PAPER

Post-Challenge Leakage Resilient Public-Key Cryptosystem in Split State Model

Eiichiro FUJISAKI[†a)], **Akinori KAWACHI**[††], **Ryo NISHIMAKI**[†], **Keisuke TANAKA**[††], *and* **Kenji YASUNAGA**[†††], ***Members***

SUMMARY Leakage resilient cryptography is often considered in the presence of a very strong leakage oracle: An adversary may submit *arbitrary* efficiently computable function f to the leakage oracle to receive $f(x)$, where x denotes the entire secret that a party possesses. This model is somewhat too strong in the setting of public-key encryption (PKE). It is known that no secret-key leakage resilient PKE scheme exists if the adversary may have access to the secret-key leakage oracle to receive only *one bit* after it was given the challenge ciphertext. Similarly, there exists no sender-randomness leakage resilient PKE scheme if *one-bit* leakage occurs after the target public key was given to the adversary. At TCC 2011, Halevi and Lin have broken the barrier of *after-the-fact leakage*, by proposing the so-called split state model, where a secret key of a party is explicitly divided into at least two pieces, and the adversary may have not access to the *entire* secret at once, but each divided pieces, one by one. In the split-state model, they have constructed post-challenge secret-key leakage resilient CPA secure PKEs from hash proof systems, but the construction of CCA secure post-challenge secret-key leakage PKE has remained open. They have also remained open to construct sender-randomness leakage PKE in the split state model. This paper provides a solution to the open issues. We also note that the proposal of Halevi and Lin is post-challenge secret-key leakage CPA secure against a *single* challenge ciphertext; not against *multiple* challenges. We present an efficient generic construction that converts any CCA secure PKE scheme into a *multiple-challenge CCA* secure PKE that simultaneously tolerates *post-challenge* secret-key and sender-randomness leakage in the split state model, *without any additional assumption.* In addition, our leakage amount of the resulting schemes is the *same* as that of Halevi and Lin CPA PKE, i.e., $(1/2+\gamma)\ell/2$ where ℓ denotes the length of the entire secret (key or randomness) and γ denotes a universal (possitive) constant less than $1/2$. Our conversion is generic and available for many other public-key primitives. For instance, it can convert any identity-based encryption (IBE) scheme to a post-challenge master-key leakage and sender-randomness leakage secure IBE.

key words: *post-challenge (bounded) leakage, simultaneous secret-key and sender-randomness leakage, CCA2 security for multiple messages*

1. Introduction

Recently, leakage resilient cryptography has emerged (e.g., [1], [7], [13], [15], [19], [20], [22], [24]). One of major leakage modelings is of memory leakage attacks [1], which was inspired by the attacks provided by Halderman et al. [16], where they showed that it might not be difficult to retrieve residual data from a computer's memory. They demonstrated that ordinary DRAMs lose their contents gradually over a period of seconds when they lose power and succeeded in recovering a significant fraction of the bits of a cryptographic key. Their attack suggests that an adversary might be able to obtain any intermediate state of a cryptographic task with secret information. Memory leakage attacks capture this intuition, in which an adversary may have (adaptively) access to leakage oracles with any efficiently computable function f to obtain $f(x)$ from the leakage oracles, where x is the target secret. The only constraint is that the total amount of $f(x)$'s is bounded by some parameter, which is of course less than the size of secrets.

This model is, however, somewhat too strong in public-key encryption (PKE). It is known that no secret-key leakage resilient PKE scheme exists if the adversary may have access to the leakage oracle and receive only one bit information on the target secret-key after it saw the challenge ciphertext — Consider the following strategy of an adversary in the IND-CPA game in the presence of post-challenge memory leakage. The adversary sets $m_0 = 0^k$ and $m_1 = 1^k$, and feeds them to the challenger, who encrypts either of them, $c = \mathsf{Enc}_{pk}(m_b)$, where $b \in \{0,1\}$. After receiving c, the adversary submits function $f(\cdot) := \mathsf{lsb}(\mathsf{Dec}_{(\cdot)}(c))$ to obtain $f(sk)$, the least significant bit of the decryption of c under sk. Finally he outputs $f(sk)$ and wins, because $b = f(sk)$.

Similarly, there exists no sender-randomness leakage resilient PKE scheme if one-bit leakage occurs after the target public key is given to the adversary. In this case, the adversary instead selects $i \in \{1,\dots,|c|\}$ at random and sends $f(\cdot) := [\mathsf{Enc}_{pk}(m_0;\cdot)](i)$, where $[y](i)$ denotes the i-th bit of string y. Since $\mathsf{Enc}_{pk}(m_0;r) \neq \mathsf{Enc}_{pk}(m_1;r')$ for any r, r', and any m_0, m_1, s.t. $m_0 \neq m_1$, the adversary can win the game at least $\frac{1}{2}(1+\frac{1}{|c|})$, which is significant.

At TCC 2011, to bypass the impossibility results, Halevi and Lin [17] proposed a new restricted bounded memory leakage model, called the split state model. They considered a special type of PKEs such that the secret key consists of a pair of two strings, sk_1 and sk_2. The leakage occurs on each split secret, not both of them at the same time. An adversary may still have access to each leakage oracle, $\mathcal{L}_{sk_i}$, adaptively. Since the adversary may adaptively query, the adversary may submit a query function of sk_1 to $\mathcal{L}_{sk_1}$, which depends on some leakage of sk_2 that he has already obtained from $\mathcal{L}_{sk_2}$. However, he cannot directly ask with function $F(\cdot,\cdot)$ so that he can obtain $F(sk_1, sk_2)$. In this two-split model, Halevi and Lin presented a construc-

Manuscript received October 22, 2014.

[†]The authors are with NTT Secure Platform Laboratories, NTT Corporation, Musashino-shi, 180-8585 Japan.
[††]The authors are with Tokyo Institute of Technology, Tokyo, 152-8550 Japan.
[†††]The author is with Kanazawa University, Kanazawa-shi, 920-1192 Japan.
a) E-mail: fujisaki.eiichiro@lab.ntt.co.jp

DOI: 10.1587/transfun.E98.A.853

tion of post-challenge secret-key leakage resilient CPA secure PKEs from hash proof systems, but they remained it open how to construct CCA2 secure PKEs resilient to post-challenge key leakage in the two split state model. Additionally, they also mentioned that it was an open problem to find a way of simultaneously addressing secret-key and sender-randomness leakage.

In addition, we note that although their scheme [17] is post-challenge leakage resilient, it is not (proven) post-challenge leakage resilient against multiple challenge ciphertexts.

1.1 Our Contribution

In this work, we present a generic construction of a CCA2 secure PKE scheme that is simultaneously resilient to post-challenge secret-key and sender-randomness leakage from any CCA2 secure PKE scheme in the two split state model, which solves the open issues suggested by Halevi and Lin [17]. The scheme of Halevi and Lin is post-challenge secret-key leakage CPA secure against a *single* challenge ciphertext; not against *multiple* challenges, whereas our resulting scheme is CCA2 secure for multiple challenge messages simultaneously resilient to secret-key and sender-randomness leakage. We start with an arbitrary CCA2 secure PKE and convert it to the above post-challenge leakage secure scheme *without any additional assumption* with leakage rates equivalent to that of Halevi and Lin [17], that is, $\frac{1}{2}(\frac{1}{2}+\gamma)$, where γ is a universal (positive) constant less than $1/2$.

