. We note that actually we do not use this assumption in this paper. We include this only for the information of the reader.

Definition 2 *We say that the decisional RSA subgroup (DRSA) assumption holds with respect to* IGen *if for any PPT algorithm* $\mathcal{A}$*,* $|\Pr[1 \leftarrow \mathcal{A}(N, g) : (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda); g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g) : (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda); g \xleftarrow{\$} G]|$ *is negligible.*

Attacks. We review factoring attacks against SS moduli as discussed in [13]. As shown in [13], by using Pollard's ρ-method [30], we can factorize an SS modulus in time $\tilde{O}(\min(\sqrt{p'}, \sqrt{q'}))$. As another method, by using Naccache et al.'s algorithm [26], if a divisor of $P - 1$ or $Q - 1$ larger than $N^{1/4}$ is given, then N can be factorized efficiently. Thus ℓ_B should be large enough so that it is difficult to guess a significant portion of factors of p or q. In 2011, Coron et al. [7] proposed a new factoring algorithm for a certain class of RSA moduli that includes SS moduli. For the case of SS moduli, their algorithm work in time $\tilde{O}(\min(\sqrt{p'}, \sqrt{q'}))$, which matches the time complexity of Pollard's ρ-method. As observed in [13], other methods such as the baby-step giant-step algorithm [34], Pollard's λ-method [31] or Pollard's $p - 1$ method [29] require $O(\min(p', q'))$ time.

The above attacks use the structure of SS moduli. On the other hand, there are algorithms such as the elliptic curve method [22] or the general number field sieve [9], which can be applied to general RSA moduli. Among these algorithms, general number field sieve is asymptotically the most efficient and its heuristic running time is $\exp((1.92 + o(1)) \ln(N)^{1/3} \ln\ln(N)^{2/3})$.

Parameter settings. Here, we discuss parameter settings of SS moduli. We have to set parameters to avoid the above attacks. We first give an asymptotic parameter setting. We set $\ell_{p'} = \ell_{q'} = O(\lambda)$, $\ell_B = \lfloor 4 \log \lambda \rfloor$ and $t_p = t_q = O(\lambda^3 / \log \lambda)$ (then we have $\ell_N \approx \ell_{p'} + \ell_{q'} + t\ell_B = O(\lambda^3)$). In this setting, we have $M_{\ell_B} = O(\lambda^4 / \log \lambda)$ by the prime number theorem and thus there exists exponentially many choices of p and q. If we set $d := \lfloor t/4 \rfloor$, then $\frac{(t2^{1-\ell_B})^{d+1}}{(1-t2^{1-\ell_B})(d+1)!}$ is negligible[1]. We use the fact that in this parameter setting, given N, $g \in \mathbb{QR}_N$ and $p_1, \ldots, p_m$ for $m \leq M_{\ell_B}$, $g^{p_1 \cdots p_m}$ can be computed in polynomial time in λ. This is because we have $m \leq M_{\ell_B} = O(\lambda^4 / \log(\lambda))$ and $p_1 \ldots p_m \leq 2^{\ell_B M_{\ell_B}} = 2^{O(\lambda^4)}$, and thus $p_1 \ldots p_m$-th power can be computed by $O(\lambda^4)$ multiplications. We use this asymptotic parameter setting throughout the paper. As a concrete parameter, Groth [13] proposed to set $\ell_p' = \ell_q' = 160$, $\ell_B = 15$, $t_p = t_q = 32$ and $d = 7$ for 80-bit security (then we have $\ell_N = 160 \cdot 2 + 15 \cdot 2 \cdot 32 = 1280$). We use

[1] In fact, d can be set as $d := \lfloor ct \rfloor$ for any small enough constant c.

this parameter for the construction of CCA secure PKE scheme with compact ciphertext (Section 6). However, this parameter does not give us enough lossiness in the construction of ad-LTDFs. Thus we propose to set $\ell'_p = \ell'_q = 160$, $\ell_B = 15$, $t_p = t_q = 70$ and $d = 8$ (then we have $\ell_N = 160 \cdot 2 + 15 \cdot 2 \cdot 70 = 2420$) for 80-bit security for other applications (Sec. 4 and 5). We note that the number of $\ell_B = 15$-bit primes is 1612. Therefore the possible choice of $t = 64$ or 140 primes out of them is much larger than 2^{80} and thus it is hard to guess the significant portion of their factors.

3 Adversary-dependent Decisional RSA Subgroup Assumption

In this section, we generalize the DRSA assumption. Specifically, we define the m-adversary-dependent decisional RSA subgroup (m-ad-DRSA) assumption for any integer $m \leq M_{\ell_B}$ with respect to (ℓ_B, t_p, t_q)-SS moduli. Intuitively, this assumption claims that for any PPT algorithm $\mathcal{A}$, there exist distinct ℓ_B-bit primes $p_1, \dots, p_m$ such that $\mathcal{A}$ does not distinguish g from $g^{p_1 \cdots p_m}$ where g is a random element of $\mathbb{QR}_N$. We prove that under a certain condition, the m-ad-DRSA assumption holds under the factoring assumption.

First we give the precise definition of the m-ad-DRSA assumption.

Definition 3 *Let* IGen *be a PPT algorithm that generates an* (ℓ_B, t_p, t_q)*-SS RSA modulus. We say that for any integer* $m \leq M_{\ell_B}$*, the* m*-adversary-dependent decisional RSA subgroup (*m*-ad-DRSA) assumption holds with respect to* IGen *if for any noticeable function* ϵ *and PPT algorithm* $\mathcal{A}$*, there exists a PPT algorithm* $\mathcal{S}_{\mathcal{A},\epsilon}$ *that is given* (ℓ_B, t_p, t_q)*-SS RSA modulus* N *and outputs distinct* ℓ_B*-bit primes* $p_1, \dots, p_m$*, such that the following is satisfied. If we let*

$$P_0 := \Pr\left[1 \leftarrow \mathcal{A}(N, g) : \begin{array}{c} (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda) \\ g \xleftarrow{\$} \mathbb{QR}_N \end{array}\right]$$

$$P_1 := \Pr\left[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : \begin{array}{c} (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda) \\ g \xleftarrow{\$} \mathbb{QR}_N \\ \{p_1, \dots, p_m\} \leftarrow \mathcal{S}_{\mathcal{A},\epsilon}(N) \end{array}\right]$$

then we have $|P_0 - P_1| \leq \epsilon(\lambda)$ *for sufficiently large* λ.

Remark 2 *One may think that the assumption defined above cannot be used for proving security of any cryptographic scheme since* ϵ *is noticeable. However, an important remark here is that* ϵ *can be an arbitrary noticeable function. Thus, in security proofs, we can set* ϵ *depending on an adversary* $\mathcal{A}$*'s advantage against the scheme that we want to prove secure, such that* ϵ *is smaller than the advantage of* $\mathcal{A}$ *(for infinitely many security parameters). This can be done if* $\mathcal{A}$ *breaks the security of the scheme since in these cases, the advantage of* $\mathcal{A}$ *should be non-negligible. See security proofs in Sec. 5 and 6 to see this argument indeed works.*

Remark 3 *In the above definition, if m is so small that there exists a choice of $p_1, \ldots, p_m$, all of which are coprime to $\Phi(N)$, then $g^{p_1 \cdots p_m}$ is distributed uniformly on $\mathbb{QR}_N$. In this case, m-ad-DRSA assumption is trivial. This occurs if and only if we have $M_{\ell_B} - m \geq t$. In this paper, we set m to be relatively large so that m-ad-DRSA assumption is non-trivial. (See Remark 4.)*

The following theorem claims that the m-ad-DRSA assumption holds under the factoring assumption if m is small enough.

Theorem 1 *Let* IGen *be a PPT algorithm that generates an (ℓ_B, t_p, t_q)-SS RSA modulus where $\ell_B = O(\log \lambda)$. If the factoring assumption holds with respect to* IGen *and there exists a constant c such that $(M_{\ell_B} - m + 1)(\ell_B - 1) \geq (1/2 + c)\ell_N$ holds, then the m-ad-DRSA assumption holds with respect to* IGen.

Remark 4 *If we set $m := \lfloor M_{\ell_B} + 1 - (1/2 + c)\ell_N/(\ell_B - 1) \rfloor$ for sufficiently small c, then by the above theorem, the m-ad-DRSA assumption holds under the factoring assumption. Moreover, by setting the parameter as given in 2.4, we have $M_{\ell_B} - m \approx (1/2 + c)\ell_N/(\ell_B - 1) \approx (1/2 + c)(\ell_{p'} + \ell_{q'} + t\ell_B)/\ell_B \leq t$ for sufficiently large λ if $c < 1/2$ since $t = O(\lambda^3/\log \lambda)$ and $\ell_{p'} = \ell_{q'} = O(\lambda)$. Thus the m-ad-DRSA assumption is non-trivial.*

Before proving the theorem, we prepare a lemma related to the Coppersmith attack. Though a heuristic proof appeared in [26], to the best of our knowledge, this has not been proven rigorously in the literature.

Lemma 8 *Let P and Q be primes with the same length and $N = PQ$. Let e be a divisor of $\Phi(N) = (P-1)(Q-1)$. If there exists a positive constant c such that $e > N^{1/2+c}$ holds, then there exists a polynomial time algorithm that is given N and e, and factorizes N.*

Proof. We define e_1 and e_2 such that $e = e_1 e_2$, $e_1 | P-1$ and $e_2 | Q-1$. (Note that we cannot always compute e_1 and e_2 from e.) Then we can write $P = e_1 k_1 + 1$ and $Q = e_2 k_2 + 1$ by using integers k_1 and k_2. Then we have $N = PQ = (e_1 k_1 + 1)(e_2 k_2 + 1) = e k_1 k_2 + e_1 k_1 + e_2 k_2 + 1$. Therefore if we define $p(x, y) = N + ex + y$, then $p(x, y) = 0$ has a solution $(x_0, y_0) = (-k_1 k_2, -(e_1 k_1 + e_2 k_2 + 1))$. Let $X := N^{1/2-c}$, $Y := 3N^{1/2}$ and $W := \max(N, eX, Y)$. One can see that $|x_0| < X$, $|y_0| < Y$ and $XY = 3N^{1-c} < 2^{-8} \cdot N \leq 2^{-8} \cdot W$ hold (for sufficiently large N). Therefore one can compute the solution $(x_0, y_0) = (-k_1 k_2, -(e_1 k_1 + e_2 k_2 + 1))$ in polynomial time in $\log N$ by Lemma 4. Then one can compute $P + Q = e_1 k_1 + e_2 k_2 + 2 = -y_0 + 1$ and factorize N. $\square$

Intuition for the proof of Theorem 1. Here, we give an intuition for the proof of Theorem 1. We remark that the following argument is not a rigorous one. What we have to do is to construct a PPT algorithm $\mathcal{S}_{\mathcal{A},\epsilon}$ that is given N and outputs $\{p_1, \ldots, p_m\}$ such that $\mathcal{A}$'s advantage to distinguish g from $g^{p_1 \cdots p_m}$ is smaller than ϵ where $g \xleftarrow{\$} \mathbb{QR}_N$. Let list $L := \mathcal{P}_{\ell_B}$, which is the set of all ℓ_B-bit primes. First, $\mathcal{S}_{\mathcal{A},\epsilon}$ randomly chooses m distinct primes $\{p_1, \ldots, p_m\}$ from L and test whether $\mathcal{A}$'s advantage to distinguish g from $g^{p_1 \cdots p_m}$ is smaller than ϵ or not.

More precisely, $\mathcal{S}_{\mathcal{A},\epsilon}$ approximates $\mathcal{A}$'s advantage by iterating the execution of $\mathcal{A}(g)$ and $\mathcal{A}(g^{p_1 \cdots p_m})$ for independently random $g \xleftarrow{\$} \mathbb{QR}_N$ a number of times and counting the number that each of them outputs 1. We denote the approximated advantage by ϵ'. Due to the Hoeffding inequality [15], the approximation error can be made smaller than $\epsilon/4$ by polynomial times iterations since ϵ is noticeable. If $\epsilon' < \epsilon/2$, then $\mathcal{A}$'s real advantage is smaller than $3\epsilon/4 < \epsilon$ and thus $\mathcal{S}_{\mathcal{A},\epsilon}$ outputs $\{p_1, \ldots, p_m\}$ and halts. Otherwise, $\mathcal{A}$'s advantage to distinguish g from $g^{p_1 \cdots p_m}$ is larger than $\epsilon/4$. Then there exists p_j such that $\mathcal{A}$'s advantage to distinguish $g^{p_1 \cdots p_{j-1}}$ from $g^{p_1 \cdots p_j}$ is larger than $\epsilon/(4m)$ by the hybrid argument. $\mathcal{S}_{\mathcal{A},\epsilon}$ can find this p_j in polynomial time since $\epsilon/(4m)$ is noticeable. We remark that we have $p_j | \Phi(N)$. This is because, otherwise $\mathcal{A}$'s advantage to distinguish $g^{p_1 \cdots p_{j-1}}$ from $g^{p_1 \cdots p_j}$ is 0 since their distributions are completely identical and thus ϵ' should be smaller than $\epsilon/2$. Then $\mathcal{S}_{\mathcal{A},\epsilon}$ removes p_j from L. Then it randomly chooses m distinct primes $\{p_1, \ldots, p_m\}$ from L again, and do the same as the above. Then it outputs $\{p_1, \ldots, p_m\}$ and halts if approximated $\mathcal{A}$'s advantage to distinguish g from $g^{p_1 \cdots p_m}$ is smaller than $\epsilon/2$, or otherwise removes some $p_{j'} | \Phi(N)$ from L. $\mathcal{S}_{\mathcal{A},\epsilon}$ repeat this procedure many times. Assume that $\mathcal{S}_{\mathcal{A},\epsilon}$ does not halts by the time it cannot choose m distinct primes from L. By that time, $M_{\ell_B} - m + 1$ distinct ℓ_B-bit primes that divide $\Phi(N)$ are removed from L. Let e be the product of them. Then we have $e | \Phi(N)$ and $e \geq 2^{(\ell_B - 1)(M_{\ell_B} - m + 1)} \geq N^{1/2 + c}$. Therefore if e is given, then we can factorize N efficiently by Lemma 8. Thus under the factoring assumption, $\mathcal{S}_{\mathcal{A},\epsilon}$ must output some $\{p_1, \ldots, p_m\}$ before $|L|$ becomes smaller than m with overwhelming probability, and $\mathcal{A}$'s advantage to distinguish g from $g^{p_1 \cdots p_m}$ is smaller than $3\epsilon/4 < \epsilon$.

