On the Security of Cryptosystems with All-or-Nothing Transform

Rui Zhang, Goichiro Hanaoka, and Hideki Imai

Institute of Industrial Science, University of Tokyo
{zhang,hanaoka}@imailab.iis.u-tokyo.ac.jp, imai@iis.u-tokyo.ac.jp

Abstract. An AONT is an efficiently computable transform with two properties. Given all the bits of its output, it is easy to retrieve the message. On the other hand, if sufficiently many bits of the output are missing, it is computationally infeasible for an polynomial-time adversary to learn any information about the message. The natural intuition then may be deduced that if an secure AONT is used in a cryptosystem, the whole system will be secure as long as sufficiently many bits are "protected". However, we show this is not enough. Our results are three-fold: First we answer an open problem raised in [6], showing that previous definitions are not sufficient to guarantee a provably secure cryptosystem with strong data privacy, namely, indistinguishability against chosen ciphertext attack (IND-CCA). Second, we give a new definition to AONT, showing this definition suffices to guarantee an AONT integrated with any encryption functions to acquire IND-CCA secure cryptosystems. Third, we give concrete constructions that satisfy the new definition.

1 Introduction

The Concept. All-or-Nothing transform (AONT) was introduced by Rivest in [17] to increase the cost of brute force attacks on block ciphers without changing the key length. As originally defined in [17], an AONT is a randomized transform T that can be computed efficiently mapping sequences of blocks $(x_1, ..., x_n)$ to sequences of blocks $(y_1, ..y_{n'})$, with the following properties:

- If all the $T(x_1, ..., x_n) = (y_1, ..., y_{n'})$ blocks are given, it is easy to compute $(x_1, ..., x_n)$.
- Even if one of the blocks of output $(y_1, ..., y_{n'})$ is missing, it is infeasible to find out any information of any of the original blocks $(x_1, ..., x_n)$.

If such a transform is applied to a message producing a sequence of output blocks, and each of these blocks is encrypted by a block cipher, interestingly, an adversary will have no information unless it can decrypt all the cipher blocks. Thus the attack will be slowed down by a factor of n' without even changing the length of the secret key. However, since the security of AONT and the data privacy of a cryptosystem were independently developed in literature, one may naturally ask the following questions: Is a cryptosystem secure if it is composed by a "secure" AONT with an encryption component? In other words, how can we safely utilize an AONT in a cryptosystem? In this paper, we try to give an answer to such questions.

M. Jakobsson, M. Yung, J. Zhou (Eds.): ACNS 2004, LNCS 3089, pp. 76–90, 2004.

Applications of AONT. First possible category of applications, as has been addressed already, can be used as a mode of operation for block cipher to enhance the security against exhaustive search attack security without increasing key length, as proposed in [17,8]. AONT can also be combined with cryptosystems to reduce the computation cost of a bandwidth limited device. This was also known as remotely keyed encryption. If an AONT is performed on a long message to be sent, because of the nice property of AONT, only a small proportion, say a few blocks of output of AONT needs to be encrypted, as shown by Jakobsson, Stern and Yung [13]. In [14] for inclusion in the IEEE P1363a standard, an ANOT was proposed to make fixed-block size encryption schemes more efficient. The authors further claim that this method is encryption algorithm independent, that is, any asymmetric or symmetric key encryption. However, this needs *more careful discussions*, as we shall show later.

With AONT, one can design a cryptosystem with separate component, say, a smart card, which holds the secret key independent from the main system. By updating the secret keys from time to time, one can acquire strong key-insulated cryptosystem [9]. It was further generalized in [21] in constructing a parallel construction of multiple encryption to enhance the security of a single component cipher. Besides, as pointed out in [6], one might use AONT for gradual exchange of information. Suppose two users Alice and Bob want to exchange the secrets they hold. One possible problem is that the secret might be of different lengths. Then we can apply AONT to "pad" both secrets to equal length. Additional zero-knowledge proof should be attached to prevent cheating.

AONT Enhances Data Privacy? From above discussion, one may naturally think that if a secure AONT is used in the system, the data privacy can be protected, as long as the underlying AONT is secure and efficiently many bits of the transformed message are protected by the encryption component. However, we argue that this intuition may be *not* true. At least, it may be fallacious according to chosen ciphertext security (CCA), which is considered as a standard security notion for practical cryptosystems. For why chosen ciphertext security is important, one may refer to [18]. We have noticed that in the context of authenticated encryption, it has been pointed out in [3] that for several construction methods by combining a secure message authentication code (MAC) with a secure encryption scheme, the resulting authenticated encryption may be insecure at all.

Previous Definitional Efforts and Related Primitives. The first definition was given in Rivest's original work [17], however, the definition simply mentions the case where the adversary "loses" a particular message block. It did not, however, mention the exact information that an adversary learn about the input with several bits invisible and how the adversary learns the information of the input related to bits that the adversary holds regarding the output was not addressed yet.

Desai studied AONT in the context of the security of symmetric key encryption against key search attack [8], and gave a definition of AONT. Again, in his model the security is defined in a block-wise manner: if there are some missing blocks cannot be learned by the adversary, it is considered secure. He claims

that this suffices in building an operation mode of block ciphers secure in the terms of non-separability of keys. Stinson has considered AONT from the point of view of unconditional security [20]. However, his treatment is also considered the amount of information leaked by a particular block and the definition is just straightforward formalization of Rivest's definition in the information-theoretic security.