The idea behind the construction comes from the following observation: There is a two-source extractor $\mathsf{Ext2} : (\{0,1\}^t)^2 \to \{0,1\}^m$, such that its output is statistically indistinguishable from a truly random string within statistical distance of $2^{-\Omega(m)}$ if $x_1, x_2 \in \{0,1\}^t$ are mutually independent and the entropy of each random string is at least $(1/2-\gamma)t$ for a universal constant $0 < \gamma < 1/2$ [6]. We then show that the output of the two-source extractor still looks random against any unbounded adversary except for a negligible probability, *even if leakage occurs after the adversary is given the output of two-source extractor*, assuming that the statistical distance of the two-source extractor is exponentially small. We then replace inner random strings of the key generation algorithm with the outputs of the two-source extractors. Similarly, we replace each random string used in encrypting every message by a sender with the output of the two-source extractors. If the starting PKE scheme is CCA2 secure, the resulting scheme is also CCA2 secure for multiple challenge messages and resilient simultaneously to post-challenge key and randomness leakage. Our resulting scheme is resilient to leakage of $(1/2+\gamma)\ell_{sk}/2$ and $(1/2+\gamma)\ell_{rd}/2$, where ℓ_{sk} is the length of the entire secret key and ℓ_{rd} is the length of the random string used in encrypting a message by a sender.

We note that the idea of employing the two-source extractors resilient to post-challenge leakage is not new. Halevi and Lin [17] implicitly use them, but we explicitly define such an extractor as a module and apply it to cryptographic primitives in a black-box way. By this, one could be easily aware of the essence of the idea of [17] and notice a smarter usage that can be applied to other applications.

1.2 Related Work

Micali and Reyzin [22] formalized a general framework for modeling side-channel attacks. In their work, they assume that there exists a leakage-free hardware and only the computation leaks information. Inspired by [16], Akavia et al. [1] formalized a stronger model of (key) memory leakage attacks, where no leakage-free hardware was assumed, and showed that Regev's lattice-based scheme [26] is secure against memory leakage attacks. Naor and Segev [24] extended the notion of [1] and presented a general construction of a PKE scheme resilient to key leakage from any universal hash proof system [9]. Due to the impossibility result mentioned above, the resulting schemes are resilient only to pre-challenge key leakage.

Several papers addressed leakage of randomness used in an encryption algorithm. Bellare et al. [3] studied the security of PKE in the case that a random string used in the encryption algorithm by a sender is not uniformly random. However, leakage functions for the random string is independent of public keys. Therefore, the work does not tackle post-public key leakage.

Namiki et al. [23] addressed randomness leakage in the KEM-DEM framework, in which an adversary may have access to KEM's leakage oracle only before he takes the (whole) ciphertext, whereas he may have access to DEM's leakage oracle only after the ciphertext is given. Although their model is clearly stronger than the split state model, the resulting scheme bypasses the impossibility result in some sense. We note that the resulting scheme is only CPA secure.

Birrel et al. [5] also studied the leakage of random strings used for encryption in the context of randomness dependent message security. They considered situations where leakage functions can depend on the public key but the size of the functions must be *a priori* bounded by some polynomial.

Several variants of the split state model have recently appeared, e.g., [10], [21], in order to bypass impossibility results or intractably difficulty problems.

Independent of us, Zhang, Chow, and Cao [27] presented post-challenge secret-key leakage PKE. Their scheme is an extention of Halevi and Lin scheme [17], by replacing entropic leakage-resilient CPA PKEs with entropic leakage-resilient CCA ones. We note that they only provided an inefficient instantiation of entropic leakage-resilient CCA PKEs using general non-interactive zero-knowledge proofs, which combines entropic leakage-resilient CPA PKE with another CPA PKE in the Naor-Yung double encryption paradigm.

2. Definitions and Tools

(1) Notations.

For random variable or distribution D, we write $y \overset{\mathsf{R}}{\leftarrow} D$ to denote that y is randomly chosen from D according to its distribution. For set S, $x \overset{\mathsf{U}}{\leftarrow} S$ denotes that x is uniformly chosen from S. $y := z$ denotes the operation of assigning the value of z to the variable y. We say that function $f : \mathbb{N} \to \mathbb{R}$ is negligible in $\lambda \in \mathbb{N}$ if for every constant $c \in \mathbb{N}$ there exists $k_c \in \mathbb{N}$ such that $f(\lambda) < \lambda^{-c}$ for any $\lambda > k_c$. Hereafter, we use $f = \mathsf{negl}(\lambda)$ to mean that f is negligible in λ. The statistical distance between two random variables, X and Y, over a finite domain Ω is defined as $\Delta(X, Y) \triangleq \frac{1}{2}\sum_{\omega\in\Omega} |\Pr[X = \omega]-\Pr[Y = \omega]|$. Let $\mathcal{X} = \{X_\lambda\}_{\lambda\in\mathbb{N}}$ and $\mathcal{Y} = \{Y_\lambda\}_{\lambda\in\mathbb{N}}$ denote two ensembles of random variables indexed by λ. For string x, $|x|$ denotes the length of x. We write PPT and DPT algorithms to denote probabilistic polynomial-time and deterministic poly-time algorithms, respectively.

Definition 2.1: We say that $\mathcal{X}$ and $\mathcal{Y}$ are statistically indistinguishable if $\Delta(X_\lambda, Y_\lambda) = \mathsf{negl}(\lambda)$.

Definition 2.2: We say that $\mathcal{X}$ and $\mathcal{Y}$ are computationally indistinguishable if for every non-uniform PPT algorithm D,

$$|\Pr[D(X_\lambda) = 1] - \Pr[D(Y_\lambda) = 1]| = \mathsf{negl}(\lambda).$$

We write $\mathcal{X} \overset{\mathsf{c}}{\approx} \mathcal{Y}$ (resp., $\mathcal{X} \overset{\mathsf{s}}{\approx} \mathcal{Y}$) to denote that $\mathcal{X}$ and $\mathcal{Y}$ are computationally (resp., statistically) indistinguishable. Let U_ℓ denote the ℓ-bit uniform distribution and $\mathbf{E}[X]$ denote the expectation of random variable X.

(2) Average Min-Entropy.

Let X and Y be random variables defined over domain Ω. We define the min-entropy of random variable X as $H_\infty(X) \triangleq -\log(\max_{x\in\Omega}\Pr[X = x])$. The notion of the average min-entropy is defined by Dodis, Ostrovsky, Reyzin, and Smith [11]. The average min-entropy of random variable X given random variable Y is interpreted as the remaining unpredictability of X conditioned on the value of Y:

$$\widetilde{H}_\infty(X|Y) \triangleq -\log\left(\mathop{\mathbf{E}}_{y\overset{\mathsf{R}}{\leftarrow}Y}[2^{-H_\infty(X|Y=y)}]\right)$$
$$= -\log\left(\mathop{\mathbf{E}}_{y\overset{\mathsf{R}}{\leftarrow}Y}\left[\max_x \Pr[X = x|Y = y]\right]\right).$$

Dodis et al. showed the following lemma:

Lemma 2.3 ([11]): When Y takes at most 2^r possible values and Z is any random variable, then

$$\widetilde{H}_\infty(X|(Y, Z)) \geq \widetilde{H}_\infty(X|Z) - r.$$

2.1 Two-Source Extractors

Our main technical tool is the so-called two-source extractors.

Definition 2.4 (Worst-case-2-source extractor): A function $\mathsf{Ext2} : \{0, 1\}^t \times \{0, 1\}^t \to \{0, 1\}^m$ is a worst-case (v, ϵ)-two-source extractor if for independent random variables $X, Y \in \{0, 1\}^t$, with $H_\infty(X), H_\infty(Y) \geq v$, it holds that $\Delta(\mathsf{Ext2}(X, Y), U_m) \leq \epsilon$.