Now we give the full proof of Theorem 1

Proof. (of Theorem 1) First, we prove the following two claims.

Claim 1 *For any PPT algorithm $\mathcal{A}$ and a noticeable function δ, there exists a PPT algorithm $\mathsf{Approx}_{\mathcal{A},\delta}$ that satisfies the following. Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be descriptions of distributions that are samplable in polynomial time in λ, and $\epsilon := |\Pr[1 \leftarrow \mathcal{A}(X) : X \xleftarrow{\$} \mathcal{D}_1] - \Pr[1 \leftarrow \mathcal{A}(X) : X \xleftarrow{\$} \mathcal{D}_2]|$. Then $\mathsf{Approx}_{\mathcal{A},\delta}(1^\lambda, \mathcal{D}_1, \mathcal{D}_2)$ outputs ϵ' such that $|\epsilon' - \epsilon| \leq \delta(\lambda)$ with overwhelming probability. (We say that $\mathsf{Approx}_{\mathcal{A},\delta}$ succeeds if it outputs such ϵ'.)*

Proof. The construction of $\mathsf{Approx}_{\mathcal{A},\delta}$ is as follows.

$\mathsf{Approx}_{\mathcal{A},\delta}(1^\lambda, \mathcal{D}_1, \mathcal{D}_2)$: For $i = 1$ to K where $K := \lambda/\delta(\lambda)^2$, choose X_i and Y_i according to $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively, and run $\mathcal{A}(X_i)$ and $\mathcal{A}(Y_i)$ for each i. Let k_1 be the number of the event that $\mathcal{A}(X_i)$ outputs 1 and k_2 be the number of the event that $\mathcal{A}(Y_i)$ outputs 1. Output $|k_1 - k_2|/K$.

Since δ is noticeable, K is polynomial in λ and therefore $\mathsf{Approx}_{\mathcal{A},\delta}$ is a PPT algorithm. It can be seen by Lemma 1 that $\mathsf{Approx}_{\mathcal{A},\delta}$ satisfies the desired property. □

Claim 2 *For any PPT algorithm $\mathcal{A}$ and a noticeable function ϵ, there exists a PPT algorithm $\mathsf{Find}_{\mathcal{A},\epsilon}$ that satisfies the following. For any (ℓ_B, t_p, t_q)-SS RSA modulus N and a set $I = \{p_1, \ldots, p_m\}$ of distinct ℓ_B-bit primes, if $|\Pr[1 \leftarrow \mathcal{A}(N, g) : g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : g \xleftarrow{\$} \mathbb{QR}_N]| > \epsilon(\lambda)$ holds, then $\mathsf{Find}_{\mathcal{A},\epsilon}(N, I)$ outputs $p_j \in I$ that divides $\Phi(N)$ with overwhelming probability. (We say that $\mathsf{Find}_{\mathcal{A},\epsilon}$ succeeds if it outputs such p_j or the inequality assumed is false.)*

Proof. The construction of $\mathsf{Find}_{\mathcal{A},\epsilon}$ is as follows.

```
$\mathsf{Find}_{\mathcal{A},\epsilon}(N, I = \{p_1, \ldots, p_m\})$: Define distributions $\mathcal{D}_0 := \{(N, g) : g \xleftarrow{\$} \mathbb{QR}_N\}$
    and $\mathcal{D}_j := \{(N, g^{p_1 \cdots p_j}) : g \xleftarrow{\$} \mathbb{QR}_N\}$ $(j = 1, \ldots, m)$. For $j := 1$ to $m$, repeat
    the following.
        Compute $\epsilon' \leftarrow \mathsf{Approx}_{\mathcal{A},\epsilon/(2m)}(1^\lambda, \mathcal{D}_{j-1}, \mathcal{D}_j)$.
        If $\epsilon' > \epsilon/(2m)$, then output $p_j$ and halt.
    If it does not halt by the time the above loop is finished, then output $\perp$.
```

First, we show $\mathsf{Find}_{\mathcal{A},\epsilon}$ is a PPT algorithm. Since $m \leq M_{\ell_B}$ is polynomial in λ and thus $\epsilon/(2m)$ is noticeable, $\mathsf{Approx}_{\mathcal{A},\epsilon/(2m)}$ is a PPT algorithm. Therefore $\mathsf{Find}_{\mathcal{A},\epsilon}$ is a PPT algorithm. We prove that $\mathsf{Find}_{\mathcal{A},\epsilon}$ satisfies the desired property. First, we assume that all executions of $\mathsf{Approx}_{\mathcal{A},\epsilon/(2m)}$ called by $\mathsf{Find}_{\mathcal{A},\epsilon}$ succeed. The probability that this assumption is satisfied is overwhelming since the number of executions of $\mathsf{Approx}_{\mathcal{A},\epsilon/(2m)}$ is polynomial in λ and each execution succeeds with overwhelming probability. First, we prove that $\mathsf{Find}_{\mathcal{A},\epsilon}$ outputs any prime $p_j \in I$ if we have $|\Pr[1 \leftarrow \mathcal{A}(N, g) : g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : g \xleftarrow{\$} \mathbb{QR}_N]| > \epsilon$. By the hybrid argument, there exists $j \in [m]$ such that $|\Pr[1 \leftarrow \mathcal{A}(X) : X \xleftarrow{\$} \mathcal{D}_{j-1}] - \Pr[1 \leftarrow \mathcal{A}(X) : X \xleftarrow{\$} \mathcal{D}_j]| > \epsilon/m$ holds. For such j, if we let $\epsilon' := \mathsf{Approx}_{\mathcal{A},\epsilon/(2m)}(\mathcal{D}_{j-1}, \mathcal{D}_j)$, then we have $\epsilon' > \epsilon/m - \epsilon/(2m) = \epsilon/(2m)$ and thus p_j is output. Then we prove that if p_j is output by $\mathsf{Find}_{\mathcal{A},\epsilon}$, then $p_j | \Phi(N)$ holds. If p_j does not divide $\Phi(N)$, then p_j is coprime to $\mathrm{ord}(\mathbb{QR}_N)$, and especially p_j-th power is a permutation on the group $\{g^{p_1 \cdots p_{j-1}} : g \in \mathbb{QR}_N\}$. Therefore $\mathcal{D}_{j-1}$ and $\mathcal{D}_j$ are completely the identical distributions. Therefore we have $|\Pr[1 \leftarrow \mathcal{A}(X) : X \xleftarrow{\$} \mathcal{D}_{j-1}] - \Pr[1 \leftarrow \mathcal{A}(X) : X \xleftarrow{\$} \mathcal{D}_j]| = 0$. Thus if we let $\epsilon' := \mathsf{Approx}_{\mathcal{A},\epsilon/(2m)}(\mathcal{D}_{j-1}, \mathcal{D}_j)$, then we have $\epsilon' < \epsilon/(2m)$, and thus such p_j cannot be output. □

Then we go back to the proof of Theorem 1. For any PPT algorithm $\mathcal{A}$ and a noticeable function ϵ, we construct a PPT algorithm $\mathcal{S}_{\mathcal{A},\epsilon}$ such that $\Pr[1 \leftarrow \mathcal{A}(N, g) : (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda); g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda); g \xleftarrow{\$} \mathbb{QR}_N; \{p_1, \ldots, p_m\} \leftarrow \mathcal{S}_{\mathcal{A},\epsilon}(N)] \leq \epsilon(\lambda)$ holds for sufficiently large λ. The construction of $\mathcal{S}_{\mathcal{A},\epsilon}$ is as follows.

```
$\mathcal{S}_{\mathcal{A},\epsilon}(N)$ : Let $L := \mathcal{P}_{\ell_B}$. (Recall that $\mathcal{P}_{\ell_B}$ is the set of all $\ell_B$-bit primes.)
    While $|L| \geq m$, repeat the following.
        Choose distinct $\ell_B$-bit primes $p_1, \ldots, p_m$ from $L$ randomly, and let $I :=$
        $\{p_1, p_2, \ldots, p_m\}$, $\mathcal{D}_0 := \{(N, g) : g \xleftarrow{\$} \mathbb{QR}_N\}$ and $\mathcal{D}_m := \{(N, g^{p_1 \cdots p_m}) :$
```

$g \xleftarrow{\$} \mathbb{QR}_N\}$. Compute $\epsilon' \leftarrow \mathsf{Approx}_{\mathcal{A},\epsilon/4}(1^\lambda, \mathcal{D}_0, \mathcal{D}_m)$. If $\epsilon' < \epsilon/2$, then output I and halts. Otherwise run $\tilde{p} \leftarrow \mathsf{Find}_{\mathcal{A},\epsilon/4}(N, I)$. If $\tilde{p} \in L$ then remove $\tilde{p}$ from L, otherwise remove a random element from L.

If it does not halt by the time the above loop finishes, then it outputs $\perp$.

First, we prove that $\mathcal{S}_{\mathcal{A},\epsilon}(N)$ is a PPT algorithm. Since ϵ is noticeable, $\mathsf{Approx}_{\mathcal{A},\epsilon/4}$ and $\mathsf{Find}_{\mathcal{A},\epsilon/4}$ are PPT algorithms. Moreover the number of repeat is at most $M_{\ell_B} - m + 1 \leq M_{\ell_B}$, which is polynomial in λ. Therefore $\mathcal{S}_{\mathcal{A},\epsilon}(N)$ is a PPT algorithm.

Then we prove that $\mathcal{S}_{\mathcal{A},\epsilon}(N)$ satisfies the desired property. In the following, we assume that all executions of $\mathsf{Approx}_{\mathcal{A},\epsilon/4}$ and $\mathsf{Find}_{\mathcal{A},\epsilon/4}$ called by $\mathcal{S}_{\mathcal{A},\epsilon}(N)$ succeed. The probability that the above assumption holds is overwhelming since the number of executions is polynomial and each execution succeeds with overwhelming probability. If $\mathcal{S}_{\mathcal{A},\epsilon}$ outputs some $I = \{p_1, \ldots, p_m\}$, then we have $\Pr[1 \leftarrow \mathcal{A}(N, g) : g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : g \xleftarrow{\$} \mathbb{QR}_N]| < \epsilon' + \epsilon/4 < \epsilon/2 + \epsilon/4 = 3\epsilon/4$. Next, we prove that for overwhelming fraction of N generated by IGen, the probability that $\mathcal{S}_{\mathcal{A},\epsilon}(N)$ outputs $\perp$ is negligible. First, we prove that in each repeat, $\tilde{p}$ that is removed from L divides $\Phi(N)$. We let $\epsilon' \leftarrow \mathsf{Approx}_{\mathcal{A},\epsilon/4}(1^\lambda, \mathcal{D}_0, \mathcal{D}_m)$. If $\epsilon' \geq \epsilon/2$, then we have $|\Pr[1 \leftarrow \mathcal{A}(N, g) : g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : g \xleftarrow{\$} \mathbb{QR}_N]| > \epsilon' - \epsilon/4 \geq \epsilon/2 - \epsilon/4 = \epsilon/4$. Therefore $\mathsf{Find}_{\mathcal{A},\epsilon/4}(N, I)$ outputs $p_j \in I$ that divides $\Phi(N)$ since it succeeds. Thus if $\mathcal{S}_{\mathcal{A},\epsilon}(N)$ outputs $\perp$, then one ℓ_B-bit prime factor of $\Phi(N)$ is removed from L in each repeat, and the repeat is done $M_{\ell_B} - m + 1$ times. Therefore throughout the execution of $\mathcal{S}_{\mathcal{A},\epsilon}(N)$, $M_{\ell_B} - m + 1$ distinct ℓ_B-bit prime factors of $\Phi(N)$ are removed from L. If we let e be the product of these primes, then we have $e > (2^{\ell_B - 1})^{M_{\ell_B} - m + 1} \geq 2^{(1/2 + c)\ell_N} > N^{1/2 + c}$ and $e | \Phi(N)$. By Lemma 8, we can factorize N efficiently by using e. Therefore for overwhelming fraction of N generated by IGen, the probability that $\mathcal{S}_{\mathcal{A},\epsilon}(N)$ outputs $\perp$ is negligible under the factoring assumption. Therefore for overwhelming fraction of N generated by IGen, we have $\Pr[1 \leftarrow \mathcal{A}(N, g) : g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : \{p_1, \ldots, p_m\} \leftarrow \mathcal{S}_{\mathcal{A},\epsilon}(N); g \xleftarrow{\$} \mathbb{QR}_N]| < 3\epsilon/4$ with overwhelming probability over the randomness of $\mathcal{S}_{\mathcal{A},\epsilon}$. Since ϵ is noticeable, by the averaging argument, $\Pr[1 \leftarrow \mathcal{A}(N, g) : (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda); g \xleftarrow{\$} \mathbb{QR}_N] - \Pr[1 \leftarrow \mathcal{A}(N, g^{p_1 \cdots p_m}) : (N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda); g \xleftarrow{\$} \mathbb{QR}_N; \{p_1, \ldots, p_m\} \leftarrow \mathcal{S}_{\mathcal{A},\epsilon}(N)] \leq \epsilon(\lambda)$ holds for sufficiently large λ.