Aware of this shortage, Boyko [6] gave a new definition, namely, indistinguishability [12] against adaptive attack in the random oracle model [11,4]. In this model, an adversary can adaptively choose the positions of bits of the output of AONT to learn, however, below a certain threshold. It is also proved in [6] that OAEP, which was proposed by Bellare and Rogaway [5] with a different goal to obtain IND-CCA secure encryption schemes [12,16,15,10,2], is a secure implementation satisfying this definition, moreover, no AONT can do significantly better that OAEP. Later, Canetti et al. [7] gave a similar definition in the standard model (cf. random oracle model), furthermore, they constructed secure AONT under their definition based on exposure-resilient functions (ERF). They also proved the existence of ERF is equivalent to that of oneway functions. Though the existence of special class of exposure-resilient functions that are used in their OAEP-like construction is still left open.

A similar notion, *concealment*, was proposed in the context of remotely keyed authenticated encryption by An and Dodis [1]. Both of these two notions provide secrecy of the message, when even most of the blocks are given to the adversary. The difference is that concealment also provides authentication (knowledge of the plaintext), while an AONT does not necessarily need.

1.1 Our Contribution

Adjusted Security Notion on AONT. We show that previous definitions of AONT are insufficient to guarantee cryptosystems with strong security, e.g., IND-CCA. We demonstrate that there exist cryptosystems, with an AONT secure in the sense of above definitions, however, are not secure against CCA attack. This also answers an open problem raised in [6] negatively, where Boyko wondered if OAEP can be replaced by an arbitrary AONT in the construction of a CCA secure encryption scheme. We pointed out previous definitions of AONT were either defined in a scenario where only chosen plaintext attack (CPA) is considered, or operates with some "ideal" encryption component, e.g., block cipher (often modeled as random permutation), or IND-CCA secure encryption component (strongest security for public key encryption). The security of AONT joined with arbitrary encryption component against adaptive attacks has not been thoroughly considered yet.

New Definition regarding AONT. Actually, since AONT is only a randomized transform, which contains no secret key information, it may lead to fallacious conclusion if the security of the whole system is considered merely based on the security of AONT. In the real world an active attacker, who may be a legal user of this system, is capable of launching adaptive attacks. Thus

we suggest that the security of the system should be considered as a joint contribution of AONT and the encryption component. We give a new definition of AONT based on indistinguishability, called *extended-indistinguishability*, which is defined together with encryption component. A straightforward consequence turns out that if an AONT with extended-indistinguishability is used in a cryptosystem together with arbitrary encryption scheme, the resulting cryptosystem is IND-CCA secure.

Construction of Extended-Indistinguishable AONTs. We also give two constructions of AONT satisfying the new definition. The first one, provably secure in the random permutation model, is capable for deterministic encryption primitives. The second one, provably secure in the random oracle model, is capable for probabilistic encryption primitives.

2 Preliminary

2.1 Notations and Model

Throughout this paper, we limit our scope within "efficiently computable" algorithms, which means that algorithms have expected polynomial execution time. A function $f : \mathsf{D} \to \mathbf{R}$ is called *negligible* if for every constant $l \geq 0$ there exists an integer k such that $f(k) \leq k_c^{-l}$ for all $k \geq k_c$, denoted by $\mathsf{neg}(k)$.

$X \approx Y$ denotes that probability distribution X are computationally indistinguishable from Y. We shall use $x \overset{R}{\leftarrow} X$ to denote x is uniformly selected from distribution X. Suppose X is an algorithm, $x \leftarrow X$ denotes x is set to the output of X. We also use $x \oplus y$ to denote bit-wise XOR of two binary strings x and y. Let Ω be all the mappings from set of infinite strings $\{0,1\}^\infty$ to set of finite strings $\{0,1\}^*$, then $G, H \leftarrow \Omega$ denotes two random function G and H are selected uniformly from Ω, whose input and output sizes should be restricted accordingly in proper context. For an integer n and $L \in [1, n]$, we define $h_{n,L} : \{0,1\}^n \to \{0,1\}^{n-|L|}$ as for an input binary string of length n, $h_{n,L}$ returns a punctured string with the bit positions that is indicated by label L.

2.2 Public Key Encryption

A public key encryption scheme $\mathcal{E}$ is a 3-tuple algorithm: $\mathcal{E} = (\mathsf{Enc\text{-}Gen}, \mathsf{Enc}, \mathsf{Dec})$. $\mathsf{Enc\text{-}Gen}(1^k)$ is a probabilistic algorithm, where k is the security parameter, with internal random coin flipping outputs a pair of keys (pk, sk). pk is the encryption key which is made public, and sk is the decryption which is kept secret. Enc may be a probabilistic algorithm that takes as input a key pk and a message m from associated message space $\mathcal{M}$, and internally flips some coins and outputs a ciphertext c, denoted by $c \leftarrow \mathsf{Enc}_{pk}(m)$, in short $c \leftarrow \mathsf{Enc}(m)$. Dec is a deterministic algorithm takes as input the ciphertext c and the secret key sk, and outputs some message $m \in \mathcal{M}$, or "$\perp$" in case c is "invalid". We denote it by $m \leftarrow \mathsf{Dec}_{sk}(c)$, in short $m \leftarrow \mathsf{Dec}(c)$.