Definition 2.5 (Average-case 2-source extractor): A function $\mathsf{Ext2} : \{0, 1\}^t \times \{0, 1\}^t \to \{0, 1\}^m$ is an average-case (v, ϵ)-two-source extractor if for all random variables $X, Y \in \{0, 1\}^t$ and Z, such that, conditioned on Z, X and Y are independent and have average min-entropy v, it holds that $\Delta((\mathsf{Ext2}(X, Y), Z), (U_m, Z)) \leq \epsilon$.

The next lemma follows from the same argument of Lemma 2.3 in [11], which transforms a worst-case two-source extractor to an average-case one.

Lemma 2.6: For any $\delta > 0$, if $\mathsf{Ext2} : \{0, 1\}^t \times \{0, 1\}^t \to \{0, 1\}^m$ is a worst-case $(v - \log(1/\delta), \epsilon)$-two-source extractor, then $\mathsf{Ext2}$ is an average-case $(v, \epsilon + 2\delta)$-two-source extractor.

The following theorem give us an explicit construction of a (worst-case) two-source extractor.

Theorem 2.7 ([6]): There exists a universal constant $\gamma < \frac{1}{2}$ and a polynomial-time computable (worst-case) (v, ϵ)-two-source extractor $\mathsf{Ext2} : \{0, 1\}^t \times \{0, 1\}^t \to \{0, 1\}^m$ such that $v = (1/2 - \gamma)t$, $\epsilon = 2^{-\Omega(m)}$ and $m = \Omega(t)$.

By Lemma 2.6, we immediately obtain an average two-source extractor.

We note that Bourgain's two-source extractor takes two independent sources, each of with has min-entropy rate less than 1/2, i.e., $(1/2-\gamma)$, and outputs almost a uniform string. The universal constant $0 < \gamma < 1/2$ is determined by the construction of [6]. We refer the reader to [25] for more details.

We prepare the following useful lemmas.

Lemma 2.8: Let X, Y be random variables over Ω and $f : \Omega \to \mathbb{R}$ be an arbitrary function. Then, $\Delta(f(X), f(Y)) \leq \Delta(X, Y)$.

Proof. The proof follows from the triangle inequality, $|\Pr[f(X) = \alpha] - \Pr[f(Y) = \alpha]| \leq \sum_{\omega\in f^{-1}(\alpha)} |\Pr[X = \omega] - \Pr[Y = \omega]|$. □

Lemma 2.9 ([12]): Let A, B be independent random variables. Consider a sequence $V_1, \ldots, V_m$ of random variables, where $V_i = \phi(V_1, \ldots, V_{i-1}, C_i)$ for some function ϕ, where C_i is either A or B. Then A and B are independent conditioned on $V_1, \ldots, V_m$.

2.2 Public-Key Encryption

(3) PKE.

A public-key encryption (PKE) scheme PKE consists of three algorithms $\mathsf{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ satisfying the following properties. The key-generation algorithm Gen is a PPT algorithm that takes as input security parameter 1^λ and outputs a public/secret key pair, that is, $(pk, sk) \overset{\mathsf{R}}{\leftarrow} \mathsf{Gen}(1^\lambda)$. The public-key encryption algorithm Enc is a PPT algorithm that takes as input pk and message $m \in \{0,1\}^*$, picks up inner randomness r, and outputs ciphertext $c \leftarrow \mathsf{Enc}(pk, m)$. We write $c := \mathsf{Enc}(pk, m; r)$ when we want to clarify randomness r used in Enc. The decryption algorithm Dec is a deterministic algorithm that takes as input sk and c, and outputs plaintext m'. That is, $m' := \mathsf{Dec}(sk, c)$. It is required that $m = \mathsf{Dec}(sk, \mathsf{Enc}(pk, m))$ for $\forall\lambda \in \mathbb{N}$, $\forall(pk, sk) \in \mathsf{Gen}(1^\lambda)$, and $\forall m \in \{0,1\}^*$.

We use the standard security notion of IND-CCA2 [4] for PKE as CCA security.

(4) Split-State PKE.

As Akavia, Goldwasser, and Vaikuntanathan [1] pointed out, there is no secret-key leakage resilient PKE scheme if the adversary is allowed to query leakage functions after it can obtain the target ciphertext. To bypass the impossibility result, we use the split state model [17].

We say that PKE = $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ has two-split-state with respects to secret-key (resp. randomness) if there is a partition that divides secret key sk (resp. randomness r) into $sk = (sk_1, sk_2)$ (resp. $r = (r_1, r_2)$). From now, two-split-state PKE scheme denotes an arbitrary PKE scheme with two-split states with respects to secret key and randomness.

Remark 2.10: We note that the decryption algorithm of the split-state PKE schemes defined in [17] is syntactically different from ours, where the decryption algorithm is a pair of algorithms $(\mathsf{Dec}_1, \mathsf{Dec}_2, \mathsf{Comb})$ such that Dec_i $(i = 1, 2)$ takes partial secret key sk_i and ciphertext c and outputs s_i, and Comb takes s_1, s_2 and ciphertext c, and outputs message m. To tailor ours to the original one, we simply define Dec_i and Comb such that Dec_i is a dummy algorithm that takes (sk_i, c) and outputs $s_i := sk_i$ and Comb takes (s_1, s_2, c) and outputs $\mathsf{Dec}(sk, c)$.

2.3 Post-Challenge Key and Randomness Leakage Security for Split-State PKE

We now model post-challenge secret-key and sender-randomness leakage CCA security for split-state PKE schemes, in the *multiple* challenge messages setting, in the split state model. In this model, the leakage occurs on each divided secret one by one, not both of them at the same time. We note that the adversary can still obtain leakage related to both sk_1 and sk_2, because leakage query function f_s on sk_2 may depend on leakage on sk_1. So does randomness leakage.

We write $\mathcal{L}_s$ to denote the leakage oracle on secret-key that takes query (i, f_s), where $i \in \{1, 2\}$ and f_s is an efficiently computable function, and returns $f_s(sk_i)$. Similarly, we write $\mathcal{L}_r$ to denote the leakage oracle on randomness that takes query (i, f_r), where $i \in \{1, 2\}$ and f_r is an efficiently computable function, and returns $f_r(r_i)$.

We consider the following game between the challenger and adversary $\mathcal{A}$.