4 Adversary-dependent Lossy Trapdoor Function

In this section, we define ad-LTDFs. Then we give a construction of an ad-LTDF based on the m-ad-DRSA assumption, which can be reduced to the factoring assumption by Theorem 1.

4.1 Definition

Here we define ad-LTDFs. Intuitively, ad-LTDFs are defined by weakening LTDFs so that descriptions of lossy functions that cannot be distinguished from those of injective functions may depend on a specific distinguisher. Namely, the algorithm that generates lossy functions takes a "lossy function index" I as well as a public parameter as input, and we require that for any PPT algorithm $\mathcal{A}$, there exists at least one I such that $\mathcal{A}$ does not distinguish lossy functions generated with index I from injective functions. Moreover, we require that such I can be efficiently computed given $\mathcal{A}$. The precise definition is as follows. For integers n and k such that $0 < k < n$, an (n,k)-ad LTDF consists of 5 algorithms $(\mathsf{ParamsGen}, \mathsf{SampleInj}, \mathsf{SampleLossy}, \mathsf{Evaluation}, \mathsf{Inversion})$ with a family $\{\mathcal{I}(\lambda)\}_{\lambda \in \mathbb{N}}$ of lossy function index sets.

$\mathsf{ParamsGen}(1^\lambda) \to (PP, SP)$: It takes a security parameter 1^λ as input, and outputs a public parameter PP and a secret parameter SP.

$\mathsf{SampleInj}(PP) \to \sigma$: It takes a public parameter PP as input, and outputs a function description σ, which specifies an injective function f_σ over the domain $\{0,1\}^n$.

$\mathsf{SampleLossy}(PP, I) \to \sigma$**:** It takes a public parameter PP and a lossy function index $I \in \mathcal{I}(\lambda)$ as input, and outputs a function index σ, which specifies a "lossy" function f_σ over the domain $\{0,1\}^n$.

$\mathsf{Evaluation}(PP, \sigma, x) \to f_\sigma(x)$**:** It takes a public parameter PP, function description σ and $x \in \{0,1\}^n$ as input, and outputs $f_\sigma(x)$

$\mathsf{Inversion}(SP, \sigma, y) \to f_\sigma^{-1}(y)$**:** It takes a secret parameter SP, a function description σ and y and outputs $f_\sigma^{-1}(y)$.

We require ad-LTDFs to satisfy the following three properties.
Correctness: For all $x \in \{0,1\}^n$, we have $\mathsf{Inversion}(SP, \sigma, \mathsf{Evaluation}(PP, \sigma, x)) = x$ with overwhelming probability where $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleInj}(PP)$.

Lossiness: For all $\lambda \in \mathbb{N}$, $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$, $I \in \mathcal{I}(\lambda)$ and $\sigma \leftarrow \mathsf{SampleLossy}(PP, I)$, the image of f_σ has size at most 2^{n-k}.

Indistinguishability between injective and lossy functions. Intuitively, we require that for any PPT adversary $\mathcal{A}$, there exists at least one lossy function index $I \in \mathcal{I}(\lambda)$ such that $\mathcal{A}$ cannot distinguish an injective function from a lossy function with the lossy function index I.

The more precise definition is as follows. For any PPT adversary $\mathcal{A}$ and noticeable function $\epsilon(\lambda)$, there exists a PPT algorithm $\mathcal{S}_{\mathcal{A},\epsilon}$ that takes a public parameter PP as input and outputs $I \in \mathcal{I}(\lambda)$ such that the following is satisfied. If we let

$$P_{\mathsf{inj}} := \Pr\left[1 \leftarrow \mathcal{A}(PP, \sigma) : \begin{array}{c}(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda) \\ \sigma \leftarrow \mathsf{SampleInj}(PP)\end{array}\right]$$

$$P_{\mathsf{lossy}} := \Pr\left[1 \leftarrow \mathcal{A}(PP, \sigma) : \begin{array}{c}(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda) \\ I \leftarrow \mathcal{S}_{\mathcal{A},\epsilon}(PP) \\ \sigma \leftarrow \mathsf{SampleLossy}(PP, I)\end{array}\right]$$

then we have $|P_{\mathsf{inj}} - P_{\mathsf{lossy}}| \leq \epsilon(\lambda)$ for sufficiently large λ.

As mentioned in Remark 2, though ϵ must be noticeable in the above definition, ad-LTDFs can be used for many cryptographic applications. This is because ϵ can be set depending on the advantage of an adversary in security reductions.

Remark 5 *Besides what is explained above, there is a minor difference between the definition of ad-LTDFs and that of LTDFs. In the definition of ad-LTDFs,* ParamsGen *is explicitly separated from* SampleInj *or* SampleLossy*, whereas there is no separation between them in the definition of LTDFs [27]. This is only for simplifying the presentation, and there is no significant difference here.*

4.2 Construction

We construct an ad-LTDF based on the m-ad-DRSA assumption. Let IGen be an algorithm that generates an ℓ_N-bit (ℓ_B, t_p, t_q)-SS RSA modulus with the parameter given in Sec. 2.4 and $n := (t-d)(\ell_B - 1)$.

Definition of $\mathcal{I}(\lambda)$: $\mathcal{I}(\lambda)$ is defined as the set of all m-tuple of distinct primes of length ℓ_B. That is, we define $\mathcal{I}(\lambda) := \{\{p_1, \ldots, p_m\} : p_1, \ldots, p_m$ are distinct ℓ_B bit primes$\}$.

$\mathsf{ParamsGen}(1^\lambda) \to (PP, SP)$**:** Generate $(N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda)$, set $PP := N$ and $SP := (P, Q)$, and output (PP, SP).

$\mathsf{SampleInj}(PP = N) \to \sigma$**:** Choose $g \xleftarrow{\$} \mathbb{QR}_N$ and output $\sigma := g$.

$\mathsf{SampleLossy}(PP = N, I = \{p_1, \ldots, p_m\}) \to \sigma$**:** Choose $g \xleftarrow{\$} \mathbb{QR}_N$ and output $\sigma := g^{p_1 \cdots p_m}$.

$\mathsf{Evaluation}(PP = N, \sigma = g, x \in \{0,1\}^n) \to f_\sigma(x)$**:** Interpret x as an element of $[2^n]$ and output g^x.

$\mathsf{Inversion}(SP = (P, Q), \sigma = g, y) \to f_\sigma^{-1}(y)$**:** Compute $x = \mathsf{PLog}(P, Q, g, y)$ and output x where PLog is the algorithm given in Lemma 7.

Theorem 2 *If the m-ad-DRSA assumption holds with respect to* IGen*, then the above scheme is an $(n, n - (\ell_{p'} + \ell_{q'} + (M_{\ell_B} - m)\ell_B))$-ad-LTDF.*

Then the following corollary follows by combining the above theorem and Theorem 1.

Corollary 1 *If the factoring assumption holds with respect to* IGen *for the parameter setting given in Sec. 2.4, then there exists an ad-LTDF.*

Proof. (of Corollary 1.) Recall that we set $\ell_{p'} = \ell_{q'} = O(\lambda)$, $\ell_B = \lfloor 4 \log \lambda \rfloor$, $t_p = t_q = O(\lambda^3 / \log \lambda)$ (then we have $\ell_N \approx \ell_{p'} + \ell_{q'} + t\ell_B = O(\lambda^3)$) and $d := t/4$. We let $m := \lfloor M_{\ell_B} + 1 - (1/2 + c)\frac{\ell_N}{(\ell_B - 1)} \rfloor$ for a constant $c < 1/4$. Then we have $(M_{\ell_B} - m + 1)(\ell_B - 1) \geq (1/2 + c)\ell_N$ and therefore the m-ad-DRSA assumption holds under the factoring assumption by Theorem 1. Then we prove that the above ad-LTDF for this m is non-trivial, i.e., we have $n -$

$(\ell_{p'} + \ell_{q'} + (M_{\ell_B} - m)\ell_B) > 0$. Since we have $m \approx M_{\ell_B} - (1/2 + c)\frac{\ell_N}{(\ell_B-1)}$, we have $n - (\ell_{p'} + \ell_{q'} + (M_{\ell_B} - m)\ell_B) \approx (t-d)\ell_B - (\ell_{p'} + \ell_{q'} + (1/2+c)\frac{\ell_N \ell_B}{(\ell_B-1)}) \approx (1/4 - c)t\ell_B - (3/2 + c)(\ell_{p'} + \ell_{q'}) > 0$ for sufficiently large λ since $t\ell_B = O(\lambda^3)$ and $\ell_{p'} + \ell_{q'} = O(\lambda)$. Thus the obtained ad-LTDF for this m is non-trivial.

Remark 6 *If we set $\ell_{p'} = \ell_{q'} = 160$, $\ell_B = 15$, $t = 64$, $d = 7$ and $\ell_N = 2420$ as given in Sec. 2.4, and $c = 1/20$ then by setting $m := \lfloor M_{\ell_B} + 1 - (1/2+c)\frac{\ell_N}{(\ell_B-1)} \rfloor$, the obtained scheme is a $(1848, 103)$-ad-LTDF. If better lossiness is required, then one may set t larger (as long as factorizing N is hard).*

Then we prove Theorem 2.

Proof. (of Theorem 2)
Correctness. If g is generated by $\mathsf{SampleInj}$, then it is a random element of $\mathbb{QR}_N$. Thus $\mathsf{Inversion}((P,Q), g, \mathsf{Evaluation}(N, \sigma, x)) = \mathsf{Inversion}((P,Q), g, g^x) = x$ holds by the correctness of PLog given in Lemma 7.

Lossiness. Next, we prove that the above construction satisfies $(n, n - (\ell_{p'} + \ell_{q'} + (M_{\ell_B} - m)\ell_B))$-lossiness. Let σ be a function description generated by $\mathsf{SampleLossy}(N, I = \{p_1, \ldots, p_m\})$. What we should prove is that the image size of f_σ is at most $2^{\ell_{p'} + \ell_{q'} + (M_{\ell_B} - m)\ell_B}$. There exists $g \in \mathbb{QR}_N$ such that $\sigma = g^{p_1 \cdots p_m}$, and thus any output of f_σ is an element of the group $S := \{h^{p_1 \cdots p_m} : h \in \mathbb{QR}_N\}$. We consider the order of S. S is a subgroup of $\mathbb{QR}_N = G \times G^\perp$ and $p_1 \ldots p_m$ is coprime to $\mathrm{ord}(G) = p'q'$. Therefore there exists a subgroup $S^\perp$ of $G^\perp$ such that $S = G \times S^\perp$. We can see that $\mathrm{ord}(S^\perp)$ is the product of some distinct ℓ_B-bit primes and coprime to $p_1 \ldots p_m$ by the definition. Therefore that is the product of at most $M_{\ell_B} - m$ such primes, and can be bounded by $2^{(M_{\ell_B} - m)\ell_B}$. Therefore the order of S is at most $2^{\ell_{p'} + \ell_{q'} + (M_{\ell_B} - m)\ell_B}$.

Indistinguishability between injective and lossy functions. This immediately follows from the m-ad-DRSA assumption. Indeed, clearly we have $P_{\mathsf{inj}} = P_0$ and $P_{\mathsf{lossy}} = P_1$ where P_0 and P_1 are defined in Def. 3, and the m-ad-DRSA assumption requires $|P_0 - P_1| < \epsilon(\lambda)$ for sufficiently large λ.

As an analogue of all-but-one lossy trapdoor function defined in [27], we can define adversary-dependent all-but-one lossy trapdoor functions (ad-ABO). The definition and constructions are given in Appendix B.

5 Applications

Here we discuss applications of ad-LTDFs. As mentioned before, ad-LTDFs can replace LTDFs in many applications. Informally, ad-LTDFs can replace LTDFs if a lossy function is used only in the security proof and not used in the real protocol. In such cases, a lossy function may depend on an adversary that tries to distinguish it from an injective function since an adversary is firstly fixed in security proofs. As a result, we can immediately obtain a collision resistant hash function [27], a CPA secure PKE scheme [27] and a DE scheme [5] based on ad-LTDFs by simply replacing LTDFs by ad-LTDFs. Among them, by using our ad-LTDF based on the factoring assumption given in Sec. 4, we obtain the first DE scheme that satisfies the PRIV security for block-sources defined in [5] under the factoring assumption.

5.1 Collision Resistant Hash Function

Here, we give an analogue of the collision resistant hash function in [27] based on ad-LTDFs. In fact, our scheme is obtained by simply replacing LTDFs in the scheme in [27] by ad-LTDFs. The concrete construction is as follows. Let $(\mathsf{ParamsGen}, \mathsf{SampleInj}, \mathsf{SampleLossy}, \mathsf{Evaluation}, \mathsf{Inversion})$ be an (n, k)-ad-LTDF and $\mathcal{H}$ be a family of pairwise independent hash functions from $\{0,1\}^k$ to $\{0,1\}^{\kappa n}$ where $\kappa := 2\rho + \delta$, $\rho < 1/2$ is a constant that satisfies $n - k \le \rho n$ and δ is some constant in $(0, 1 - 2\rho)$,

$\mathsf{Gen}_{\mathsf{crh}}(1^\lambda)$**:** Run $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleInj}(PP)$, and choose $H \xleftarrow{\$} \mathcal{H}$. Output a function description $h := (H, PP, \sigma)$.