Indistinguishability under chosen-ciphertext attack (IND-CCA), is defined as: if no PPT adversary $\mathcal{A}$ can distinguish encryptions of any two messages (M_0, M_1)

of equal length chosen by it with negligible advantage than random guess in the following game. We require that $\mathcal{A}$ runs in two stages $\mathcal{A}_{\mathsf{find}}$ and $\mathcal{A}_{\mathsf{guess}}$, in which $\mathcal{A}_{\mathsf{find}}$ gets side information α from the queries and output a pair of challenge messages, and $\mathcal{A}_{\mathsf{guess}}$ outputs a guess $\tilde{b}$ on b according to the ciphertext C_b encrypted by the Encryption Oracle with randomly chosen $b \in \{0,1\}$. According to the ability of the adversary, $\mathcal{A}_{\mathsf{find}}$ and $\mathcal{A}_{\mathsf{guess}}$ can be assisted by an Decryption Oracle $\mathcal{DO}$ that returns the plaintext for a decryption query other than the target ciphertext. Note that according to the adversary's ability, sometimes $\mathcal{DO}$ is unavailable, (this can be equivalently denoted by $\mathcal{DO}$ outputting an empty string ϵ). In our analysis, it is sufficient to consider the case where $\mathcal{DO}$ is available. We denote this as:

$$\Pr\left[b = \tilde{b} \middle| \begin{array}{c}(pk, sk) \leftarrow \mathsf{Enc\text{-}Gen}(1^k), (M_0, M_1, \alpha) \leftarrow \mathcal{A}_{\mathsf{find}}^{\mathcal{DO}}(pk), \\ b \overset{R}{\leftarrow} \{0,1\}, C_b \leftarrow \mathsf{Enc}(M_b), \tilde{b} \leftarrow \mathcal{A}_{\mathsf{guess}}^{\mathcal{KE},\mathcal{DO}}(C_b, \alpha)\end{array}\right] \leq \frac{1}{2} + \mathsf{neg}(k)$$

If no such PPT adversary exists against $\mathcal{E}$, then we call $\mathcal{E}$ IND-CCA secure.

2.3 Previous Definitions on AONT

Definition from [6] In fact, in [6], several definitions are presented based on semantic security and indistinguishability [12], against adaptive and non-adaptive attacks. From the quantitive results given in [6], we notice that the upper bounds of sematic security and indistinguishability against adaptive attacks are essentially the same. It is sufficient to only consider the indistinguishability-based security definition.

Definition 1. *AONT is a randomized transform $T(x) : \{0,1\}^n \rightarrow \{0,1\}^{n'}$, which is efficiently computable. with all bits of the output, there is an inverse function I, which can uniquely recover x: $I(T(x)) = x$. Suppose an adversary runs the experiment in the following stages:*

1. Select*: The adversary is given l and access to Γ. It selects l bit positions and outputs labels of positions $L \in \{^{n'}_{l}\}$ and side-information $c_s \in \{0,1\}^*$.*
2. Find*: The adversary is given c_s and access to Γ. It outputs $x_0 \in \{0,1\}^n$, $x_1 \in \{0,1\}^n$ and side-information $c_f \in \{0,1\}^*$.*
3. Guess*: The adversary is given c_f and for random bit b, $\mathrm{AONT}^{\Gamma}(x_b)$ with bit positions L missing. The adversary has access to Γ and tries to guess b.*

Let AONT be a randomized transform mapping n-bit messages to n'-bit outputs and using random oracle Γ. Let l between 1 and n'. An adversary $\mathcal{A}$ is said to succeed in (T, q_Γ, ϵ)-adaptively-distinguishing AONT with l missing bits if

$$\Pr\left[\tilde{b} = b \middle| \begin{array}{c}\Gamma \leftarrow \Omega, (L, c_s) \leftarrow \mathcal{A}^{\Gamma}_{\mathsf{select}}(l), (x_0, x_1, c_f) \leftarrow \mathcal{A}^{\Gamma}_{\mathsf{find}}(c_s), \\ b \overset{R}{\leftarrow} \{0,1\}, y \leftarrow \mathrm{AONT}^{\Gamma}(x_b), \tilde{b} \leftarrow \mathcal{A}^{\Gamma}_{\mathsf{guess}}(h_{n',L}(y), c_f)\end{array}\right] \geq \tfrac{1}{2} + \varepsilon$$

and moreover, in the experiment above, A runs at most T steps and makes at most q_Γ queries to Γ. Then the AONT is secure if no probabilistic polynomial time adversary exists.

Furthermore, it is proved that OAEP [5] is a secure implementation of AONT in the above sense in the random oracle model.

Definition from [7] We can see this definition is significantly the same as the Definition 1, except that the latter can only be defined in the random oracle model. Definition 2 indicates more general case. In addition, [7] also divides the output y of an AONT into two sections: one is called the *public part* y_2, which does not need protection, that is, it can be revealed to the adversary. The other section is called *secret part* y_1, which needs some protection. The security guarantee is: as long as l bits of the secret output y_1 remain hidden, while all the bit of y_2 can be revealed, the adversary should have no information about the message.

Definition 2. *A randomized polynomial time computable function* $T(x) : \{0,1\}^k \rightarrow \{0,1\}^{n'}$ *is* l*-AONT if*

1. T *is efficiently invertible, i.e., there is a polynomial time machine* I *such that for any* $x \in \{0,1\}^k$ *and any* $y \in T(x)$*, we have* $I(y) = x$*.*
2. *For any label* $L \in \{^{n'}_{l}\}$ *and any* $x_0, x_1 \in \{0,1\}^k$ *chosen by the adversary* adaptively*, we have*

$$\langle x_0, x_1, [T(x_0)]_L \rangle \approx \langle x_0, x_1, [T(x_1)]_L \rangle$$

The construction of [7] makes use of *exposure-resilient functions* (ERF). Informally, an l-ERF is a special type of pseudorandom generator whose output remains computationally indistinguishable from a random sequence as long as l bits of its seed remain hidden. Refer [7] for formal definition and construction of ERF. A construction satisfying above Definition 2 was proposed and has been proved secure (theorem 5.1 of [7]).

3 AONT Enhances Data Privacy?

As we know, since an AONT contains no secret information itself, and it does no encryption, when integrated in a cryptosystem, the security of the whole system rather than AONT itself should be considered. Above two definitions [6,7] have considered AONT against adaptive attacks, however, the security of other component of the cryptosystem, especially the security of the encryption component is never confronted with.