1. The challenger runs $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$ where $sk = (sk_1, sk_2)$ and feeds pk to $\mathcal{A}$.
2. **(Pre-Challenge Queries)** $\mathcal{A}$ submits a polynomial number of following queries:
   - **(Secret-Key Leakage Query)** When $\mathcal{A}$ submits (i, f_s) with $i \in \{1, 2\}$ to $\mathcal{L}_{sk}$, the challenger replies with $f_s(sk_i)$.
   - **(Decryption Query)** When $\mathcal{A}$ submits ciphertext c to the decryption oracle $\mathsf{Dec}(sk, \cdot)$, the challenger replies with $\mathsf{Dec}(sk, c)$.
3. **(ℓ-Randomness-Dependent Message)** $\mathcal{A}$ outputs polynomial (in λ) ℓ. The challenger then chooses ℓ independent random strings, $\boldsymbol{r} = (r^{(1)}, \dots, r^{(\ell)})$, where $r^{(k)} = (r_1^{(k)}, r_2^{(k)})$. We note that random string $r^{(k)}$ is later used to encrypt challenge message $m_b^{(k)}$. $\mathcal{A}$ now submits a polynomial number of following queries:
   - **(Secret-Key Leakage Query)** When $\mathcal{A}$ submits (i, f_s) with $i \in \{1, 2\}$ to $\mathcal{L}_s$, the challenger replies with $f_s(sk_i)$.
   - **(Randomness Leakage Query)** When $\mathcal{A}$ submits $((i, j), f_r)$ with $i \in \{1, 2\}$ and $j \in [\ell]$ to $\mathcal{L}_r$, the challenger replies with $f_r(r_i^{(j)})$.
   - **(Decryption Query)** When $\mathcal{A}$ submits ciphertext c to the decryption oracle $\mathsf{Dec}(sk, \cdot)$, the challenger replies with $\mathbf{D}(sk, c)$.
4. $\mathcal{A}$ outputs two vectors of ℓ challenge messages $(\boldsymbol{m}_0, \boldsymbol{m}_1)$ where $\boldsymbol{m}_b = (m_b^{(1)}, \dots, m_b^{(\ell)})$ for $b \in \{0, 1\}$ such that $|m_0^{(k)}| = |m_1^{(k)}|$ for any $k \in [\ell]$.
5. **(Challenge Ciphertexts for Multiple Messages)** The challenger picks up a random bit $b \overset{\mathsf{U}}{\leftarrow} \{0, 1\}$. It computes a vector of challenge ciphertexts, $\boldsymbol{c}^* := \mathsf{Enc}(pk, \boldsymbol{m}_b; \boldsymbol{r})$, where $\mathsf{Enc}(pk, \boldsymbol{m}_b; \boldsymbol{r})$ denotes $(\mathsf{Enc}(pk, m_b^{(1)}; r^{(1)}), \dots, \mathsf{Enc}(pk, m_b^{(\ell)}; r^{(\ell)}))$. It then sends $\boldsymbol{c}^*$ to $\mathcal{A}$.
6. **(Post-Challenge Queries)** $\mathcal{A}$ submits a polynomial number of following queries:
   - **(Secret-Key Leakage Query)** When $\mathcal{A}$ submits (i, f_s) with $i \in \{1, 2\}$ to $\mathcal{L}_s$, the challenger replies with $f_s(sk_i)$.
   - **(Randomness Leakage Query)** When $\mathcal{A}$ submits $((i, j), f_r)$ with $i \in \{1, 2\}$ and $j \in [\ell]$ to $\mathcal{L}_r$, the challenger replies with $f_r(r_i^{(j)})$.
   - **(Decryption Query)** When $\mathcal{A}$ submits ciphertext c to the decryption oracle $\mathsf{Dec}(sk, \cdot)$, the challenger replies with $\mathbf{D}(sk, c)$. We do not allow $\mathcal{A}$ to submit any challenge ciphertext $c^* \in \boldsymbol{c}^*$.
7. $\mathcal{A}$ outputs bit b'.

The advantage of $\mathcal{A}$ in the above game is defined as $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{KR\text{-}postLR\text{-}CCA2}}(\lambda) \triangleq 2\Pr[b' = b] - 1$ for security parameter λ. Let L_s be the length of the total amount of leaked secret-key information,

$$L_s \triangleq \sum_{j \in [q_s]} |f_s^{(j)}(sk_i)|,$$

where q_s denotes the maximum number of queries of $\mathcal{A}$ to $\mathcal{L}_s$. Let $L_r^{(k)}$ be the length of the total amount of leaked information from k-th randomness $r^{(k)}$, where $k \in [\ell]$.

$$L_r^{(k)} \triangleq \sum_{j\in[q_r]} |f_r^{(j)}(r_i^{(k)})|,$$

where q_r denotes the maximum number of queries of $\mathcal{A}$ to $\mathcal{L}_r$.

We say that a split-state PKE is (L_s, L_r)-KR-postLR-CCA2 secure for multiple messages in the split-state model, if for all PPT algorithm $\mathcal{A}$ that may receives secret-key leakage of at most L_s bit and sender-randomness leakage of at most L_r bit on *each randomness* $r^{(k)}$ (i.e., $L_r^{(k)} \le L_r$ for all $k \in [\ell]$), it holds that $\mathsf{Adv}_{\mathcal{A}}^{\text{KR-postLR-CCA2}}(\lambda) = \mathsf{negl}(\lambda)$.

Remark 2.11 (Multi-Message Security): We note that in the standard PKE, single-message (semantic) security implies multiple-message security [14]. It is not the case of post-challenge leakage PKEs. Indeed, the proof of security for Halevi and Lin [17] fails if an adversary is allowed to choose multiple challenge messages.

Remark 2.12 (Randomness-Dependent Messages [5]): The above game stipulates the CCA attack game for multiple-challenge messages of split-state PKEs in the presence of pre/post-challenge secret-key and sender-randomness leakage. Our definition allows adversary $\mathcal{A}$ to have access to the randomness leakage oracle before it chooses challenge messages, so that it may choose messages *dependent on random strings* used in encryption, i.e., randomness-dependent security [5].

Remark 2.13: In the above definition, simultaneous leakage on (partial) secret-key and randomness, such as $F(sk_i, r_j^{(k)})$, is not assumed. One reason for this is that a sender (an encryptor) and a receiver (a decryptor) would be different in the most cases. We note that even if such simultaneous leakage is allowed, security of our proposed scheme is not affected.

3. Two-Source Extractor Resilient to Pre/Post-Challenge Leakage

We consider two-source extractors resilient to pre/post-challenge leakage, which play an important role in the security proof of our scheme.

Let $\mathsf{Ext2} : \{0,1\}^t \times \{0,1\}^t \to \{0,1\}^m$ be a worst-case (v, ϵ)-two-source extractor. Let $\boldsymbol{x} = (x_1, x_2)$ where $x_1, x_2 \in \{0,1\}^t$ and $\mathsf{Ext2}(\boldsymbol{x}) = \mathsf{Ext2}(x_1, x_2)$. We now consider the following game G_{Ext2}.

1. The challenger picks up $x_1, x_2 \overset{\mathsf{U}}{\leftarrow} \{0,1\}^t$, independently. It also chooses $w_0 \overset{\mathsf{U}}{\leftarrow} \{0,1\}^m$ and sets $\boldsymbol{x} := (x_1, x_2)$ and $w_1 := \mathsf{Ext2}(x_1, x_2)$. It then runs adversary $\mathcal{A}$.
2. (Pre-Challenge Stage) $\mathcal{A}$ may adaptively submit an a-priori unbounded polynomial number of queries with the form of function $(i, f_{\mathsf{Pre}}(\cdot))$ (where $i \in \{1,2\}$). For query $(i, f_{\mathsf{Pre}}(\cdot))$, the challenger returns $f_{\mathsf{Pre}}(x_i)$.
3. $\mathcal{A}$ then asks for the challenge. The challenger then picks up $b \overset{\mathsf{U}}{\leftarrow} \{0,1\}$ and sends w_b to $\mathcal{A}$.
4. (Post-Challenge Stage) $\mathcal{A}$ may adaptively submit an a-priori unbounded polynomial number of queries with the form of function $(i, f_{\mathsf{Post}}(\cdot))$ (where $i \in \{1,2\}$), and the challenger returns $f_{\mathsf{Post}}(x_i)$.
5. $\mathcal{A}$ finally outputs a bit b' and wins if $b' = b$.

The advantage of $\mathcal{A}$ in the above game is defined as $\mathsf{Adv}_{\mathcal{A}}^{(\mathsf{L_{Pre}},\mathsf{L_{Post}})}(m) := 2\Pr[b' = b] - 1$, where L_{Pre} denotes the bit length of the total amount of leaked information in the pre-challenge stage, i.e., $L_{\mathsf{Pre}} \triangleq \sum_j |f_{\mathsf{Pre}}^{(j)}(x_i)|$ where $f_{\mathsf{Pre}}^{(j)}(x_i)$ denotes the answer of the challenger for the j-th query in the pre-challenge stage, and L_{Post} denotes the bit length of the total amount of leaked information in the post-challenge stage, i.e., $L_{\mathsf{Post}} \triangleq \sum_j |f_{\mathsf{Post}}^{(j)}(x_i)|$ where $f_{\mathsf{Post}}^{(j)}(x_i)$ denotes the answer of the challenger for the j-th query in the post-challenge stage. We call the above game the pre-challenge leakage game if $L_{\mathsf{Post}} = 0$. On the contrary, we call it the post-challenge leakage game if $L_{\mathsf{Post}} > 0$.