$\mathsf{Eval}_{\mathsf{crh}}((H, PP, \sigma), x)$**:** Compute $H(\mathsf{Evaluation}(PP, \sigma, x))$ and output it.

Theorem 3 *The above hash function is collision resistant.*

We omit the proof since this can be proven by modifying the proof in [27] in a similar way as in Sec. 5.3.

5.2 CPA Secure Public Key Encryption

Here, we give an analogue of the CPA secure PKE scheme in [27] based on ad-LTDFs. In fact, our scheme is obtained by simply replacing LTDFs in the scheme in [27] by LTDFs. The concrete construction is as follows.

Let $(\mathsf{ParamsGen}, \mathsf{SampleInj}, \mathsf{SampleLossy}, \mathsf{Evaluation}, \mathsf{Inversion})$ be an (n, k)-ad-LTDF and $\mathcal{H}$ be a family of pairwise independent hash functions from $\{0,1\}^n$ to $\{0,1\}^\ell$, where $\ell \le k - 2\log(1/\delta)$ for some negligible δ. The construction of our scheme $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is as follows.

Key generation : $\mathsf{Gen}(1^\lambda)$ generates $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleInj}(PP)$. It also chooses a hash function $H \xleftarrow{\$} \mathcal{H}$. It outputs a public key $PK = (PP, \sigma, H)$ and a secret key $SK = (SP, H)$.

Encryption : Enc takes as input a public key $PK = (PP, \sigma, H)$ and a message $msg \in \{0,1\}^\ell$. It chooses $x \xleftarrow{\$} \{0,1\}^n$, sets $C_1 := \mathsf{Evaluation}(PP, \sigma, x)$ and $C_2 := msg \oplus H(x)$ and outputs $C = (C_1, C_2)$

Decryption : Dec takes as input a secret key $SK = (SP, H)$ and a ciphertext $C = (C_1, C_2)$, computes $x := \mathsf{Inversion}(SP, \sigma, C_1)$ and $msg := C_2 \oplus H(x)$, and outputs msg.

Theorem 4 *The above scheme is CPA secure.*

This can be proven by modifying the proof in [27] in a similar way as in Sec. 5.3. The proof is given in the Appendix A.1.

Remark 7 *If we use ad-ABO given in the full version, we can construct CCA secure PKE scheme similarly as in [27].*

5.3 Deterministic Encryption

Here, we construct a DE scheme based on ad-LTDFs. The construction is a simple analogue of the scheme in [5] based on LTDFs. Indeed, our scheme is obtained by simply replacing LTDFs by ad-LTDFs in their scheme. The concrete construction is as follows. Let $(\mathsf{ParamsGen}, \mathsf{SampleInj}, \mathsf{SampleLossy}, \mathsf{Evaluation}, \mathsf{Inversion})$ be an (n, k)-ad-LTDF and $\mathcal{H}$ be a family of pairwise independent permutations on $\{0,1\}^n$, where $u \geq n - k + 2\log(1/\delta) - 2$ holds for some negligible δ. The construction of our scheme $\mathsf{DE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is as follows.

$\mathsf{Gen}(1^\lambda)$**:** Generate $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleInj}(PP)$ and choose $H \xleftarrow{\$} \mathcal{H}$. Output a public key $PK = (PP, \sigma, H)$ and a secret key $SK = (SP, \sigma)$.

$\mathsf{Enc}(PK = (PP, \sigma, H), msg)$**:** Compute $C \leftarrow \mathsf{Evaluation}(PP, \sigma, H(msg))$ and output C.

$\mathsf{Dec}(SK, C)$**:** Compute $msg' \leftarrow \mathsf{Inversion}(SP, \sigma, C)$ and $msg := H^{-1}(msg')$ and output msg.

Theorem 5 *The above scheme is PRIV1-IND-CPA secure deterministic encryption for (u, n)-sources.*

Proof. Assume that the above scheme is not PRIV1-IND-CPA secure. There exists (u, n)-sources M_0, M_1 and a PPT adversary $\mathcal{A}$ such that $\mathsf{Adv}_{\mathcal{A},\mathsf{DE}}^{\mathsf{PRIV-IND-CPA}}(\lambda)$ is non-negligible. Then there exist a polynomial poly such that for infinitely many λ, $\mathsf{Adv}_{\mathcal{A},\mathsf{DE}}^{\mathsf{PRIV-IND-CPA}}(\lambda) > 1/\mathsf{poly}(\lambda)$ holds. We consider the following sequence of games.

$\mathsf{Game}\ 1$: This game is the original PRIV1-IND-CPA game with respect to M_0, M_1 and $\mathcal{A}$. That is, a challenger computes $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleInj}(PP)$, chooses $H \leftarrow \mathcal{H}$, sets $PK := (PP, \sigma, H)$, chooses $b \xleftarrow{\$} \{0,1\}$, $msg^* \xleftarrow{\$} M_b$ and computes $C^* \leftarrow \mathsf{Evaluation}(PP, \sigma, H(msg^*))$. $\mathcal{A}$ is given (PK, C^*) and outputs b'.

$\mathsf{Game}\ 2$: This game is the same as the previous game except that σ is generated by $\mathsf{SampleLossy}(PP, I)$, where intuitively, I is an index such that "it is difficult to distinguish an injective function from a lossy function with index I for $\mathcal{A}$". To describe this precisely, we consider the following PPT algorithm $\mathcal{B}$.

$\mathcal{B}(PP, \sigma)$: Choose $H \xleftarrow{\$} \mathcal{H}$, $b \xleftarrow{\$} \{0,1\}$, $msg^* \xleftarrow{\$} M_b$, set $PK := (PP, \sigma, H)$, compute $C^* \leftarrow \mathsf{Evaluation}(PP, \sigma, H(msg^*))$, run $b' \leftarrow \mathcal{A}(PK, C^*)$ and output 1 if $b = b'$, and otherwise 0.

Let $\mathcal{S}_{\mathcal{B},1/(2\mathsf{poly})}$ be the algorithm that is assumed to exists in the definition of ad-LTDFs. (Note that $\mathcal{B}$ is a PPT algorithm and $1/(2\mathsf{poly})$ is noticeable.) In this game, we let $I \leftarrow \mathcal{S}_{\mathcal{B},1/(2\mathsf{poly})}(PP)$ and $\sigma \leftarrow \mathsf{SampleLossy}(PP, I)$.

$\mathsf{Game}\ 3$: This game is the same as the previous game except that a challenge ciphertext is set as $C^* \leftarrow \mathsf{Evaluation}(PP, \sigma, H(U))$ where $U \in \{0,1\}^n$ is a uniformly random string.

Let T_i be the event that $b = b'$ in $\mathsf{Game}\ i$. Clearly we have $|\Pr[T_1] - 1/2| = \mathsf{Adv}^{\mathsf{PRIV-IND-CPA}}_{\mathcal{A},\mathsf{DE}}(\lambda)$. Then we prove the following lemmas.

Lemma 9 *For sufficiently large any λ, we have $|\Pr[T_2] - \Pr[T_1]| \le 1/(2\mathsf{poly}(\lambda))$.*

Proof. By the definition of an adversary-dependent lossy trapdoor function, if we let

$$P_{\mathsf{inj}} := \Pr\left[1 \leftarrow \mathcal{B}(PP,\sigma) : \begin{array}{c}(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)\\ \sigma \leftarrow \mathsf{SampleInj}(PP)\end{array}\right]$$
$$P_{\mathsf{lossy}} := \Pr\left[1 \leftarrow \mathcal{B}(PP,\sigma) : \begin{array}{c}(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)\\ I \leftarrow \mathcal{S}_{\mathcal{B},1/(2\mathsf{poly})}(PP)\\ \sigma \leftarrow \mathsf{SampleLossy}(PP, I)\end{array}\right]$$

then we have $|P_{\mathsf{inj}} - P_{\mathsf{lossy}}| \le 1/(2\mathsf{poly})$ for sufficiently large λ. It is clear that $P_{\mathsf{inj}} = \Pr[T_1]$ and $P_{\mathsf{lossy}} = \Pr[T_2]$ holds. Therefore the lemma follows.

Lemma 10 *We have $|\Pr[T_3] - \Pr[T_2]| \le \delta$.*

In Lemma 3, we let $f := \mathsf{Evaluation}(PP, \sigma, \cdot)$, $X := msg^*$ and $Y := (PP, \sigma)$. Then by the lossiness, $|S| \le 2^{n-k}$ holds where S is the range of f. By the definition of (u,n)-sources, we have $\widetilde{H}_\infty(X|Y) \ge u$ and $u \ge n - k + 2\log(1/\delta) - 2 \ge |S| + 2\log(1/\delta) - 2$. By Lemma 3, the statistical distance between $(C^*, H, (PP, \sigma))$ in $\mathsf{Game}\ 2$ and that in $\mathsf{Game}\ 3$ is at most δ. Thus the lemma follows.

Lemma 11 *We have $\Pr[T_3] = 1/2$.*

Proof. In $\mathsf{Game}\ 3$, $\mathcal{A}$ is given no information about b. Therefore the probability that $\mathcal{A}$ can correctly guess b is $1/2$.

By combining these lemmas, for all sufficiently large λ, we have $|\Pr[T_1] - 1/2| \le 1/(2\mathsf{poly}(\lambda)) + \delta$, equivalently, $\mathsf{Adv}^{\mathsf{PRIV-IND-CPA}}_{\mathcal{A},\mathsf{DE}}(\lambda) \le 1/(2\mathsf{poly}(\lambda)) + \delta$. On the other hand, we assumed, $\mathsf{Adv}^{\mathsf{PRIV-IND-CPA}}_{\mathcal{A},\mathsf{DE}}(\lambda) > 1/\mathsf{poly}(\lambda)$ for infinitely many λ. Combining these two inequalities, we have $1/(2\mathsf{poly}(\lambda)) < \delta$ for infinitely many λ, which contradicts to that δ is negligible. Therefore there does not exist a PPT adversary that breaks the scheme.

Remark 8 *If we use ad-ABO given in the full version, we can construct PRIV1-IND-CCA secure DE scheme similarly as in [5].*

6 CCA Secure PKE with Short Ciphertext

In this section, we construct a CCCA secure KEM under the m-ad-DRSA assumption. By Theorem 1, under certain condition, this scheme is CCCA secure under the factoring assumption w.r.t. SS moduli. By setting a parameter appropriately, we obtain a PKE scheme whose ciphertext overhead is minimum among schemes that are CCA secure under the factoring assumption by combining our KEM and an authenticated symmetric key encryption scheme.

6.1 Construction

Idea of our construction. Since the m-ad-DRSA assumption is a type of subgroup decision assumptions, we can consider an "adversary-dependent version" of hash proof systems as in [8], where it is shown that a hash proof system can be constructed based on any subgroup decision assumption. Then we construct a KEM similarly as in [16], where the authors constructed a CCCA secure KEM based on a hash proof system. Though our construction is based on the above idea, for clarity, we give a direct construction of our KEM rather than defining the "adversary-dependent version" of hash proof systems.

The construction of our scheme $\mathsf{KEM}_{\mathsf{CCCA}}$ is as follows. Let IGen be a PPT algorithm that generates (ℓ_B, t_p, t_q)-SS RSA modulus, $\mathcal{H}$ be a family of pairwise independent hash functions from $(\mathbb{Z}_N^*)^n$ to $\{0,1\}^\lambda$ where $n := \lceil \frac{(2\ell_N+1)\lambda}{\ell_B-1} \rceil$, and $h : G \to \{0,1\}^\lambda$ be a target collision resistant hash function. For simplicity, we assume that the KEM key length is equal to the security parameter λ.

$\mathsf{Gen}(1^\lambda)$: Generate $(N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda)$. Choose $H \xleftarrow{\$} \mathcal{H}$, $g \xleftarrow{\$} \mathbb{QR}_N$ and $x_{i,j}^{(k)} \xleftarrow{\$} [(N-1)/4]$ and set $X_{i,j}^{(k)} := g^{x_{i,j}^{(k)}}$ for $i = 1, \ldots, \lambda$, $j = 1, \ldots, n$ and $k = 0, 1$. Output $PK := (N, h, H, \{X_{i,j}^{(k)}\}_{i\in[\lambda], j\in[n], k\in\{0,1\}})$ and $SK := (\{x_{i,j}^{(k)}\}_{i\in[\lambda], j\in[n], k\in\{0,1\}}, PK)$.

$\mathsf{Enc}(PK)$: Choose $r \xleftarrow{\$} [(N-1)/4]$, compute $C := g^r$, $t := h(C)$ and $K := H((\prod_{i=1}^{\lambda} X_{i,1}^{(t_i)})^r, \ldots, (\prod_{i=1}^{\lambda} X_{i,n}^{(t_i)})^r)$ where t_i denotes the i-th bit of t. Output (C, K).

$\mathsf{Dec}(SK, C)$: Compute $t := h(C)$ and $K := H(C^{\sum_{i=1}^{\lambda} x_{i,1}^{(t_i)}}, \ldots, C^{\sum_{i=1}^{\lambda} x_{i,n}^{(t_i)}})$ where t_i denotes the i-th bit of t, and output K.