Actually, definitions given in [6,7] are sufficient for a chosen plaintext attack (CPA). A simple reasoning is listed here: if the attacker can break the security of the cryptosystem then it can be used as a subroutine, to break either the indistinguishability of the AONT or the encryption component. A similar argument in proving the CPA security of a generic construction for key-insulated cryptosystem can be found in [9], yet in a different context.

However, the same argument is not applicable in discussing the CCA security of the cryptosystem. Problems may occur when an AONT meeting security definitions of [6,7] works with a malleable encryption scheme. Here we demonstrate two examples.

3.1 Example 1

The first example is an attack on OAEP, which was first exhibited by Shoup [19] in disproving the original secure result of OAEP. OAEP can be described as follows: two hash functions G, H are considered as random functions, a message m is masked by: $s = m \oplus G(r)$, $t = r \oplus H(s)$. Then the ciphertext is $c = \varphi_{pk}(s, t)$, where φ is oneway trapdoor function defined by pk and sk. For decryption, one computes

$$(s, t) = \varphi_{sk}(s, t), \quad r = t \oplus H(s), \quad m = s \oplus G(r).$$

Suppose there exists XOR-malleable f (refer [19] for precise definition), which is oneway trapdoor function with following properties: Given $\varphi_{pk}(x) = (s, f(t))$, one can efficiently compute $f(x \oplus \triangle x)$, where $\triangle x$ is any binary string with the same length as x.

For any challenge ciphertext $c_b = (s, f(t))$ given by the encryption oracle in the IND-CCA game, the adversary can choose any random string $\triangle x$ and compute $s' = s \oplus \triangle x$ and $f(t') = f(t \oplus H(s) \oplus H(s'))$, which yields a new ciphertext c'. If the adversary queries c' at the Decryption Oracle, which will returns $m_b \oplus \triangle x$, and the adversary can easily recover m_b and guess b correctly. The reason why this attack works is that the adversary can make the ciphertext malleable.

From above description, one can see that if OAEP is used as AONT in a cryptosystem, and the encryption component (encrypting t or part of t) happens to be XOR-malleable, the whole system is not IND-CCA secure.

3.2 Example 2

The second example is more straightforward. The following construction is given in [7]: Let $f : \{0,1\}^n \to \{0,1\}^{n'}$ be computational l-ERF. Define $T : \{0,1\}^k \to \{0,1\}^n \times \{0,1\}^k$ (with n random bits r) as follows: $T(x; r) = \langle r, f(r) \oplus x \rangle$. Then T is l-AONT with secret part r and public part $f(r) \oplus x$.

For this one-time pad like construction, with the same seed r as the secret part and $f(r) \oplus x$ as the public part (without any encryption), one can compute $f(r) \oplus x \oplus \triangle x$, where $\triangle x$ is any binary string. The resulting ciphertext becomes $E(r)$ (secret part under encryption) and $f(r) \oplus x \oplus \triangle x$ (public part to transmit in plaintext). Again, it is easily seen that the whole cryptosystem is not IND-CCA secure, either.

4 New Definition Regarding AONT

We have manifested that under present definition of AONT, it is not sufficient to guarantee CCA security of the whole cryptosystem. However, the fact that there are no obvious attacks to the security of previous constructions of AONT seems to contradict above counterexamples. We figure that AONT was originally proposed for block cipher, and in the theoretical analysis block cipher is usually modeled as random permutation. One may think that a random permutation is somehow a transform with authentication, at least in a weak sense. For block cipher, informally speaking chosen ciphertext attack is almost the same

as chosen plaintext attack, for the random permutation will leave the ciphertext non-malleable. On the other hand, for a public key cryptosystem, chosen ciphertext attack is more powerful attack. In a cryptosystem with AONT, whenever the encryption component is malleable, the security of the whole system, regardless of that of AONT, may be insecure.

Then two natural questions arise that how should one consider the security of AONT and how should it be implemented in designing a secure cryptosystem? We proceed to solve these problems.

4.1 Public Key Encryption Schemes with AONT

Before we can formalize our solutions, we would like to give a new syntax on public key encryption scheme with AONT, which leads to better model practice.

Definition 3. *A public key encryption scheme with AONT as a component is an encryption scheme with following algorithms:* (K, S, E, Com, D)*, where:*

- K *is the key generation algorithm, necessarily to be randomized. It calls the key generation algorithm of a public key encryption scheme* $\mathsf{Gen\text{-}Enc}(1^k)$*, where* k *is the security parameter, and outputs a pair* (pk, sk) *of keys defining a oneway trapdoor permutation. It also pick an AONT*$= (T, I)$*, where* T *is a randomized transform algorithm, taking a message* m*, with internal randomness* r*, outputs* $y = T(m)$ *as the output;* I *is the deterministic inverse algorithm, takes a binary string* $\bar{y}$*, and return* $\bar{m} = I(\bar{y})$*.*
- S *is the deterministic plaintext split algorithm, taking* $y = T(m)$ *as input, returning two section* y_1 *and* y_2*, called secret part and public part respectively.*
- E *may be a probabilistic algorithm, calls the encryption algorithm* Enc *of a normal public key encryption with* (y_1, pk) *as input, outputs the ciphertext* c_1 *that returns by* Enc*.*
- Com *is a deterministic combine algorithm, output* $C = (c_1, y_2)$ *as the final ciphertext.*
- D *is the deterministic decryption algorithm. It first takes* $\bar{C}$ *as input, splits it into two parts:* $(\bar{c}_1$ *and* $\bar{y}_2)$*, then calls* Dec *of the public key encryption with* $(\bar{c}_1, sk)$ *as input, and gets* $\bar{y}_1$*, otherwise* $\perp$ *if "invalid" and terminates right away. It then returns* $\bar{m} = I(\bar{y}_1, \bar{y}_2)$ *as plaintext and terminates.*

4.2 Extended-Indistinguishability

We solve the first question by giving a new definition, called extended-indistin guishability, on AONT, where besides what the adversary can get in previous model, some additional side-information is given to it. The justification lies in that, the adversary not only has the resources he could in the previous game, e.g., Definition 1, additionally, it also has access to c_1, the output of the encryption component. The adversary then plays the a modified game with oracle queries. The adversary wins if it can distinguish the input of the AONT, which also turns out to be the plaintext of the whole system. Note that the side-information may

be useless for the adversary, for instance, in the case that the cryptosystem is IND-CCA secure.