The advantage of $\mathcal{A}$ in the pre-challenge leakage game is defined as $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Pre}}(m) \triangleq \mathsf{Adv}_{\mathcal{A}}^{(\mathsf{L_{Pre}},0)}(m)$.

Similarly, the advantage of $\mathcal{A}$ in the post-challenge leakage game is defined as $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Post}}(m) \triangleq \mathsf{Adv}_{\mathcal{A}}^{(\mathsf{L_{Pre}},\mathsf{L_{Post}})}(m)$ with $L_{\mathsf{Post}} > 0$.

We show in the following theorems that the two-source extractor given in [6] is actually resilient to the pre/post-challenge leakage games.

Theorem 3.1: There exist constant γ, with $0 < \gamma < 1/2$, and polynomial-time computable two-source extractor $\mathsf{Ext2} : \{0,1\}^t \times \{0,1\}^t \to \{0,1\}^m$, such that, for every sufficiently large t and for any *unbounded* distinguisher $\mathcal{A}$, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Pre}}(m) \le 2^{-3m/2}$ as long as $L_{\mathsf{Pre}} \le (1/2 + \gamma)t$. In addition, constant γ is very close to a universal constant given by Bourgain's extractor [6].

Proof. Let $\mathsf{Ext2} : \{0,1\}^t \times \{0,1\}^t \to \{0,1\}^m$ be a worst-case (v, ϵ)-two-source extractor presented in Theorem 2.7, where $v = (1/2 - \gamma')t$ for a universal constant $\gamma' < 1/2$. It follows from Lemma 2.6 that $\mathsf{Ext2}$ is an average-case $(v + \log(1/\alpha), \epsilon + 2\alpha)$-two-source extractor for any $\alpha > 0$.

Let $\boldsymbol{f}(\boldsymbol{x})$ denote the sequence of the responses of the challenger for $\mathcal{A}$'s queries (in the pre-challenge stage), that is, $\boldsymbol{f}(\boldsymbol{x}) = (f_{\mathsf{Pre}}^{(1)}(x_{i_1}), f_{\mathsf{Pre}}^{(1)}(x_{i_2}), \ldots)$ where $i_1, i_2, \ldots \in \{1,2\}$. By Lemma 2.3, the average min-entropy of x_1 and x_2 given $\boldsymbol{f}(\boldsymbol{x})$ is at least $t - L_{\mathsf{Pre}}$, because $\widetilde{H}_\infty(x_i|\boldsymbol{f}(\boldsymbol{x})) \ge \widetilde{H}_\infty(x_i) - L_{\mathsf{Pre}} = t - L_{\mathsf{Pre}}$ for $i \in \{1,2\}$. In addition, secret x_1 and x_2 are still independent conditioned on $\boldsymbol{f}(\boldsymbol{x})$, due to Lemma 2.9. Therefore, it holds that

$$\Delta((\mathsf{Ext2}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{x})), (U_m, \boldsymbol{f}(\boldsymbol{x}))) \le \epsilon + 2\alpha,$$

as long as $t - L_{\mathsf{Pre}} \ge v + \log(1/\alpha)$, because $\mathsf{Ext2}$ is an average-case $(v + \log(1/\alpha), \epsilon + 2\alpha)$-two-source extractor (for any $\alpha > 0$).

Hence, we have that $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Pre}}(m) \le \epsilon + 2\alpha$ with $L_{\mathsf{Pre}} \le$

$t - v - \log(1/\alpha)$ for any unbounded distinguisher $\mathcal{A}$.

Here we remark that for sufficiently large t, we can take enough small $\epsilon = 2^{-2m}$, due to Lemma 2.8. Suppose that $\mathsf{Ext2}' : (\{0,1\}^t)^2 \to \{0,1\}^{m'}$ is a worst-case $(v, 2^{-cm'})$-two-source extractor for some constant $c > 0$, and $\mathsf{Ext2} : (\{0,1\}^t)^2 \to \{0,1\}^m$ is the extractor such that it outputs the first $m = m'/d$ bit of the output of $\mathsf{Ext2}'$. By Lemma 2.8, the latter is $(v, 2^{-(cd)m})$-two-source extractor, with $2^{-cm'} = 2^{-(cd)m}$.

We set $\alpha = 2^{-2m}$. Then $\epsilon + 2\alpha \leq 2^{-3m/2}$, and $t - v - \log(1/\alpha) = t - (1/2 - \gamma')t - 2m = (1/2 + (\gamma' - 2m/t))t$. Hence, we can take another constant $0 < \gamma \leq \gamma' - 2m/t$ for sufficiently large t. By construction, we can indeed choose t, m (almost) independently, conditioned on $t > m$ [6].

Therefore, for sufficiently large t and any unbounded distinguisher $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{Pre}}_{\mathcal{A}}(m) \leq 2^{-3m/2}$ as long as $L_{\mathsf{Pre}} \leq (1/2+\gamma)t$, where $0 < \gamma < 1/2$ is (close to) a universal constant given by Bourgain's extractor. □

We now observe the following: If there is a two-source extractor resilient to the pre-challenge leakage such that $\mathsf{Adv}^{\mathsf{Pre}}_{\mathcal{A}'}(m) \leq 2^{-3m/2}$ for any adversary $\mathcal{A}'$, the two-source extractor is also resilient to the post-challenge leakage such that $\mathsf{Adv}^{\mathsf{Post}}_{\mathcal{A}}(m) \leq 2^{-m/2}$ for any adversary $\mathcal{A}$, because one can correctly guess the challenge value w at least with 2^{-m}.

Theorem 3.2: There exist a constant γ, with $0 < \gamma < 1/2$, $m = \Omega(t)$, and polynomial-time computable two-source extractor $\mathsf{Ext2} : \{0,1\}^t \times \{0,1\}^t \to \{0,1\}^m$, such that, for every sufficiently large t, for any unbounded distinguisher $\mathcal{A}$, and for any $L_{\mathsf{Pre}}, L_{\mathsf{Post}}$ with $L_{\mathsf{Pre}} + L_{\mathsf{Post}} \leq (1/2 + \gamma)t$, $\mathsf{Adv}^{\mathsf{Post}}_{\mathcal{A}}(m) \leq 2^{-m/2}$. In addition, constant γ is close to a universal constant given by Bourgain's extractor [6].

Proof. Let $\mathsf{Ext2}$ be the two-source extractor given in Theorem 3.1. Namely, for any distinguisher $\mathcal{A}'$ that attacks $\mathsf{Ext2}$ in the pre-challenge game, it holds that $\mathsf{Adv}^{\mathsf{Pre}}_{\mathcal{A}'}(m) \leq 2^{-3m/2}$ as long as $L_{\mathsf{Pre}} \leq (1/2 + \gamma)t$.