6.2 Security

Theorem 6 *If m-ad-DRSA assumption holds with respect to* IGen *and $(\ell_B - 1)(t_p + t_q + m - M_{\ell_B}) \geq \lambda$ holds, then* $\mathsf{KEM}_{\mathsf{CCCA}}$ *is CCCA secure.*

Corollary 2 *If the factoring assumption holds with respect to* IGen *for the parameter setting given in 2.4, then* $\mathsf{KEM}_{\mathsf{CCCA}}$ *is CCCA secure for $n = O(\lambda^4/\log(\lambda))$.*

Proof. (of Corollary 2) Let $m := \lfloor M_{\ell_B} + 1 - (1/2+c)\frac{\ell_N}{(\ell_B-1)} \rfloor$ for a constant $c < 1/4$. Then we have $(M_{\ell_B} - m + 1)(\ell_B - 1) \geq (1/2+c)\ell_N$ and therefore the m-ad-DRSA assumption holds under the factoring assumption by Theorem 1. Moreover, we have $(\ell_B - 1)(t_p + t_q + m - M_{\ell_B}) = O(\lambda^3)$ if we use the parameter setting given in Sec. 2.4. Thus the obtained scheme is CCCA secure under the factoring assumption. □

Theorem 6 can be proven almost similarly as the security of the CCCA secure KEM based on a hash proof system in [16]. However, for a technical reason, we need the following variant of the leftover hash lemma unlike in [16]. Specifically, in the leftover hash lemma (Lemma 2), a random variable X should be independent from H. On the other hand, in our proof, we need a variant in which a random variable X may depend on H. The following lemma states that this is possible with the loss of the number of possible random variables X. We note that this idea is already used in some existing works [35, 32]. This lemma is necessary because in our proof, we set X to be a decryption query, which is chosen by an adversary after seeing a public key which includes a pairwise independent hash function H.

Lemma 12 *Let $\mathcal{X}$ be a set of random variables X on $\{0,1\}^{n_1}$ such that $H_\infty(X) \geq n_2 + 2\log(1/\delta)$, and $\mathcal{H}$ be a family of pairwise independent hash functions from $\{0,1\}^{n_1}$ to $\{0,1\}^{n_2}$. Then for any computationally unbounded algorithm $\mathcal{F}$, which is given $H \in \mathcal{H}$ and outputs a description of a distribution $X \in \mathcal{X}$, we have $\Delta((H(X), H), (U, H)) \leq |\mathcal{X}|\delta$ where $H \xleftarrow{\$} \mathcal{H}$ and $X \leftarrow \mathcal{F}(H)$.*

The proof is given in the full version.

Then we give the proof of Theorem 6.

Proof. (of Theorem 6) Assume that there exists a valid PPT adversary $\mathcal{A}$ that breaks the CCCA security of the above scheme. Then there exists a polynomial poly such that $\mathsf{Adv}^{\mathsf{CCCA}}_{\mathcal{A},\mathsf{KEM}_{\mathsf{CCCA}}}(\lambda) > 1/\mathsf{poly}(\lambda)$ for infinitely many λ. We consider the following sequence of games.

Game 1 : This game is the original CCCA game of $\mathsf{KEM}_{\mathsf{CCCA}}$ for $\mathcal{A}$. That is, a challenger $\mathcal{C}$ generates $(N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda)$, chooses $H \xleftarrow{\$} \mathcal{H}$, $g \xleftarrow{\$} \mathbb{QR}_N$ and $x^{(k)}_{i,j} \xleftarrow{\$} [(N-1)/4]$ and sets $X^{(k)}_{i,j} := g^{x^{(k)}_{i,j}}$ for $i = 1, \ldots, \lambda$, $j = 1, \ldots, n$ and $k = 0, 1$ and sets $PK := (N, h, H, \{X^{(k)}_{i,j}\}_{i\in[\lambda], j\in[n], k\in\{0,1\}})$. Then it chooses $b \xleftarrow{\$} \{0,1\}$ and $r^* \xleftarrow{\$} [(N-1)/4]$, and computes $C^* := g^{r^*}$, $t^* := h(C^*)$ and $K^* := H((\prod_{i=1}^{\lambda} X^{(t^*_i)}_{i,1})^{r^*}, \ldots, (\prod_{i=1}^{\lambda} X^{(t^*_i)}_{i,n})^{r^*})$ where t^*_i denotes the i-th bit of t^* if $b = 1$ and $K^* \xleftarrow{\$} \{0,1\}^\lambda$ otherwise. Then it gives (PK, C^*, K^*) to $\mathcal{A}$. In the game, $\mathcal{A}$ can query pairs of ciphertexts and predicates to an oracle $\mathcal{O}_{\mathsf{Dec}}$. When $\mathcal{A}$ queries (C, pred), $\mathcal{O}_{\mathsf{Dec}}$ computes $K \leftarrow \mathsf{Dec}(SK, C)$ and returns K to $\mathcal{A}$ if $C \neq C^*$ and $\mathsf{pred}(K) = 1$, and otherwise $\perp$. Finally, $\mathcal{A}$ outputs a bit b'.

Game 2 : This game is the same as the previous game except that K^* is set differently if $b = 1$. Specifically, it is set as $K^* := H(C^{*\sum_{i=1}^{\lambda} x^{(t^*_i)}_{i,1}}, \ldots, C^{*\sum_{i=1}^{\lambda} x^{(t^*_i)}_{i,n}})$ if $b = 1$.

Game 3 : This game is the same as the previous game except that C^* is set differently. Specifically, it is uniformly chosen from $\mathbb{QR}_N$.

Game 4 : This game is the same as the previous game except that g is uniformly chosen from a subgroup S of $\mathbb{QR}_N$, which is defined as follows. First, we define a PPT algorithm $\mathcal{B}$ as follows.

$\mathcal{B}(N, g)$**:** Choose $H \xleftarrow{\$} \mathcal{H}$, $x_{i,j}^{(k)} \xleftarrow{\$} [(N-1)/4]$ and set $X_{i,j}^{(k)} := g^{x_{i,j}^{(k)}}$ for $i \in [\lambda]$, $j \in [n]$ and $k = 0, 1$ and $PK := (N, h, H, \{X_{i,j}^{(k)}\}_{i\in[\lambda],j\in[n],k\in\{0,1\}})$, choose $C^* \xleftarrow{\$} \mathbb{QR}_N$ and $b \xleftarrow{\$} \{0,1\}$, and set $K^* := H(C^{*\sum_{i=1}^{\lambda} x_{i,1}^{(t_i^*)}}, \dots, C^{*\sum_{i=1}^{\lambda} x_{i,n}^{(t_i^*)}})$ where $t^* := h(C^*)$ and t_i^* is the i-th bit of t^* if $b = 1$, and $K^* \xleftarrow{\$} \{0,1\}^{\ell}$ otherwise. Run $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{Dec}}}(PK, C^*, K^*)$ and output b'. We note that $\mathcal{B}$ can simulate $\mathcal{O}_{\mathsf{Dec}}$ for $\mathcal{A}$ since it knows $SK = (\{x_{i,j}^{(k)}\}_{i\in[\lambda],j\in[n],k\in\{0,1\}}, PK)$.

Let $\mathcal{S}_{\mathcal{B},\mathsf{poly}/2}$ be the algorithm that is assumed to exist in the definition of m-ad-DRSA assumption. Note that this algorithm actually exists since $\mathcal{B}$ is a PPT algorithm and $\mathsf{poly}/2$ is noticeable. Then we define the subgroup S as follows: We run $\{p_1, \dots, p_m\} \leftarrow \mathcal{S}_{\mathcal{B},\mathsf{poly}/2}$ and define $S := \{h^{p_1,\dots,p_m} : h \in \mathbb{QR}_N\}$.

Game 5 : This game is the same as the previous game except that the decryption oracle $\mathcal{O}_{\mathsf{Dec}}$ is replaced with an alternative decryption oracle $\mathcal{O}_{\mathsf{Dec}'}$ that works as follows: $\mathcal{O}_{\mathsf{Dec}'}$, given C and pred, computes $t := h(C)$ and returns $\perp$ if $t = t^*$. Otherwise it computes $K := H(C^{\sum_{i=1}^{\lambda} x_{i,1}^{(t_i)}}, \dots, C^{\sum_{i=1}^{\lambda} x_{i,n}^{(t_i)}})$ and outputs K if $\mathsf{pred}(K) = 1$, and otherwise $\perp$.

Game 6 : This game is the same as the previous game except that $x_{i,j}^{(k)}$ is set differently. Specifically, it is uniformly chosen from $\mathrm{ord}(\mathbb{QR}_N)$ instead of from $[(N-1)/4]$ for $i = 1, \dots, \lambda$, $j = 1, \dots, n$ and $k = 0, 1$.

Game 7 : This game is the same as the previous game except that the decryption oracle $\mathcal{O}_{\mathsf{Dec}'}$ is replaced with an alternative decryption oracle $\mathcal{O}_{\mathsf{Dec}''}$ that works as follows: $\mathcal{O}_{\mathsf{Dec}''}$, given C and pred, computes $t := h(C)$ and returns $\perp$ if $t = t^*$ or $C \notin S$, where S is the group defined in Game 4. Otherwise it computes $K := H(C^{\sum_{i=1}^{\lambda} x_{i,1}^{(t_i)}}, \dots, C^{\sum_{i=1}^{\lambda} x_{i,n}^{(t_i)}})$and outputs K if $\mathsf{pred}(K) = 1$, and otherwise $\perp$.

Game 8 : This game is the same as the previous game except that K^* is always an independently random string.

Let T_i be the event that $b = b^*$ holds in Game i. Then clearly we have $\mathsf{Adv}^{\mathsf{CCCA}}_{\mathcal{A},\mathsf{PKE}_{\mathsf{CCCA}}} = |\Pr[T_1] - 1/2|$. First, we prove that the group S defined in Game 4 is a proper subgroup of $\mathbb{QR}_N$. Moreover, we prove that $\mathrm{ord}(S)/\mathrm{ord}(\mathbb{QR}_N) \leq 2^{-\lambda}$ holds. By the definition of SS moduli, $\mathrm{ord}(\mathbb{QR}_N)$ has $t_p + t_q$ distinct prime factors $p'_1, \dots, p'_{t_p+t_q}$ of ℓ_B-bit. Since the number of the all ℓ_B-bit primes is M_{ℓ_B}, there exist at least $t_p + t_q + m - M_{\ell_B}$ distinct primes contained in both $\{p'_1, \dots, p'_{t_p+t_q}\}$ and $\{p_1, \dots, p_m\}$. We denote those primes by $p''_1, \dots p''_{t_p+t_q+m-M_{\ell_B}}$. Those primes cannot be a factor of $\mathrm{ord}(S)$ since $S = \{h^{p_1\cdots p_m} : h \in \mathbb{QR}_N\}$ by the definition whereas they are a factor of $\mathrm{ord}(\mathbb{QR}_N)$. Thus $\mathrm{ord}(S)/\mathrm{ord}(\mathbb{QR}_N) \leq \frac{1}{p''_1\cdots p''_{t_p+t_q+m-M_{\ell_B}}}$ $\leq 2^{-(t_p+t_q+m-M_{\ell_B})(\ell_B-1)} \leq 2^{-\lambda}$. Then we prove the following lemmas.

Lemma 13 $\Pr[T_2] = \Pr[T_1]$ *holds.* □

Proof. The modification between Game 1 and 2 is only conceptual. □

Lemma 14 $|\Pr[T_3] - \Pr[T_2]|$ *is negligible.*

Proof. This follows from the fact that the statistical distance between the uniform distributions on $[(N - 1/4)]$ and $[\mathrm{ord}(\mathbb{QR}_N)]$ are negligible. □

Lemma 15 *We have* $|\Pr[T_4] - \Pr[T_3]| \leq 1/(2\mathsf{poly})$ *for sufficiently large* λ.

Proof. This follows immediately from the definition of m-ad-DRSA assumption. □

Lemma 16 *If* h *is collision resistant, then* $|\Pr[T_5] - \Pr[T_4]|$ *is negligible.*

Proof. From the view of $\mathcal{A}$, Game 4 and 5 may differ only if $\mathcal{A}$ makes a query (C, pred) such that $h(C) = t^*$. If $\mathcal{A}$ makes such a query, then this means that it finds a collision of h. □

Lemma 17 $|\Pr[T_6] - \Pr[T_5]|$ *is negligible.*

Proof. This follows from the fact that the statistical distance between the uniform distributions on $[(N - 1/4)]$ and $[\mathrm{ord}(\mathbb{QR}_N)]$ are negligible. □

Lemma 18 $|\Pr[T_7] - \Pr[T_6]|$ *is negligible.*

Proof. Let q be an upper bound of the number of decryption queries $\mathcal{A}$ makes. We consider hybrids $\mathsf{H}_0, \ldots, \mathsf{H}_q$ that are defined as follows. A hybrid H_ℓ is the same as Game 6 except that the oracle to which $\mathcal{A}$ accesses works similarly as $\mathcal{O}_{\mathsf{Dec}''}$ for the first ℓ queries, and similarly as $\mathcal{O}_{\mathsf{Dec}'}$ for the rest of queries. Let $T_{6,\ell}$ be the event that $b = b'$ holds in the hybrid H_ℓ. Clearly, We have $\Pr[T_{6,0}] = \Pr[T_6]$ and $\Pr[T_{6,\ell}] = \Pr[T_7]$. Let F_ℓ be the event that $\mathcal{O}_{\mathsf{Dec}''}$ returns $\perp$ for $\mathcal{A}$'s ℓ-th query $(C_\ell, \mathsf{pred}_\ell)$ but $\mathcal{O}_{\mathsf{Dec}'}$ does not return $\perp$ for it. That is, F_ℓ is the event that $C_\ell \in \mathbb{QR}_N \setminus S$, $t \neq t^*$ and $\mathsf{pred}(K_\ell) = 1$ hold where $K_\ell := H(C_\ell^{\sum_{i=1}^{\lambda} x_{i,1}^{(t_i)}}, \ldots, C_\ell^{\sum_{i=1}^{\lambda} x_{i,n}^{(t_i)}})$, $t := h(C_\ell)$ and t_i denotes the i-th bit of t. Unless F_ℓ occurs, hybrids H_ℓ and $\mathsf{H}_{\ell-1}$ are exactly the same from the view of $\mathcal{A}$. Therefore we have $|\Pr[T_{6,\ell}] - \Pr[T_{6,\ell-1}]| \leq \Pr[F_\ell]$. Let view_ℓ be the view from $\mathcal{A}$ before it is given the response for its ℓ-th query. That is, view_ℓ consists of PK, C^*, K^*, C_ℓ and decryption queries and responses for them before the ℓ-th query. We prove the following claim.