We are now ready to give the new definition. Suppose an probabilistic polynomial time adversary $\mathcal{A}$ attacking a cryptosystem with AONT is engaged in the following game:

Definition 4. *At the beginning the key generation K algorithm is run, (pk, sk) are generated. The adversary schedules the attack in two phase* find *and* guess*, where it has decryption oracle access for polynomial times. At the end of* find *phase, the adversary outputs a pair of messages and writes some internal information s to its tape. An encryption oracle randomly chooses a bit b and generates the challenge ciphertext C_b. At the end of* guess *phase, the adversary outputs its guess on b. The adversary cannot query C_b on decryption oracle and an AONT has* extended-indistinguishability *if the adversary's advantage of correctly guessing b is negligible than random guess.*

$$\Pr\left[\tilde{b} = b \middle| \begin{array}{c} (pk, sk) \leftarrow K, (m_0, m_1, s) \leftarrow \mathcal{A}_{\mathsf{find}}^{\mathcal{DO}}(pk), \\ b \overset{R}{\leftarrow} \{0,1\}, y_b \leftarrow T(m_b), \\ (y_{1b}, y_{2b}) \leftarrow S(y_b), c_{1b} \leftarrow E(y_{1b}), \\ C_b \leftarrow Com(c_{1b}, y_{2b}), \tilde{b} \leftarrow \mathcal{A}_{\mathsf{guess}}^{\mathcal{DO}}(C_b, s) \end{array} \right] \leq 1/2 + neg(k)$$

Theorem 1. *Suppose a cryptosystem is integrated by an extended-indistinguishable AONT with an encryption component that is at least oneway, then the resulting cryptosystem is* IND-CCA *secure.*

Proof. From definition, obvious. □

4.3 Relations among Definitions for AONT

We briefly discuss how the new definition relates to previous definitions. Since Definition 2 completely catches the essence of Definition 1, we focus on the relation between Definition 2 and Definition 4. As we have mentioned, when AONT is combined with an IND-CCA component, there is no gap between these two definitions for an static adversary. We give a more detailed discussion here.

Suppose (T, I) is a secure AONT in the sense of Definition 2, we want to show for secret part y_1 protected by an IND-CCA secure encryption component, T is also secure in the sense of extended-indistinguishability. Actually, if this AONT is not extended-indistinguishable, an adversary $\mathcal{B}$ attacking this AONT in the sense of Definition 2 can simply be constructed as follows:

Suppose $\mathcal{A}$ is an adversary breaks extended-indistinguishability of the AONT. When $\mathcal{A}$ as for decryption queries, $\mathcal{B}$ can simply choose random c_1, together with public part y_2 complete the input message m. Since the encryption component is IND-CCA secure, which implies that for any c_1, y_1 is independent with c_1, which implies $\mathcal{B}$ simulation is perfect. Then in the end of the game, $\mathcal{B}$ outputs whatever bit b $\mathcal{A}$ outputs, thus gets the same advantage as $\mathcal{A}$. On the other hand, AONT with extended-indistinguishability is also secure under Definition 2 with similar discussion.

Remark 1. Similar analysis applies to the case of block cipher. Above analysis explains the correctness of practical schemes built on AONTs secure in the sense of previous definitions.

5 Secure Constructions

Present public key encryption primitives can be divided into two categories: the deterministic ones and the probabilistic ones. However, different treatments should be performed on these primitives respectively, because probabilistic encryption primitive requires additional randomness. If this randomness is not carefully controlled, or more exactly, if the encryption component is malleable regarding the underlying AONT, then an adversary can still create a malleable ciphertext, thus the cryptosystem is not IND-CCA secure. We give two constructions according to the types of primitives: the first is based on random permutation and suitable for deterministic encryption component. The second is based on random oracle and suitable for probabilistic encryption component. We remark that for the latter, generic construction based on non-interactive zero-knowledge proof is also capable, however, to make the ciphertext compact and computationally efficient, we adopt the random oracle.

5.1 Construction 1

The first is a Full-Domain Permutation based construction. We note the permutation is public random permutation and not oneway.

Description. Intuitively, one can think the random permutation as a bijective random oracle. A random permutation family is a family of permutations, $\pi : Keys(\pi) \times Dom(\pi) \rightarrow Rang(\pi)$, where $Dom(\pi)$ and $Rang(\pi)$ denotes the input domain space and output range space of π. Fixing each key k, $P_k : \{0,1\}^n \rightarrow \{0,1\}^n$ is a bijective mapping over the same space. By random permutation, in fact, we mean there doesn't not exist two keys k_1 and k_2, such that P_{k_1} is the same as P_{k_2}. Thus a random permutation family of domain $\{0,1\}^n$ has the key size 2^n. Since the permutation is public, given $P_k(m)$ and k, one can easily and uniquely recover

$$m = P_k^{-1}(P_k(m)).$$

The construction is very simple: for a random permutation over space $\{0,1\}^n$, where n is the size of message space, pick key k_r, and compute $P_{k_r}(m)$, then $(k_r, P_{k_r}(m))$ is an AONT with secret part k_r and public part $P_{k_r}(m)$.[1] The following theorem guarantees the security of this construction:

Theorem 2. *An AONT from Construction 1 is extended-indistinguishable.*

[1] In fact choice of places of bits to encrypt can be flexible, if sufficiently many bits are protected.