Suppose for contradiction that there is a distinguisher $\mathcal{A}$ that attacks $\mathsf{Ext2}$ in the post-challenge leakage game with $\mathsf{Adv}^{\mathsf{Post}}_{\mathcal{A}}(m) > 2^{-m/2}$ and $L_{\mathsf{Pre}} + L_{\mathsf{Post}} \leq (1/2 + \gamma)t$. Then, we can construct a distinguisher $\mathcal{A}'$ in the pre-challenge leakage game as follows:

1. $\mathcal{A}'$ chooses $w \overset{\mathsf{U}}{\leftarrow} \{0,1\}^m$.
2. $\mathcal{A}'$ runs $\mathcal{A}$ in the *post-challenge* leakage game with the challenge w and waits until $\mathcal{A}$ outputs a bit b'.
3. After the pre-challenge stage (for $\mathcal{A}'$), $\mathcal{A}'$ receives w_b from the challenger.
4. If $w = w_b$, then $\mathcal{A}'$ outputs b'. Otherwise, it outputs a random bit.

It is easy to see that $\mathsf{Adv}^{\mathsf{Pre}}_{\mathcal{A}'}(m) = 2^{-m}\mathsf{Adv}^{\mathsf{Post}}_{\mathcal{A}}(m)$, because the probability $w = w_b$ is 2^{-m}. Therefore, $\mathsf{Adv}^{\mathsf{Pre}}_{\mathcal{A}'}(m) > 2^{-3m/2}$. This contradicts Theorem 3.1. □

We note that Halevi and Lin [17] implicitly utilize the fact that a *good* two-source extractor is a post-challenge leakage resilient two-source extractor. What we have done here is to explicitly define a post-challenge leakage resilient two-source extractor and show the fact that Halevi and Lin [17] has already shown implicitly.

4. Our Scheme

Let $\mathsf{PKE}^{\mathsf{CCA2}} = (\mathsf{Gen}^{\mathsf{CCA2}}, \mathsf{Enc}^{\mathsf{CCA2}}, \mathsf{Dec}^{\mathsf{CCA2}})$ be an arbitrary IND-CCA2 secure PKE. Let $\mathsf{Ext2} : \{0,1\}^t \times \{0,1\}^t \to \{0,1\}^m$ be an arbitrary two-source extractor shown in Theorem 3.2. For simplicity, we assume that the randomness spaces of $\mathsf{Gen}^\star$ and $\mathsf{Enc}^\star$ are both $\{0,1\}^m$. We then provide a new PKE scheme $\mathsf{PKE}^\star = (\mathsf{Gen}^\star, \mathsf{Enc}^\star, \mathsf{Dec}^\star)$ as follows.

- $\mathsf{Gen}^\star$, the key generation algorithm, is a PPT algorithm that takes 1^λ as input and outputs $(PK, SK) = (pk, (s_1, s_2))$, where
  1. $s_1, s_2 \overset{\mathsf{U}}{\leftarrow} \{0,1\}^t$;
  2. $w = \mathsf{Ext2}(s_1, s_2)$;
  3. $(pk, sk) = \mathsf{Gen}^{\mathsf{CCA2}}(1^\lambda; w)$.
- $\mathsf{Enc}^\star$, the encryption algorithm, is a PPT algorithm that takes (PK, M) and outputs $c = \mathsf{Enc}^{\mathsf{CCA2}}(pk, M; w')$, where
  1. $r_1, r_2 \overset{\mathsf{U}}{\leftarrow} \{0,1\}^{t'}$;
  2. $w' = \mathsf{Ext2}(r_1, r_2)$.
- $\mathsf{Dec}^\star$, the decryption algorithm, is a DPT algorithm that takes (SK, c), where $SK = (s_1, s_2)$, and outputs $M = \mathsf{Dec}^{\mathsf{CCA2}}(sk, c)$, where
  1. $w = \mathsf{Ext2}(s_1, s_2)$;
  2. $(pk, sk) = \mathsf{Gen}^{\mathsf{CCA2}}(1^\lambda; w)$.

We now show the security of our scheme in the following theorem:

Theorem 4.1: Let $0 < \gamma < 1/2$ be a constant close to a universal constant γ' determined by the two-source extractor of [6]. The above scheme $\mathsf{PKE}^\star$ is (L_s, L_r)-KR-postLR-CCA2 secure in the 2-split-state model, as long as $L_s \leq (1/2 + \gamma)t$ and $L_r \leq (1/2 + \gamma)t'$.

Proof. The basic idea of the security proof is as follows: Let $\mathsf{Ext2}$ be a post-challenge leakage resilient two-source extractor, as shown in Theorem 3.2. In the security proof, we first replace $w = \mathsf{Ext2}(s_1, s_2)$ with random string $\tilde{w}$, which is independent of s_1, s_2. We then show that the replacement yields no significant gaps on the adversary's advantage; Otherwise, it could give a good distinguisher that can break post-challenge leakage resilient two-source extractor $\mathsf{Ext2}$ in game G_{Ext2}, which contradicts Theorem 3.2.

We also apply a similar argument to the sender-randomness leakage. Let $w'^{(k)} = \mathsf{Ext2}(r_1^{(k)}, r_2^{(k)})$ be the random string used to encrypt the k-th challenge message for $k \in [\ell]$, where ℓ is the number of the challenge messages given by $\mathcal{A}$. We then repeat replacing $w'^{(k)}$ with truly random $\tilde{w}'^{(k)}$, one by one, from $k = 1$ to ℓ. Finally, all strings of sender-randomness used in Enc are replaced with truly random strings, $\tilde{w}'^{(k)}$, which are independent of $r^{(k)} = (r_1^{(k)}, r_2^{(k)})$.

Note that, after the replacements, any information from the leakage oracles is independent of the secret key and sender-randomnesses that the challenger indeed utilizes. Therefore, the game eventually coincides with the multiple-message IND-CCA2 game, and hence, the adversary has only negligible advantage since the multiple-message IND-CCA2 security implies the standard (single-message) IND-CCA2 security.

We now turn to the formal security proof. Consider the following game sequences from G_0 to G_3.

Game G_0: This game is the original KR-postLR CCA2 game in the 2-split-state model.

Game G_1: This game is the same as Game G_0 except that we instead use a truly random string $\tilde{w}$, which is independent of secret-key (s_1, s_2) generated by $\mathsf{Gen}^\star$.

Game G_2: This game is the same as Game G_1 except that for every $k \in [\ell]$, we apply a truly random string $\tilde{w}'^{(k)}$ to produce k-th challenge ciphertext, which is independent of $(r_1^{(k)}, r_2^{(k)})$ generated by $\mathsf{Enc}^\star$.

Game G_3: In this game, we construct adversary $\mathcal{A}^*$ that attacks $\mathsf{PKE}^{\mathsf{CCA2}}$ in the IND-CCA2 game as follows: $\mathcal{A}^*$ starts with the target public-key pk. It picks up at random (s_1, s_2) and $(r_1^{(k)}, r_2^{(k)})$ for all $k \in [\ell]$, which are completely independent from $\tilde{w}$ behind pk and randomness $\tilde{w}'^{(j)}$'s behind the sequence of the challenge ciphertexts, given by the challenger. $\mathcal{A}^*$ then runs $\mathcal{A}$ by simulating the leakage oracles. When $\mathcal{A}$ finally outputs b', $\mathcal{A}^*$ outputs it.

Our goal is to bound $\mathsf{Adv}_{\mathcal{A}}^{G_0}$ by estimating gaps between every two games and using the assumption $\mathsf{Adv}_{\mathcal{A}^*}^{G_3} \leq \epsilon_{\mathsf{CCA2}}$ for some negligible ϵ_{CCA2}, where we denote by $\mathsf{Adv}_{\mathcal{A}}^{G}$ the advantage of adversary $\mathcal{A}$ in a game G. We show that a gap between every $\mathsf{Adv}_{\mathcal{A}}^{G_{i-1}}$ and $\mathsf{Adv}_{\mathcal{A}}^{G_i}$ is quite small.