Claim 3 *If* $C_\ell \in \mathbb{QR}_N \setminus S$ *and* $t \neq t^*$*, then* K_ℓ *is distributed almost uniformly on* $\{0,1\}^\lambda$ *from the view of* $\mathcal{A}$ *in the hybrids* $\mathsf{H}_{\ell-1}$ *and* H_ℓ. *More precisely, we have* $\Delta((K_\ell, \mathsf{view}_\ell), (U, \mathsf{view}_\ell)) \leq 2^{-\lambda}$ *where* $U \xleftarrow{\$} \{0,1\}^\lambda$.

Assume this claim is true. Then we prove that $\Pr[F_\ell]$ is negligible for any $\ell \in [q]$. Since $\mathcal{A}$ is valid, pred is non-trivial. That is, for independently uniform U, $\Pr[\mathsf{pred}_i(U) = 1]$ is negligible. By Claim 3, if $C_\ell \in \mathbb{QR}_N \setminus S$ and $t \neq t^*$, then we have $\Delta((K_\ell, \mathsf{view}), (U, \mathsf{view})) \leq 2^{-\lambda}$ where $U \xleftarrow{\$} \{0,1\}^\lambda$. Therefore $\Pr[\mathsf{pred}(K_\ell) = 1]$ is negligible and thus $\Pr[F_\ell]$ is negligible. Thus $|\Pr[T_7] - \Pr[T_6]|$ is negligible by the hybrid argument. What is left is to prove Claim 3.

Proof. (of Claim 3) Since we assumed $t \neq t^*$, there exists $i \in [\lambda]$ such that $t_i \neq t_i^*$. We denote minimum such i by i^*. Since $C \in \mathbb{QR}_N \setminus S$ and S is a proper subgroup of $\mathbb{QR}_N$, there exists an ℓ_B-bit prime $\bar{p}$ that divides $\mathrm{ord}(C)$ but does not divide $\mathrm{ord}(S)$. Here, we claim that the decryption oracle before ℓ-th query can be simulated by using $\{x_{i,j}^{(k)} \mod \mathrm{ord}(S)\}_{i\in[\lambda],j\in[n],k\in\{0,1\}}$ and PK. This can be seen by that the oracle immediately returns $\perp$ for a query (C, pred) such that $C \notin S$. If we define $\mathsf{view}'_\ell := (\overline{PK}, C^*, K^*, C_\ell, \{x_{i,j}^{(k)} \mod \mathrm{ord}(S)\}_{i\in[\lambda],j\in[n],k\in\{0,1\}})$ where $\overline{PK}$ denotes a public key except H, then we have $\Delta((K_\ell, \mathsf{view}_\ell), (U, \mathsf{view}_\ell)) \leq \Delta((K_\ell, H, \mathsf{view}'_\ell), (U, H, \mathsf{view}'_\ell))$. Thus it suffices to show that conditioned on any fixed value of view'_ℓ, $\Delta((K_\ell, H), (U_\ell, H)) \leq 2^{-\lambda}$ holds. One can see that view'_ℓ does not depend on $(x_{i^*,j}^{(t_{i^*})} \mod \bar{p})$ at all for $j \in [n]$: $\overline{PK}$ does not depend on $(x_{i^*,j}^{(t_{i^*})} \mod \bar{p})$ since $g \in S$ by the modification from Game 3 to 4. (C^*, K^*) does not depend on $(x_{i^*,j}^{(t_{i^*})} \mod \bar{p})$ since we assumed $t_{i^*} \neq t_{i^*}^*$ and thus $x_{i^*,j}^{(t_{i^*})}$ is not used for generating K^*. $\{x_{i,j}^{(k)} \mod \mathrm{ord}(S)\}_{i\in[\lambda],j\in[n],k\in\{0,1\}}$ does not depend on $(x_{i^*,j}^{(t_{i^*})} \mod \bar{p})$ since $\mathrm{ord}(S)$ is coprime to $\bar{p}$. Thus conditioned on any value of view'_ℓ, $(x_{i^*,j}^{(t_{i^*})} \mod \bar{p})$ is distributed uniformly for all $j \in [n]$. Therefore we have $H_\infty(C_\ell^{\sum_{i=1}^\lambda x_{i,1}^{(t_i)}}, \ldots, C_\ell^{\sum_{i=1}^\lambda x_{i,n}^{(t_i)}} | \mathsf{view}'_\ell) \geq n \log \bar{p} \geq n(\ell_B - 1) \geq \lambda + 2\ell_N\lambda$. Here, we use Lemma 12: We set $\mathcal{X} := \{X_C\}_{C \in \mathbb{QR}_N \setminus S}$ where X_C denotes a random variable that is distributed as $(C^{\sum_{i=1}^\lambda x_{i,1}^{(t_i)}}, \ldots, C^{\sum_{i=1}^\lambda x_{i,n}^{(t_i)}})$ conditioned on view'_ℓ, $\delta := 2^{-\ell_N\lambda}$, and $\mathcal{F}$ as an algorithm that simulates the game between $\mathcal{A}$ and the challenger and outputs X_{C_ℓ} where C_ℓ is $\mathcal{A}$'s ℓ-th decryption query. Then we have $\Delta((K_\ell, H), (U, H)) \leq |\mathbb{QR}_N \setminus S| 2^{-\ell_N\lambda} \leq 2^{-\lambda}$ where $U \xleftarrow{\$} \{0,1\}^\lambda$, conditioned on any fixed value of view'_ℓ. Thus the proof of Claim 3 is completed. $\square$

This concludes the proof of Lemma 18. $\square$

Lemma 19 $|\Pr[T_8] - \Pr[T_7]|$ *is negligible.*

Proof. Since we have $\Pr[C^* \in S : C^* \xleftarrow{\$} \mathbb{QR}_N] \leq 2^{-\lambda}$, in the following, we assume $C^* \notin S$. Then there exists $\bar{p}$ that divides $\mathrm{ord}(C^*)$ but does not divide $\mathrm{ord}(S)$. Let view be the view from $\mathcal{A}$ in Game 8 except K^*, and $\mathsf{view}' := \{\overline{PK}, C^*, \{x_{i,j}^{(k)} \mod \mathrm{ord}(S)\}_{i\in[\lambda],j\in[n],k\in\{0,1\}}\}$. By a similar argument as in the proof of Claim 3, we have $\Delta((K^*, \mathsf{view}), (U, \mathsf{view})) \leq \Delta((K^*, H, \mathsf{view}'), (U, H, \mathsf{view}'))$ and $\tilde{H}_\infty({C^*}^{\sum_{i=1}^\lambda x_{i,1}^{(t_i)}}, \ldots, {C^*}^{\sum_{i=1}^\lambda x_{i,n}^{(t_i)}} | \mathsf{view}') \geq n \log \bar{p} \geq n(\ell_B - 1) \geq (2\ell_N + 1)\lambda$. If we let $X := ({C^*}^{\sum_{i=1}^\lambda x_{i,1}^{(t_i)}}, \ldots, {C^*}^{\sum_{i=1}^\lambda x_{i,n}^{(t_i)}})$, $Y := \mathsf{view}$ and $\delta := 2^{-\ell_N\lambda}$ in Lemma 2, then we have $\Delta((K^*, H, \mathsf{view}'), (U, H, \mathsf{view}')) \leq 2^{-\ell_N\lambda}$ where $K^* = H({C^*}^{x_1 + t^* y_1}, \ldots, {C^*}^{x_n + t^* y_n})$ and $U \xleftarrow{\$} \{0,1\}^k$. Thus the lemma follows. $\square$

By the above lemmas, we have $\mathsf{Adv}^{\mathsf{CCCA}}_{\mathcal{A},\mathsf{KEM}_{\mathsf{CCCA}}}(\lambda) = |\Pr[T_1] - \Pr[T_8]| \leq \mathsf{negl}(\lambda) + 1/(2\mathsf{poly}(\lambda))$ for sufficiently large λ where negl is some negligible function. On the other hand, we assumed that there are infinitely many λ such that $\mathsf{Adv}^{\mathsf{CCCA}}_{\mathcal{A},\mathsf{KEM}_{\mathsf{CCCA}}}(\lambda) > 1/\mathsf{poly}(\lambda)$. Therefore for infinitely many λ, we have $1/(2\mathsf{poly}(\lambda)) < \mathsf{negl}(\lambda)$, which contradicts to that $\mathsf{negl}(\lambda)$ is negligible. Thus there does not exist a valid PPT adversary that breaks the CCCA security of the scheme. $\square$

Discussion. Here, we discuss the efficiency of the CCA secure PKE scheme that is obtained by combining the above KEM and an authenticated symmetric key encryption scheme. Table 1 shows the efficiency and hardness assumption of CCA secure PKE schemes based on the factoring in the standard model. Among existing schemes, the scheme proposed by Hofheinz and Kiltz [18] is one of the best in regard to the ciphertext overhead, which consists of 2 elements of $\mathbb{Z}_N^*$. In contrast, the ciphertext overhead of our scheme consists of only 1 element of $\mathbb{Z}_N^*$ plus a MAC. By giving a concrete parameter ($\ell'_p = \ell'_q = 160$, $\ell_B = 15$, $t_p = t_q = 32$ and $\ell_N = 1280$), the ciphertext overhead of our scheme is 1360-bit for 80-bit security whereas that of [18] is 2048-bit. On the other hand, the public key size of our scheme is much larger than that of [18], and an encryption and decryption are much less efficient than those in [18].

Acknowledgment

We would like to thank the anonymous reviewers and members of the study group "Shin-Akarui-Angou-Benkyou-Kai" for their helpful comments. Especially, we would like to thank the reviewer of EUROCRYPT 2016 who suggested to use the term "adversary-dependent" instead of "generalized", and Atsushi Takayasu for giving us useful comments on the Coppersmith theorem. We thank Haiyang Xue for pointing out a problem of the parameter setting in the previous version. This work was supported by CREST, JST and JSPS KAKENHI Grant Number 14J03467.

References

1. Mihir Bellare, Alexandra Boldyreva, and Adam O'Neill. Deterministic and efficiently searchable encryption. In *CRYPTO*, pages 535–552, 2007.
2. Mihir Bellare, Marc Fischlin, Adam O'Neill, and Thomas Ristenpart. Deterministic encryption: Definitional equivalences and constructions without random oracles. In *CRYPTO*, pages 360–378, 2008.
3. Mihir Bellare, Dennis Hofheinz, and Scott Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In *EUROCRYPT*, pages 1–35, 2009.
4. Lenore Blum, Manuel Blum, and Mike Shub. A simple unpredictable pseudo-random number generator. *SIAM J. Comput.*, 15(2):364–383, 1986.
5. Alexandra Boldyreva, Serge Fehr, and Adam O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In *CRYPTO*, pages 335–359, 2008.
6. Don Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. In *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 178–189, 1996.
7. Jean-Sébastien Coron, Antoine Joux, Avradip Mandal, David Naccache, and Mehdi Tibouchi. Cryptanalysis of the RSA subgroup assumption from TCC 2005. In *Public Key Cryptography*, pages 147–155, 2011.

8. Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *EUROCRYPT*, pages 45–64, 2002.
9. Richard Crandall and Carl Pomerance. *Prime Numbers: A Computational Perspective*. Springer, 2nd edition, 2005.
10. Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.
11. David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. In *Public Key Cryptography*, pages 279–295, 2010.
12. Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32, 1989.
13. Jens Groth. Cryptography in subgroups of $\mathbb{Z}_n^*$. In *TCC*, pages 50–65, 2005.
14. Goichiro Hanaoka, Takahiro Matsuda, and Jacob C. N. Schuldt. On the impossibility of constructing efficient key encapsulation and programmable hash functions in prime order groups. In *CRYPTO*, pages 812–831, 2012.
15. Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963.
16. Dennis Hofheinz and Eike Kiltz. Secure hybrid encryption from weakened key encapsulation. In *CRYPTO*, pages 553–571, 2007.
17. Dennis Hofheinz and Eike Kiltz. The group of signed quadratic residues and applications. In *CRYPTO*, pages 637–653, 2009.
18. Dennis Hofheinz and Eike Kiltz. Practical chosen ciphertext secure encryption from factoring. In *EUROCRYPT*, pages 313–332, 2009.
19. Eike Kiltz, Payman Mohassel, and Adam O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. In *EUROCRYPT*, pages 673–692, 2010.
20. Eike Kiltz, Adam O'Neill, and Adam Smith. Instantiability of rsa-oaep under chosen-plaintext attack. In *CRYPTO*, pages 295–313, 2010.
21. Eike Kiltz, Krzysztof Pietrzak, Martijn Stam, and Moti Yung. A new randomness extraction paradigm for hybrid encryption. In *Advances in Cryptology - EUROCRYPT 2009, 28th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Cologne, Germany, April 26-30, 2009. Proceedings*, pages 590–609, 2009.
22. Jr. Lenstra, H. W. Factoring integers with elliptic curves. *The Annals of Mathematics*, 126(3):pp. 649–673, 1987.
23. Xianhui Lu, Bao Li, and Yamin Liu. How to remove the exponent GCD in HK09. In *ProvSec*, pages 239–248, 2013.
24. Xianhui Lu, Bao Li, Qixiang Mei, and Yamin Liu. Improved efficiency of chosen ciphertext secure encryption from factoring. In *ISPEC*, pages 34–45, 2012.
25. Qixiang Mei, Bao Li, Xianhui Lu, and Dingding Jia. Chosen ciphertext secure encryption under factoring assumption revisited. In *Public Key Cryptography*, pages 210–227, 2011.
26. David Naccache and Jacques Stern. A new public key cryptosystem based on higher residues. In *ACM Conference on Computer and Communications Security*, pages 59–66, 1998.
27. Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *STOC*, pages 187–196, 2008.
28. Stephen C. Pohlig and Martin E. Hellman. An improved algorithm for computing logarithms over gf(p) and its cryptographic significance (corresp.). *IEEE Transactions on Information Theory*, 24(1):106–110, 1978.