PROOF IDEA. The goal of the proof is to simulate the oracles P and P^{-1}, such that the adversary cannot distinguish this from the real oracles. In the simulation. If a new query is encountered, for P is a random permutation, we have to reply with a new random value in order to keep the simulation consistent. On simulation of decryption oracle queries, if the pre-image of the encryption component is asked, there will be a small error probability. However, we prove this is negligible, and the simulation is almost perfect. On the other hand, the challenge of m_b from the pair (m_0, m_1) is independent of the simulation, thus the adversary has no advantage. If there exist such an adversary breaks the extended-indistinguishability, then we can construct an adversary breaks the onewayness of the encryption component.

Proof. Assume there exists an adversary $\mathcal{A}$ that breaks the extended-indistin guishability of above construction. We can then construct an adversary $\mathcal{B}$ that breaks the onewayness of the encryption component denoted as φ. Namely, on input $c^* = \varphi_{pk}(r^*)$, $\mathcal{B}$ outputs r^*.

CONSTRUCTION OF $\mathcal{B}$. The key generation algorithm is run, generating (pk, sk). $\mathcal{B}$ maintains an ordered P-list of 4 data-entry (m, r, p, c) as follows:

On P query on $m \in \{0,1\}^n$ from $\mathcal{A}$ for P, $\mathcal{B}$ chooses $r \xleftarrow{R} \{0,1\}^n$, replies $p = P_r(m)$, computes the corresponding ciphertext $\varphi_{pk}(r)$ and stores $(m, r, p, \varphi_{pk}(r))$ in P-list. On P^{-1} query on $p \in \{0,1\}^n$, $\mathcal{B}$ chooses $r \xleftarrow{R} \{0,1\}^n$, replies $m = P^{-1}(p)$, and stores $m, r, p, \varphi_{pk}(r)$ in the P-list.

On decryption query on $C = (p, c)$, $\mathcal{B}$ searches in P-list whether there exists entry with (p, c). If there exists such entry, answers with m and quits. If there is no such entry, replies $m \xleftarrow{R} \{0,1\}^n$, computes $r = P^{-1}(p)$ and writes (m, r, p, c) to P-list.

On encryption oracle query with chosen messages (m_0, m_1) by $\mathcal{A}$, $\mathcal{B}$ chooses random p^*. Instead of giving correct challenge to $\mathcal{A}$, $\mathcal{B}$ takes his challenge $c^* = \varphi_{pk}(m^*)$, replies the challenge $C_b = (p^*, c^*)$.

When $\mathcal{A}$ terminates and outputs a guess b, $\mathcal{B}$ then searches in the P-list and if there is an entry (m,r,p,c) with $c = c^*$, then it outputs $m^* = m$ as the pre-image of $c^* = \varphi_{pk}(m^*)$. If $\mathcal{A}$ does not terminate in polynomial time or it encounters an error, $\mathcal{B}$ aborts the simulation and chooses random m from the list as output. Define some probability events as:

- PBad: $\mathcal{B}$ answers one P or P^{-1} query incorrectly.
- DBad: $\mathcal{B}$ answers one decryption query incorrectly.

Suppose the $\mathcal{A}$ issues Q_P direct P-oracle and P^{-1}-oracle queries and Q_D decryption queries respectively. Since P is a random permeation, then when a new entry is added to the list, it fails when there is already one entry with the same r, c in the list. Then this time, the simulation aborts. This implies the probability of failing to simulate of P or P^{-1} queries are:

$$\Pr[\mathsf{PBad}] \leq Q_P \cdot 2^{-n}$$

For $|r| = n$, the only exception on decryption query is when the corresponding ciphertext is $C = (p, \varphi_{pk}(r^*))$, that is, $\mathcal{A}$ is asking on $\varphi_{pk}(r^*)$ and gets a wrong

reply from $\mathcal{B}$, for r^* is unknown to the simulator because of onewayness of encryption component $\varphi(r^*)$. This time we have:

$$\Pr[\mathsf{DBad}] \leq (Q_P + Q_D)^2 \cdot 2^{-n}$$

Since there is $Q_P + Q_D$ elements on the P-list, the fail probability of simulation of adversary $\mathcal{B}$ denoted as $\Pr[\mathsf{BadB}]$ is given as:

$$\Pr[\mathsf{BadB}] = \Pr[\mathsf{DBad} \vee \mathsf{PBad}] \leq \Pr[\mathsf{DBad}] + \Pr[\mathsf{PBad}] \tag{1}$$

Define advantage of $\mathcal{A}$ as ε_1 and $\mathcal{B}$ as ε, which is non-negligible, since the challenge is completely independent of (m_0, m_1), since C is independent from m_0 and m_1, the success probability of $\mathcal{A}$ should be exactly $1/2$.

$$\Pr[\mathsf{SucA} \wedge \neg\mathsf{BadB}] = 1/2 \tag{2}$$

On the other hand, we have

$$\Pr[\mathsf{BadB}] \geq \Pr[\mathsf{SucA} \wedge \mathsf{BadB}] \geq \Pr[\mathsf{SucA}] - 1/2 = \varepsilon_1 \tag{3}$$

For failed simulation, if there is an entry (m, r, p, c^*) in the list there will appear a collision. In this case, the simulation fails but $\mathcal{B}$ can know it has already inverted $\varphi_{pk}(r)$. Now from (1,2,3), we have:

$$\begin{aligned} \varepsilon &\geq \Pr[\mathsf{SucA} \wedge \mathsf{BadB}] - \Pr[\mathsf{PBad}] - \Pr[\mathsf{PBad}] \\ &\geq \varepsilon_1 - Q_P \cdot 2^{-n} - (Q_P + Q_D)^2 \cdot 2^{-n} \end{aligned}$$