Claim 4.2: $|\mathsf{Adv}_{\mathcal{A}}^{G_0} - \mathsf{Adv}_{\mathcal{A}}^{G_1}| = 2^{-\Omega(m)}$.

Proof. We construct a distinguisher $\mathcal{D}$ that attacks two-source extractor $\mathsf{Ext2}$ in the post-challenge leakage game G_{Ext2} defined in Section 3 as follows.

1. The challenger picks up $s_1, s_2 \overset{\mathsf{U}}{\leftarrow} \{0,1\}^t$ and sets w^*, which is either $\mathsf{Ext2}(s_1, s_2)$ or a random string. The challenger then feeds w^* to distinguisher $\mathcal{D}$.
2. $\mathcal{D}$ sets $(pk, sk) = \mathsf{Gen}^{\mathsf{CCA2}}(1^\lambda; w^*)$ and runs $\mathcal{A}$ with pk. If $\mathcal{A}$ submits query (i, f_s) (where $i \in \{1,2\}$) to the secret-key leakage oracle, $\mathcal{D}$ submits it to the challenger as a query in the post-challenge stage, and return $f_s(s_i)$ (responded by the challenger) to $\mathcal{A}$. If $\mathcal{A}$ submits c for decryption, $\mathcal{D}$ responds with $M = \mathsf{Dec}^{\mathsf{CCA2}}(sk, c)$ using sk.
3. When $\mathcal{A}$ outputs ℓ, $\mathcal{D}$ chooses 2ℓ random strings, $r_i^{(k)} \overset{\mathsf{U}}{\leftarrow} \{0,1\}^{t'}$, for $i \in \{1,2\}$ and $k \in [\ell]$, and sets $w'^{(k)} := \mathsf{Ext2}(r_1^{(k)}, r_2^{(k)})$ for every $k \in [\ell]$. If $\mathcal{A}$ submits (i, k, f_r), where $i \in \{1,2\}$ and $k \in [\ell]$, to the sender-randomness leakage oracle, $\mathcal{D}$ returns $f_r(r_i^{(k)})$ to $\mathcal{A}$. If $\mathcal{A}$ submits c for decryption, $\mathcal{D}$ responds with $M = \mathsf{Dec}^{\mathsf{CCA2}}(sk, c)$ using sk.
4. When $\mathcal{A}$ submits $(\boldsymbol{M_0}, \boldsymbol{M_1})$, $\mathcal{D}$ chooses $b \overset{\mathsf{U}}{\leftarrow} \{0,1\}$ and sets $\boldsymbol{c}^* := \mathsf{Enc}^{\mathsf{CCA2}}(pk, \boldsymbol{M_b}; \boldsymbol{w'})$. $\mathcal{D}$ feeds $\boldsymbol{c}^*$ to $\mathcal{A}$.
5. When $\mathcal{A}$ submits a query, $\mathcal{D}$ replies to it in the same manner as in Steps, 4 and 5. We note that $\mathcal{A}$ is not allowed to query any of the challenge ciphertexts for decrytion.
6. Finally, when $\mathcal{A}$ outputs b', $\mathcal{D}$ outputs 1 if $b = b'$ and a random bit otherwise.

We note that Game G_0 corresponds to the case $w^* = \mathsf{Ext2}(s_1, s_2)$, while Game G_1 corresponds to the case that w^* is a truly random string. By construction, we have $|\mathsf{Adv}_{\mathcal{A}}^{G_0} - \mathsf{Adv}_{\mathcal{A}}^{G_1}| \leq 2\mathsf{Adv}_{\mathcal{D}}^{\mathsf{Post}}$. Therefore, by Theorem 3.2, we have $|\mathsf{Adv}_{\mathcal{A}}^{G_0} - \mathsf{Adv}_{\mathcal{A}}^{G_1}| \leq 2^{-m/2+1}$. □

We now consider the difference of the advantages of $\mathcal{A}$ between Games, G_1 and G_2.

Claim 4.3: $|\mathsf{Adv}_{\mathcal{A}}^{G_1} - \mathsf{Adv}_{\mathcal{A}}^{G_2}| = \ell \cdot 2^{-\Omega(m)}$.

Proof. The proof strategy is substantially the same as in the claim above, where we replace $\mathsf{Ext2}(r_1^{(k)}, r_2^{(k)})$ with random string $\tilde{w}'^{(k)}$ from $k = 1$ to ℓ one by one. So, we consider a sequence of games $G_1^{(0)}, ..., G_1^{(\ell)}$, where $G_1^{(0)} = G_1$ and $G_1^{(\ell)} = G_2$. In Game $G_1^{(k)}$, with $1 \leq k \leq \ell$, we use uniform random strings $\tilde{w}'^{(j)}$ for $j \leq k$ while the outputs of extractor $w'^{(j)} = \mathsf{Ext2}(r_1^{(j)}, r_2^{(j)})$ from random $r_1^{(j)}, r_2^{(j)}$ for $k < j \leq \ell$.

We discuss the difference of $\mathcal{A}$'s advantages between $G_1^{(k-1)}$ and $G_1^{(k)}$ for each k. For this purpose, we construct distinguisher $\mathcal{D}$ that attacks two-source extractor $\mathsf{Ext2}$ in the post-challenge leakage game G_{Ext2} defined in Section 3 as follows:

1. The challenger picks up $r_1^{(k)}, r_2^{(k)} \overset{\mathsf{U}}{\leftarrow} \{0,1\}^{t'}$ and sets $w'^{(k)}$, which is either $\mathsf{Ext2}(r_1^{(k)}, r_2^{(k)})$ or a random string. The challenger feeds $w'^{(k)}$ to $\mathcal{D}$. Later, $\mathcal{D}$ uses $w'^{(k)}$ to produce k-th ciphertext.
2. $\mathcal{D}$ picks up random $s_1, s_2 \in \{0,1\}^t$ and $\tilde{w}$, independently. $\mathcal{D}$ sets $(pk, sk) := \mathsf{Gen}^{\mathsf{CCA2}}(1^\lambda; \tilde{w})$ and runs $\mathcal{A}$ with pk. If $\mathcal{A}$ submits query (i, f_s) (where $i \in \{1,2\}$) to the secret-key leakage oracle, $\mathcal{D}$ returns $f_s(s_i)$ to $\mathcal{A}$. If $\mathcal{A}$ submits ciphertext c for decryption, $\mathcal{D}$ responds with $M = \mathsf{Dec}^{\mathsf{CCA2}}(sk, c)$ using sk.
3. When $\mathcal{A}$ outputs ℓ, $\mathcal{D}$ first picks up $(k-1)$ random strings, $\tilde{w}'^{(1)}, \ldots, \tilde{w}'^{(k-1)}$. Then, $\mathcal{D}$ picks up $2(\ell - 1)$ random strings, $r_i^{(j)} \overset{\mathsf{U}}{\leftarrow} \{0,1\}^{t'}$, for $i \in \{1,2\}$ and $j \in [\ell]\backslash\{k\}$, and sets $w'^{(j)} = \mathsf{Ext2}(r_1^{(j)}, r_2^{(j)})$ for every $j > k$. $\mathcal{D}$ sets $\boldsymbol{w'} := (\tilde{w}'^{(1)}, \ldots, \tilde{w}'^{(k-1)}, w'^{(k)}, w'^{(k+1)}, \ldots, w'^{(\ell)})$. If $\mathcal{A}$ submits (i, j, f_r), where $i \in \{1,2\}$ and $j \in [\ell]$, to the sender-randomness leakage oracle, $\mathcal{D}$ returns $f_r(r_i^{(j)})$ to $\mathcal{A}$ when $j \neq k$. If $j = k$, $\mathcal{D}$ submits the query to the challenger as in the post-challenge stage and passes the challenger's response to $\mathcal{A}$. If $\mathcal{A}$ submits ciphertext c for decryption, $\mathcal{D}$ responds with $M = \mathsf{Dec}^{\mathsf{CCA2}}(sk, c)$ using sk.
4. When $\mathcal{A}$ submits $(\boldsymbol{M_0}, \boldsymbol{M_1})$, with $\boldsymbol{M_i} = (M_i^{(1)}, \ldots,$