29. John M. Pollard. Theorems of factorization and primality testing. In *Proceedings of the Cambridge Philosophical Society*, volume 76, pages 521–528, 1974.
30. John M. Pollard. A monte carlo method for factorization. In *BIT*, volume 15, pages 331–334, 1975.
31. John M. Pollard. Monte carlo methods for index computation (mod p). In *Math.Comp.*, volume 32, pages 918–924, 1978.
32. Ananth Raghunathan, Gil Segev, and Salil P. Vadhan. Deterministic public-key encryption for adaptively chosen plaintext distributions. In *EUROCRYPT*, pages 93–110, 2013.
33. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC 2005*, pages 84–93, 2005.
34. Daniel Shanks. Class number, a theory of factorization, and genera. In *1969 Number Theory Institute (Proc. Sympos. Pure Math., Vol. XX, State Univ. New York, Stony Brook, N.Y., 1969)*, pages 415–440. Providence, R.I., 1971.
35. Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *41st Annual Symposium on Foundations of Computer Science, FOCS 2000, 12-14 November 2000, Redondo Beach, California, USA*, pages 32–42, 2000.
36. Haiyang Xue, Bao Li, Xianhui Lu, Dingding Jia, and Yamin Liu. Efficient lossy trapdoor functions based on subgroup membership assumptions. In *CANS*, pages 235–250, 2013.
37. Takashi Yamakawa, Shota Yamada, Koji Nuida, Goichiro Hanaoka, and Noboru Kunihiro. Chosen ciphertext security on hard membership decision groups: The case of semi-smooth subgroups of quadratic residues. In *SCN*, pages 558–577, 2014.

A Omitted Proofs

A.1 Proof of Theorem 4

Proof. Assume that the scheme is not CPA secure. Then there exists a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ such that $\mathsf{Adv}^{\mathsf{CPA}}_{\mathcal{A},\mathsf{PKE}}(\lambda)$ is non-negligible. Then there exists a polynomial poly such that for infinitely many λ, $\mathsf{Adv}^{\mathsf{CPA}}_{\mathcal{A},\mathsf{PKE}}(\lambda) > 1/\mathsf{poly}(\lambda)$ holds. We consider the following sequence of games.

Game 1 This is the original CPA game between $\mathcal{A}$ and the challenger $\mathcal{C}$. That is, $\mathcal{C}$ generates $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleInj}(PP)$, and chooses $H \xleftarrow{\$} \mathcal{H}$. Then $\mathcal{C}$ gives $PK := (PP, \sigma, H)$, and $\mathcal{A}_1$ outputs (msg_0, msg_1, st). Then $\mathcal{C}$ chooses $b \xleftarrow{\$} \{0,1\}$ and $x \xleftarrow{\$} \{0,1\}^n$, computes $C_1 \leftarrow \mathsf{Evaluation}(PP, \sigma, x)$ and $C_2 := msg_b \oplus H(x)$ and gives $C := (C_1, C_2)$ to $\mathcal{A}_2$. Then $\mathcal{A}_2(C, st)$ outputs b'.

Game 2 This game is the same as the previous game except that σ is generated by $\mathsf{SampleLossy}(PP, I)$, where intuitively, I is an index such that "it is difficult to distinguish an injective function from a lossy function with index I for $\mathcal{A}$". To describe this precisely, we consider the following PPT algorithm $\mathcal{B}$.

$\mathcal{B}(PP, \sigma)$: Choose $H \xleftarrow{\$} \mathcal{H}$, run $(msg_0, msg_1, st) \leftarrow \mathcal{A}_1(PP, \sigma, H)$, choose $b \xleftarrow{\$} \{0,1\}$ and $x \xleftarrow{\$} \{0,1\}^n$, compute $C_1 \leftarrow \mathsf{Evaluation}(PP, \sigma, x)$ and $C_2 := msg_b \oplus H(x)$, run $b' \leftarrow \mathcal{A}_2((C_1, C_2), st)$. If $b = b'$, then output 1, and otherwise 0.

Let $\mathcal{S}_{\mathcal{B},1/(2\mathsf{poly})}$ be the algorithm that is assumed to exist in the definition of ad-LTDFs. (Note that $\mathcal{B}$ is a PPT algorithm and $1/(2\mathsf{poly})$ is noticeable.) In this game, we let $I \leftarrow \mathcal{S}_{\mathcal{B},1/(2\mathsf{poly})}(PP)$ and $\sigma \leftarrow \mathsf{SampleLossy}(PP, I)$.

Game 3 This game is the same as the previous game except that C_2 is set as $C_2 \leftarrow msg_b \oplus U$ where $U \xleftarrow{\$} \{0,1\}^\ell$.

Game 4 This game is the same as the previous game except that C_2 is set as $C_2 \xleftarrow{\$} \{0,1\}^\ell$.

Let T_i be the event that $b = b'$ holds in Game i. By the definition, we have $|\Pr[T_1] - 1/2| = \mathsf{Adv}^{\mathsf{CPA}}_{\mathcal{A},\mathsf{PKE}}(\lambda)$. Then we prove the following lemmas.

Lemma 20 *For sufficiently large any λ, we have* $|\Pr[T_2] - \Pr[T_1]| \leq 1/(2\mathsf{poly}(\lambda))$.

Proof. By the definition of ad-LTDFs, if we let

$$P_{\mathsf{inj}} := \Pr\left[1 \leftarrow \mathcal{B}(PP,\sigma) : \begin{array}{c}(PP,SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)\\ \sigma \leftarrow \mathsf{SampleInj}(PP)\end{array}\right]$$

$$P_{\mathsf{lossy}} := \Pr\left[1 \leftarrow \mathcal{B}(PP,\sigma) : \begin{array}{c}(PP,SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)\\ I \leftarrow \mathcal{S}_{\mathcal{B},1/(2\mathsf{poly})}(PP)\\ \sigma \leftarrow \mathsf{SampleLossy}(PP,I)\end{array}\right]$$

then we have $|P_{\mathsf{inj}} - P_{\mathsf{lossy}}| \leq 1/(2\mathsf{poly}(\lambda))$ for sufficiently large λ. It is clear that $P_{\mathsf{inj}} = \Pr[T_1]$ and $P_{\mathsf{lossy}} = \Pr[T_2]$ hold. Therefore the lemma follows.

Lemma 21 *We have* $|\Pr[T_3] - \Pr[T_2]| \leq \delta$.

In Lemma 2, we let $X := x$ and $Y := (PP, \sigma, f_\sigma(X))$. Then we have $\tilde{H}_\infty(X|Y) \geq \tilde{H}_\infty(X|PP,\sigma) - (n-k) = k \geq \ell + 2\log(1/\delta)$ since the size of the image of f_σ is at most 2^{n-k}. Then by Lemma 2, we have $\Delta((H(X),H,Y),(U,H,Y)) \leq \delta$ where $U \xleftarrow{\$} \{0,1\}^\ell$. Thus the lemma follows.

Lemma 22 *We have* $\Pr[T_4] = \Pr[T_3]$.

Proof. This is clear since U is independently random string.

Lemma 23 *We have* $\Pr[T_4] = 1/2$.

Proof. In Game 3, $\mathcal{A}$ is given any information about b. Therefore the probability that $\mathcal{A}$ can correctly guess b is $1/2$.

By combining these lemmas, for all sufficiently large λ, we have $|\Pr[T_1] - 1/2| \leq 1/(2\mathsf{poly}(\lambda)) + \delta$. That is, we have $\mathsf{Adv}^{\mathsf{CPA}}_{\mathcal{A},\mathsf{PKE}}(\lambda) \leq 1/(2\mathsf{poly}(\lambda)) + \delta$. Since we assumed for infinitely many λ, $\mathsf{Adv}^{\mathsf{CPA}}_{\mathcal{A},\mathsf{PKE}}(\lambda) > 1/\mathsf{poly}(\lambda)$, for infinitely many λ, we have $1/(2\mathsf{poly}(\lambda)) < \delta$. This contradicts to that δ is negligible. Therefore there does not exist a PPT adversary that breaks the scheme.

A.2 Proof of Lemma 12

Proof. We have

$$\begin{aligned}
&\Delta_{H \xleftarrow{\$} \mathcal{H}, X \leftarrow \mathcal{F}(H)}((H(X), H), (U, H)) \\
&= \mathbb{E}_{H \xleftarrow{\$} \mathcal{H}}[\Delta_{X \leftarrow \mathcal{F}(H)}(H(X), U)] \\
&\leq \mathbb{E}_{H \xleftarrow{\$} \mathcal{H}}[\sum_{X \in \mathcal{X}} \Delta(H(X), U)] \\
&= \sum_{X \in \mathcal{X}} \mathbb{E}_{H \xleftarrow{\$} \mathcal{H}}[\Delta(H(X), U)] \\
&= \sum_{X \in \mathcal{X}} \Delta_{H \xleftarrow{\$} \mathcal{H}}((H(X), H), (U, H)) \leq |\mathcal{X}|\delta
\end{aligned}$$

where the last inequality follows from Lemma 2. □

B Adversary-dependent All-but-one Lossy Trapdoor Function.

In this section, we define adversary-dependent all-but-one lossy trapdoor functions (ad-ABO) and construct it based on ad-LTDFs. Moreover we give more efficient construction of ad-ABO based on the ad-DRSA assumption.

B.1 Definition

For integers n and k such that $0 < k < n$, an (n, k)-adversary-dependent all-but-one lossy trapdoor function (ad-ABO) consists of 5 algorithms (ParamsGen, SampleInj, SampleABO, Evaluation, Inversion) and a family $\{\mathcal{I}(\lambda)\}_{\lambda \in \mathbb{N}}$ of lossy function index sets.

ParamsGen$(1^\lambda) \rightarrow (PP, SP)$ **:** It takes a security parameter 1^λ as input, and outputs a public parameter PP and a secret parameter SP.

SampleInj$(PP) \rightarrow \sigma$ **:** It takes a public parameter PP as input, and outputs a function description σ, which specifies an injective function f_σ over the domain $\{0,1\}^n \times \{0,1\}^{\ell_b}$.

SampleABO$(PP, b^*, I) \rightarrow \sigma$**:** It takes a public parameter PP, a lossy branch $b \in \{0,1\}^{\ell_b}$ and an all-but-one function index $I \in \mathcal{I}(\lambda)$ as input, and outputs a function index σ, which specifies a "all-but-one" function f_σ over the domain $\{0,1\}^n \times \{0,1\}^{\ell_b}$.

Evaluation$(PP, \sigma, b, x) \rightarrow f_\sigma(x, b)$**:** It takes a public parameter PP, function description σ, a branch b and $x \in \{0,1\}^n$ as input, and outputs $f_\sigma(x, b)$

Inversion$(SP, \sigma, b^*, b, y) \rightarrow x$**:** It takes a secret parameter SP, a function description σ,, $b^* \in \{0,1\}^{\ell_b} \cup \perp$, b and y and outputs the "inversion" x.

We require ad-LTDFs to satisfy the following three properties.
Correctness:

1. For all $x \in \{0,1\}^n$ and $b \in \{0,1\}^{\ell_b}$, we have $\mathsf{Inversion}(SP, \sigma, \perp, b, \mathsf{Evaluation}(PP, \sigma, b, x)) = x$ with overwhelming probability where $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleInj}(PP)$.

2. For all $x \in \{0,1\}^n$, $b^*, b \in \{0,1\}^{\ell_b}$ with $b \neq b^*$ and $I \in \mathcal{I}(\lambda)$, we have $\mathsf{Inversion}(SP, \sigma, b^*, b, \mathsf{Evaluation}(PP, \sigma, b, x)) = x$ with overwhelming probability where $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$ and $\sigma \leftarrow \mathsf{SampleABO}(PP, b^*, I)$.

All-but-one lossiness: For all $\lambda \in \mathbb{N}$, $(PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)$, $b^* \in B(\lambda)$, $I \in \mathcal{I}(\lambda)$ and $\sigma \leftarrow \mathsf{SampleABO}(PP, b^*, I)$, the image of $f_\sigma(\cdot, b^*)$ has size at most 2^{n-k}.

Indistinguishability between injective and ABO functions. Intuitively, we require that for any PPT adversary $\mathcal{A}$, there exists at least one lossy function index $I \in \mathcal{I}(\lambda)$ such that for all $b^* \in B(\lambda)$, $\mathcal{A}$ cannot distinguish an injective function from an ABO function with the lossy branch b^* and the lossy function index I.