This implies $\mathcal{B}$ successfully inverts φ_{pk} with non-negligible probability and the execution time of $\mathcal{B}$ is within polynomial time. Proof completes. □

5.2 Construction 2

Description. For probabilistic public key encryption primitive, we would like to propose another construction based on random oracles. The second construction works as follows: G, H, H' are three hash functions treated as random oracles. For a message m, and randomness r, let the transform be

$$s = G(r) \oplus x, \quad t = H(s) \oplus r$$

which takes $y = s||t$ as output. Additionally, compute $H'(r, m)$ as the randomness used in for the probabilistic encryption component ψ. We note that only a part of y needs to be encrypted. Suppose the split algorithm works as: $y = y_1||y_2$, we require $y_1 << s$ and 2^{-y_1} is negligible. Then the probabilistic encryption component ψ takes y_1 as input, using randomness $H'(r, m)$, producing partial ciphertext $c_1 = \psi_{pk}(y_1)$.

In decryption phase, after recovering y_1, one can divide y as (s, t). Set $\bar{r} = H(s) \oplus t$ and compute $\bar{m} = s \oplus G(r)$. Check whether the ciphertext is formed correctly by computing $H'(r, \bar{m})$ and encrypt y_1 again. If this test is passed, output $\bar{m}$ as plaintext, otherwise "$\perp$".

Theorem 3. *An AONT from Construction 2 is extended-indistinguishable.*

PROOF IDEA. The idea lies in that, because of the checksum $H'(r, m)$ can be reconstructed, with re-encryption, most of the invalid decryption queries will be rejected. Thus an adversary can simulate the decryption oracle almost perfect. All correct decryption queries are "plaintext aware", in other words, the adversary gains no help from the decryption oracle. On the other hand, the adversary should simulate the random oracle queries. This is achieved by letting the adversary maintain three lists. We try to prove that the simulated oracles are in fact indistinguishable from real oracles.

Proof. From assumptions, if there exists an adversary $\mathcal{A}$ breaks the extended-indistinguishability of the AONT, another adversary $\mathcal{B}$ can be built as follows:

The key generation algorithm is run, generating (pk, sk). $\mathcal{B}$ maintains three lists, named G-list, H-list and H'-list respectively. On each random oracle query on a, $\mathcal{B}$ flips coins and selects random number as output $b \in \{0,1\}^*$. Here $\{0,1\}$ should be understood as proper length according to different contexts. $\mathcal{B}$ then write the pair (a, b) to corresponding list. We also denote the data entries in each list as: $g, G(g)$, $h, H(h)$ and $h', H'(h')$ respectively.

On answering decryption queries c_1, $\mathcal{B}$ first searches for the pair G-list and H-list, and finds pairs $g, G(g)$ and $h, H(h)$, such that $s||t = h||g \oplus H(h)$. It then sets $r = g$ and $m = h \oplus G(r)$. If there is an entry in H'-list such that $h' = r||m$, it splits $s||t$ as $y_1||y_2$, and encrypts y_1 with the public key pk to get $c_1 = \psi_{pk}(y_1)$ with $H'(h')$ as randomness for ψ_{pk}. Otherwise, it outputs "$\perp$". Denote the bit length of y_1 as k_1, length of h as k_2 and length of g as k_3.

When $\mathcal{A}$ queries the encryption oracle with two chosen messages (m_0, m_1), $\mathcal{B}$ converts the message into two sub-ciphertexts (y_{10}, y_{11}) with the same random r as its chosen message and outputs to its encryption oracle. When the challenge c_{1b} is returned by its encryption oracle, $\mathcal{B}$ selects random y_2 and completes the challenge to $\mathcal{A}$ as (c_{1b}, y_2). We can see that since y_2 is selected independent of (m_0, m_1), in fact $\mathcal{A}$ has no advantage in the game.

Obviously, the simulation random oracle query is perfect except that $\mathcal{A}$ issues a decryption query containing the real challenge c_{1b} or a random oracle query contains the real m_b. For this time, $\mathcal{B}$ cannot distinguish which one is the case.

Denote some events as:

- AskG: g is asked to G before h is asked to H.
- AskH: h is asked to H before g is asked to G.
- $\mathsf{AskH'}$: h' is asked to H' before AskG and AskH happen.
- SucA: $\mathcal{A}$ succeeds in guessing b.
- DBad: $\mathcal{B}$ fails to answer decryption query.

Suppose $\mathcal{A}$ issues Q_G and Q_H for G-oracle and H-oracle queries respectively. Also $\mathcal{A}$ issues Q_D decryption oracle queries. We can count the probability of simulation failure as follows.

$$\Pr[\mathsf{SucA}] \leq \Pr[\mathsf{AskG} \vee \mathsf{AskH} \vee \mathsf{DBad}] + 1/2 \tag{4}$$

$$\Pr[\mathsf{AskG} \vee \mathsf{AskH} \vee \mathsf{AskH'} \vee \mathsf{DBad}] \geq \Pr[\mathsf{SucA}] - 1/2 = \varepsilon_1 \tag{5}$$

For random oracle queries, by definition, $\Pr[\mathsf{AskG} \wedge \mathsf{AskH}] = 0$

$$\Pr[\mathsf{AskG} \vee \mathsf{AskH}] = \Pr[\mathsf{AskG}] + \Pr[\mathsf{AskH}]$$

For G queries, the probability of one G query "happens" to be the real challenge is 2^{-k_3} and for H to be 2^{-k_2}, accordingly. Then for total Q_G queries and Q_H queries. We have:

$$\Pr[\mathsf{AskG}] \geq 1 - (1 - 2^{-k_3})^{Q_G} = Q_G \cdot 2^{-k_3}$$
$$\Pr[\mathsf{AskH}] \geq 1 - (1 - 2^{-k_2})^{Q_H} = Q_H \cdot 2^{-k_2}$$

It is time to count the decryption oracle query. In above construction, we can see easily that similar analysis applies to failure of decryption query. Since most of the invalid queries will be rejected. We omit the details here. The probability of rejected a correctly formed ciphertext is:

$$\Pr[\neg\mathsf{DBad}] \geq 1 - (1 - 2^{-k_1})^{Q_D} = Q_D \cdot 2^{-k_1}$$

Then from Equation 5,

$$\begin{aligned}\Pr[\mathsf{AskG} \vee \mathsf{AskH} \vee \mathsf{AskH}' \vee \mathsf{DBad}] &= \Pr[\mathsf{AskG} \vee \mathsf{AskH} \vee \mathsf{AskH}'] \\ &\quad - \Pr[\mathsf{AskG} \vee \mathsf{AskH} \vee \mathsf{AskH}' \wedge \neg\mathsf{DBad}] \\ &= \Pr[\mathsf{AskH}] - (\Pr[\mathsf{AskG}] + \Pr[\mathsf{AskH}] - \Pr[\neg\mathsf{DBad}]) \\ &\leq \Pr[\mathsf{AskH}] - (Q_G \cdot 2^{-k_3} + Q_H \cdot 2^{-k_2}) \qquad (6)\end{aligned}$$

When AskH happens, $\mathcal{B}$ must have known c_{1b}, thus breaks the indistinguishability of ψ_{pk}. So we have

$$\varepsilon \geq \Pr[\mathsf{AskH}] \geq \varepsilon_1 + Q_H \cdot 2^{-k_2} + Q_G \cdot 2^{-k_3}$$

It is obvious that $\mathcal{B}$ works within polynomial time and wins the game with non-negligible advantage. This completes the proof. □

6 Conclusion

A "secure" AONT of previous definitions may not yield CCA secure cryptosystem. Our new definition on AONT abstracts the essential nature of AONT when used in a practical cryptosystem. Moreover, we give concrete constructions of extended-indistinguishable AONT according to different types of encryption primitive. We remark that this justifying is important in designing real life system.

References

1. J. H. An and Y. Dodis. Concealment and Its Applications to Authenticated Encryption. In *Eurocrypt'03*, volume 2656 of *LNCS*, pages 312–329. Springer-Verlag, 2003.

2. M. Bellare, A. Desai, D. Pointcheval, and P. Rogway. Relations among Notions of Security for Public-Key Encryption Schemes. In *Crypto'98*, volume 1462 of *LNCS*. Springer-Verlag, 1998.
3. M. Bellare and C. Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In *Asiacrypt'00*, volume 1976 of *LNCS*. Springer-Verlag, 2000.
4. M. Bellare and P. Rogaway. Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. In *1st ACM Conference on Computer and Communications*, pages 62–73. ACM Press, 1993.
5. M. Bellare and P. Rogaway. Optimal Asymmetric Encryption. In *Eurocrypt'94*, volume 590 of *LNCS*, pages 92–111. Springer-Verlag, 1994.
6. V. Boyko. On the Security Properties of the OAEP as an All-or-Nothing Transform. In *Crypto'99*, volume 1666 of *LNCS*, pages 503–518. Springer-Verlag, 1999.
7. R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-Resilient Functions and All-or-Nothing Transforms. In *Eurocrypt'00*, volume 1807 of *LNCS*, pages 453–469. Springer-Verlag, 2000.
8. A. Desai. The Security of All-or-Nothing Encryption: Protecting against Exhaustive Key Search. In *Crypto'00*, volume 1880 of *LNCS*, pages 359–375. Springer-Verlag, 2000.
9. Y. Dodis, J. Katz, S. Xu, and M. Yung. Key-insulated Public Key Cryptosystems. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 65–82. Springer-Verlag, 2002.
10. D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. In *STOC'91*. ACM, 1991.
11. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto'86*, volume 263 of *LNCS*, pages 186–194. Springer-Verlag, 1987.
12. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Science*, 28(2):270–299, 1984.
13. M. Jakobsson, J. Stern, and M. Yung. Scramble All, Encrypt Small. In *FSE'99*, volume 1636 of *LNCS*, pages 95–111. Springer-Verlag, 1999.
14. D. Johnson, S. Matyas, and M. Peyravian. Encryption of Long Blocks Using a Short-Block Encryption Procedure. Available at: `http://grouper.ieee.org/groups/1363/contributinos/peyrav.ps`, 1997.
15. M. Naor and M. Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *STOC'90*, pages 427–437. ACM, 1990.
16. C. Rackoff and D. Simon. Noninteractive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *Advances in Cryptology-Crypto'91*, volume 576 of *LNCS*, pages 433–444. Springer-Verlag, 1991.
17. R.L. Rivest. All-Or-Nothing Encryption and The Package Transform. In *FSE'97*, volume 1267 of *LNCS*, pages 210–218. Springer-Verlag, 1997.
18. V. Shoup. Why Chosen Ciphertext Security Matters. Technical Report RZ 3076, IBM Research, 1998.
19. V. Shoup. OAEP Reconsidered. In *Crypto'01*, volume 2139 of *LNCS*, pages 239–259, 2001.
20. D.R. Stinson. Some Considerations on All-Or-Nothing Transforms. Available at: `http://cacr.math.uwaterloo.ca/~dstinson/papers/AON.ps`, 1998.
21. R. Zhang, G. Hanaoka, J. Shikata, and H. Imai. On the Security of Multiple Encryption or CCA-security+CCA-security=CCA-security? In *PKC'04*, volume 2947 of *LNCS*, pages 360–374. Springer-Verlag, 2004.