The more precise definition is as follows. For any PPT adversary $\mathcal{A}$ and noticeable function $\epsilon(\lambda)$, there exists a PPT algorithm $\mathcal{S}^{\mathsf{abo}}_{\mathcal{A},\epsilon}$ that takes a public parameter PP and a lossy branch b^* as input and outputs $I \in \mathcal{I}(\lambda)$ such that the following is satisfied. For all $b^* \in B(\lambda)$, if we let

$$P_{\mathsf{inj}} := \Pr\left[1 \leftarrow \mathcal{B}(PP, \sigma) : \begin{array}{c} (PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda) \\ \sigma \leftarrow \mathsf{SampleInj}(PP) \end{array}\right]$$

$$P_{\mathsf{lossy}} := \Pr\left[1 \leftarrow \mathcal{B}(PP, \sigma) : \begin{array}{c} (PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda) \\ I \leftarrow \mathcal{S}_{\mathcal{B},1/(2\mathsf{poly})}(PP) \\ \sigma \leftarrow \mathsf{SampleLossy}(PP, I) \end{array}\right]$$

$$P_{\mathsf{inj}} := \Pr\left[1 \leftarrow \mathcal{A}(PP, \sigma) : \begin{array}{l} (PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda) \\ \sigma \leftarrow \mathsf{SampleInj}(PP) \end{array}\right]$$

$$P_{\mathsf{lossy},b^*} := \Pr\left[1 \leftarrow \mathcal{A}(PP, \sigma) : \begin{array}{l} (PP, SP) \leftarrow \mathsf{ParamsGen}(1^\lambda) \\ I \leftarrow \mathcal{S}^{\mathsf{abo}}_{\mathcal{A},\epsilon}(PP, b^*) \\ \sigma \leftarrow \mathsf{SampleABO}(PP, b^*, I) \end{array}\right]$$

then we have $|P_{\mathsf{inj}} - P_{\mathsf{lossy},b^*}| \leq \epsilon(\lambda)$ for sufficiently large λ.

B.2 Generic Construction

Here, We construct an ad-ABO based on an ad-LTDF. Let $(\mathsf{ParamsGen}_{\mathsf{gltdf}}, \mathsf{SampleInj}_{\mathsf{gltdf}}, \mathsf{SampleLossy}_{\mathsf{gltdf}}, \mathsf{Evaluation}_{\mathsf{gltdf}}, \mathsf{Inversion}_{\mathsf{gltdf}})$ be an (n,k)- ad-LTDF

Definition of $\mathcal{I}(\lambda)$: $\mathcal{I}(\lambda)$ is the same as that of the underlying ad-LTDF.
$\mathsf{ParamsGen}_{\mathsf{gabo}}(1^\lambda) \rightarrow (PP, SP)$**:** Run $(PP, SP) \leftarrow \mathsf{ParamsGen}_{\mathsf{gltdf}}(1^\lambda)$ and output (PP, SP).

$\mathsf{SampleInj_{gabo}}(PP) \to \sigma$**:** For all $i \in [\ell_b]$, generate $\sigma_{i,0}, \sigma_{i,1} \leftarrow \mathsf{SampleInj_{gltdf}}(PP)$, and output $\sigma := \{\sigma_{i,j}\}_{i\in[\ell_b], j\in\{0,1\}}$.

$\mathsf{SampleABO_{gabo}}(PP, b^*, I) \to \sigma$**:** For all $i \in [\ell_b]$, generate $\sigma_{i,b_i^*} \leftarrow \mathsf{SampleLossy_{gltdf}}(PP, I)$ and $\sigma_{i,1-b_i^*} \leftarrow \mathsf{SampleInj_{gltdf}}(PP)$, and output $\sigma := \{\sigma_{i,j}\}_{i\in[\ell_b], j\in\{0,1\}}$.

$\mathsf{Evaluation_{gabo}}(PP, \sigma, b, x) \to f_\sigma(x, b)$**:** For all $i \in [\ell_b]$, compute $y_i := \mathsf{Evaluation_{gltdf}}(PP, \sigma_{i,b_i}, x)$, and output $y := \{y_i\}_{i\in[\ell_b]}$.

$\mathsf{Inversion_{gabo}}(SP, \sigma, b^*, b, y) \to f_\sigma^{-1}(y)$**:** Find $i \in [\ell_b]$ such that $b_i^* \neq b_i$, compute $x := \mathsf{Inversion_{ltdf}}(SP, \sigma_{i,b_i}, y_i)$ and output x.

Theorem 7 *If the underlying scheme is an $(n, n-r)$-ad-LTDF, then the above scheme is an $(n, n - r\ell_B)$-ad-ABO.*

The proof is almost the same as the proof of generic construction of all-but-one lossy trapdoor function from a lossy trapdoor function in [27]. Therefore we omit it.

B.3 Direct Construction

Here, we construct an ad-ABO based on the m-ad-DRSA assumption directly. Let IGen be an algorithm that generates an ℓ_N-bit (ℓ_B, t_p, t_q)-SS RSA modulus with the parameter given in Sec. 2.4 and $n := \frac{(t-d)(\ell_B-1)-\ell_b}{2}$.

Definition of $\mathcal{I}(\lambda)$: $\mathcal{I}(\lambda)$ is defined as the set of all m-tuple of distinct primes of length ℓ_B. That is, we define $\mathcal{I}(\lambda) := \{\{p_1, \dots, p_m\} : p_1, \dots, p_m$ are distinct ℓ_B bit primes$\}$.

$\mathsf{ParamsGen}(1^\lambda) \to (PP, SP)$**:** Generate $(N, P, Q) \leftarrow \mathsf{IGen}(1^\lambda)$, set $PP := N$ and $SP := (P, Q)$, and output (PP, SP).

$\mathsf{SampleInj}(PP = N) \to \sigma$**:** Choose $g, h \xleftarrow{\$} \mathbb{QR}_N$ and output $\sigma := (g, h)$.

$\mathsf{SampleABO}(PP = N, b^*, I = \{p_1, \dots, p_m\}) \to \sigma$**:** Choose $g, g' \xleftarrow{\$} \mathbb{QR}_N$, set $h := g^{-b^*} g'^{p_1 \cdots p_m}$ and output $\sigma := (g, h)$.

$\mathsf{Evaluation}(PP = N, \sigma = (g, h), b, x \in \{0,1\}^n) \to f_\sigma(x)$**:** Interpret x as an element of $[2^n]$ and output $(g^b h)^x$.

$\mathsf{Inversion}(SP = (P, Q), \sigma = (g, h), b^*, by) \to f_\sigma^{-1}(y)$**:** Compute $x = \mathsf{PLog}(P, Q, g^b h, y)$ and output x where PLog is the algorithm given in Lemma 7.

Theorem 8 *If the m-ad-DRSA assumption holds with respect to* IGen*, then the above scheme is an $(n, n - (\ell_{p'} + \ell_{q'} + (M_{\ell_B} - m)\ell_B))$-ad-ABO.*

Proof.
Correctness.

1. If $\sigma = (g, h)$ is generated by $\mathsf{SampleInj}$, then they are independently uniform elements of $\mathbb{QR}_N$. In particular, for any $b \in \{0,1\}^{\ell_b}$, $g^b h$ is a uniform element of $\mathbb{QR}_N$. Thus $\mathsf{Inversion}((P, Q), (g, h), \perp, b, \mathsf{Evaluation}(N, \sigma, b, x)) = \mathsf{Inversion}((P, Q), g^b h, (g^b h)^x) = x$ holds by the correctness of PLog given in Lemma 7.

2. Let $(N,(P,Q)) \leftarrow \mathsf{ParamsGen}$ and $\sigma = (g,h) \leftarrow \mathsf{SampleABO}(N, b^*, I = \{p_1,\ldots,p_m\})$ for some $b^* \in \{0,1\}^{\ell_b}$. Then g is a uniform element of $\mathbb{QR}_N$ and $h = g^{-b^*} g'^{p_1\cdots p_m}$ for $g' \xleftarrow{\$} \mathbb{QR}_N$. Then for any $b \neq b^*$, we have $g^b h = g^{b-b^*} g'^{p_1\cdots p_m}$. By Lemma 5, $\mathrm{ord}(g)$ has at least $t-d$ distinct ℓ_b-bit prime divisors. The number of these prime divisors that are not coprime to $b-b^*$ is at most $\log_{2^{\ell_B-1}}(b-b^*) \leq \ell_b/(\ell_B-1)$. We write $p'_1,\ldots,p'_s$ to denote the all ℓ_b-bit divisors of $\mathrm{ord}(g)$ that are coprime to $b-b^*$. Then we have $s \geq t-d-\ell_b/(\ell_B-1)$. For each p'_i, the probability that $\mathrm{ord}(g^b h)$ is comprime to p'_i is $1/p'_i \leq 2^{-\ell_B+1}$ and they are all independent. Then one can see that there exists more than $s/2$ p'_i such that $p'_i|\mathrm{ord}(g^b h)$ by similar analysis as in [13, Lemma3]. Therefore one can recover x from $(g^b h)^x$ as long as $x \leq 2^{\frac{(t-d)(\ell_B-1)-\ell_b}{2}} \leq (2^{\ell_B-1})^{s/2}$.

All-but-one lossiness. Next, we prove that the above construction satisfies the all-but-one lossiness property. Let $\sigma = (g,h)$ be a function description generated by $\mathsf{SampleLossy}(N, I = \{p_1,\ldots,p_m\})$. Then there exists $g' \in \mathbb{QR}_N$ such that $h = g^{-b^*} g'^{p_1\cdots p_m}$. In particular, we have $g^{b^*}h = g'^{p_1\cdots p_m}$. Then any output of $f_\sigma(\cdot, b^*)$ is an element of the group $S := \{h'^{p_1\cdots p_m} : h' \in \mathbb{QR}_N\}$. We consider the order of S. S is a subgroup of $\mathbb{QR}_N = G \times G^\perp$ and $p_1\ldots p_m$ is coprime to $\mathrm{ord}(G) = p'q'$. Therefore there exists a subgroup $S^\perp$ of $G^\perp$ such that $S = G \times S^\perp$. We can see that $\mathrm{ord}(S^\perp)$ is the product of some distinct ℓ_B-bit primes and coprime to $p_1\ldots p_m$ by the definition. Therefore that is the product of at most $M_{\ell_B} - m$ such primes, and can be bounded by $2^{(M_{\ell_B}-m)\ell_B}$. Therefore the order of S is at most $2^{(\ell_{p'}+\ell_{q'})(M_{\ell_B}-m)\ell_B}$.

Indistinguishability between injective and all-but-one lossy functions. This follows from the m-ad-DRSA assumption. Actually, for any PPT adversary $\mathcal{A}'$ that tries to distinguish these functions and any $b^* \in \{0,1\}^{\ell_b}$, we consider the following adversary $\mathcal{A}'_{b^*}$ against the ad-DRSA assumption.

$\mathcal{A}'_{b^*}(N,\bar{g})$**:** Generate $g \xleftarrow{\$} \mathbb{QR}_N$, set $h := g^{b^*}\bar{g}$, run $\mathcal{A}(N,(g,h))$ and output as $\mathcal{A}$ outputs.

For any noticeable ϵ, let $\mathcal{S}^{\mathsf{gdrsa}}_{\mathcal{A}_{b^*},\epsilon}$ be the algorithm that is assumed to exist in Definition 3. If we let

$$P_0 := \Pr\left[1 \leftarrow \mathcal{A}(N,g) : \begin{array}{l}(N,P,Q) \leftarrow \mathsf{IGen}(1^\lambda)\\ g \xleftarrow{\$} \mathbb{QR}_N\end{array}\right]$$

$$P_1 := \Pr\left[1 \leftarrow \mathcal{A}(N,g^{p_1\cdots p_m}) : \begin{array}{l}(N,P,Q) \leftarrow \mathsf{IGen}(1^\lambda)\\ g \xleftarrow{\$} \mathbb{QR}_N\\ \{p_1,\ldots,p_m\} \leftarrow \mathcal{S}^{\mathsf{gdrsa}}_{\mathcal{A}_{b^*},\epsilon}(N)\end{array}\right]$$

then we have $|P_0 - P_1| \leq \epsilon(\lambda)$ for sufficiently large λ by the definition of the ad-DRSA assumption.

We let $\mathcal{S}^{\mathsf{abo}}_{\mathcal{A},\epsilon,b^*} := \mathcal{S}^{\mathsf{gdrsa}}_{\mathcal{A}^*_b,\epsilon}$. In the following, we show that it works well. If $\bar{g}$ is a uniform element of $\mathbb{QR}_N$, then h is a uniform on $\mathbb{QR}_N$ and independent of g. Thus in this case, $\mathcal{A}'_{b^*}$ simulates the environment for $\mathcal{A}$ where it is given an injective function. On the other hand, if $\bar{g}$ is generated as $\bar{g} := \bar{g}'^{p_1 \cdots p_m}$ where $\{p_1, \ldots, p_m\} \leftarrow \mathcal{S}^{\mathsf{gdrsa}}_{\mathcal{A}^*_b,\epsilon}$, then $\mathcal{A}'_{b^*}$ simulates the environment for $\mathcal{A}$ where it is given an all-but-one function with lossy branch b^* and index $I := \{p_1, \ldots, p_m\}$.

Therefore if we let

$$P_{\mathsf{inj}} := \Pr\left[1 \leftarrow \mathcal{A}(PP,\sigma) : \begin{array}{l}(PP,SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)\\ \sigma \leftarrow \mathsf{SampleInj}(PP)\end{array}\right]$$

$$P_{\mathsf{abo},b^*} := \Pr\left[1 \leftarrow \mathcal{A}(PP,\sigma) : \begin{array}{l}(PP,SP) \leftarrow \mathsf{ParamsGen}(1^\lambda)\\ I \leftarrow \mathcal{S}_{\mathcal{A},\epsilon}(PP)\\ \sigma \leftarrow \mathsf{SampleABO}(PP,b^*,I)\end{array}\right]$$

then we have $P_{\mathsf{inj}} = P_0$ and $P_{\mathsf{abo},b^*} = P_1$. Therefore we have $|P_{\mathsf{inj}} - P_{\mathsf{lossy},b^*}| \leq \epsilon(\lambda)$ for sufficiently large λ as required.