$M_i^{(\ell)}$), $\mathcal{D}$ chooses $b \overset{\mathsf{U}}{\leftarrow} \{0,1\}$ and sets $\boldsymbol{c}^* := \mathsf{Enc}^{\mathsf{CCA2}}(pk, \boldsymbol{M_b}; \boldsymbol{w'})$, and sends $\boldsymbol{c}^*$ to $\mathcal{A}$. We note that $\boldsymbol{w'} := (\tilde{w}'^{(1)}, \dots, \tilde{w}'^{(k-1)}, w'^{(k)}, w'^{(k+1)}, \dots, w'^{(\ell)})$, where the first $(k-1)$ strings are truly random, the sequence after the k-th string is of pseudo random strings, and the k-th string is the one given by the challenger.

5. When $\mathcal{A}$ submits a query, $\mathcal{D}$ replies to it in the same manner as in Steps, 4 and 5. We note that $\mathcal{A}$ is not allowed to query any of the challenge ciphertexts for decrytion.
6. Finally, when $\mathcal{A}$ outputs b', $\mathcal{D}$ outputs 1 if $b' = b$ and a random bit otherwise.

We note that Game $G_1^{(k-1)}$ corresponds to the case that $w'^{(k)}$ is a truly random string, while Game $G_1^{(k)}$ corresponds to the case $w'^{(k)} = \mathsf{Ext2}(r_1^{(k)}, r_2^{(k)})$. By construction, $|\mathsf{Adv}_{\mathcal{A}}^{G_1^{(k-1)}} - \mathsf{Adv}_{\mathcal{A}}^{G_1^{(k)}}| \le 2\mathsf{Adv}_{\mathcal{D}}^{\mathsf{Post}}$. Hence, by Theorem 3.2, $|\mathsf{Adv}_{\mathcal{A}}^{G_1^{(k-1)}} - \mathsf{Adv}_{\mathcal{A}}^{G_1^{(k)}}| \le 2^{-m/2+1}$. Repeating this argument for $1 \le k \le \ell$, we obtain that $|\mathsf{Adv}_{\mathcal{A}}^{G_1} - \mathsf{Adv}_{\mathcal{A}}^{G_2}| \le \ell \cdot 2^{-m/2+1}$. □

Claim 4.4: $\mathsf{Adv}_{\mathcal{A}}^{G_2} = \mathsf{Adv}_{\mathcal{A}^*}^{G_3}$ and $\mathsf{Adv}_{\mathcal{A}^*}^{G_3} = \mathsf{negl}(\lambda)$.

Proof. The leaked information from the leakage oracle is independent of the true decryption key and random sources used in $\mathsf{Enc}^{\mathsf{CCA2}}$. Therefore, $\mathsf{Adv}_{\mathcal{A}}^{G_2} = \mathsf{Adv}_{\mathcal{A}^*}^{G_3}$. $\mathsf{Adv}_{\mathcal{A}^*}^{G_3} = \mathsf{negl}(\lambda)$, because one-message IND-CCA2 security implies multiple-message IND-CCA2 security. □

Combining these claims, we obtain the theorem statement. □

By this theorem, it is easy to see that our encryption scheme is resilient to leakage of $(1/2+\gamma)\ell_{sk}/2$ and $(1/2+\gamma)\ell_{rd}/2$ for some constant $0 < \gamma < 1/2$ where $\ell_{sk} = 2t$ is the total length of the secret key and $\ell_{rd} = 2t'$ is the total length of the random string to create one ciphertext.

4.1 In the Multiple-Split State Model

Halevi and Lin have claimed that their scheme is extended in the multiple-split state model, using another extractor [2] and the leakage of each split state turns out $(1-o(1))$. (We note however that the total leakage amount is $1/C(1-o(1))$ in the C-split state model.) We remark that our scheme is also tailored in the multiple-split state model, using the extractor [2]. An C-source (k,ϵ)-extractor [2] takes independent C random variables, $X_1, \dots, X_C \in \{0,1\}^n$ with $H_\infty(X_i) \ge k$ for every $i \in [C]$, and outputs a m-bit string which is ϵ-close to m-bit uniform distribution.

Theorem 4.5 (Corollary4.2 in [2]): There are constants c_1, c_2 and a C-source (k,ϵ)-extractor $\mathsf{Ext} : (\{0,1\}^n)^C \to \{0,1\}^m$ for every n, k with $k > \log^{10}(n)$, $C = O(\frac{\log n}{\log k})$ and $m = c_2 k$. Namely, if $X_1, \dots, X_C$ are independent random variables with with $H_\infty(X_i) \ge k$ for every $i \in [C]$, it holds that $\Delta(\mathsf{Ext}(X_1, \dots, X_C), U_m) < 2^{-k^{c_1}}$.

Here, we set $k = n^{0.9}$. Then, $m = O(n^{0.9})$ and the allowable leakage amount of each source turns out $n(1-o(1))$ because of $n(1-n^{-0.1})$. We then construct a C-source extractor resilient to post-challenge leakage as in Sect. 3. We note that a multiple-source version of Lemma 2.9 simply holds and we can take enough small statistical distance ϵ due to Lemma 2.8. We then apply our C-source post-challenge resilient extractor to an arbitrary PKE as in Sect. 4. We note that we can use a pseudo random generator to extend an $O(n^{0.9})$-length random string to an $O(n)$-length pseudo random string (where $n = O(poly(n^{0.9}))$) and apply it to a PKE scheme with security parameter n, although the security level is lowered to $n^{0.9}$.

5. Comparison with Halevi-Lin Scheme [17]

We review the Halevi-Lin scheme [17] to clarify its properties and compare with our scheme.

5.1 Hash Proof Systems

We recall the hash proof systems [8], [9], by borrowing the notation of [18], [24]. Let $\mathcal{C}, \mathcal{K}, SK$, and PK be efficiently samplable sets and let $\mathcal{V}$ be a subset in $\mathcal{C}$. Let $\Lambda_{sk} : \mathcal{C} \to \mathcal{K}$ be a hash function indexed by $sk \in SK$. A hash function family $\Lambda : SK \times \mathcal{C} \to \mathcal{K}$ is projective if there is a projection $\mu : SK \to PK$ such that $\mu(sk) \in PK$ defines the action of Λ_{sk} over subset $\mathcal{V}$. That is to say, for every $C \in \mathcal{V}$, $K = \Lambda_{sk}(C)$ is uniquely determined by $\mu(sk)$ and C. Λ is called ϵ-almost 1-universal if for all $C \in \mathcal{C} \backslash \mathcal{V}$,

$$\Delta((pk, \Lambda_{sk}(C)), (pk, K)) \le \epsilon,$$

where $sk \overset{\mathsf{U}}{\leftarrow} SK$ and $K \overset{\mathsf{U}}{\leftarrow} \mathcal{K}$ and $pk = \mu(sk)$. A hash proof system $\mathsf{HPS} = (\mathsf{HPS.param}, \mathsf{HPS.pub}, \mathsf{HPS.priv})$ consists of three algorithms such that $\mathsf{HPS.param}$ takes 1^λ and outputs an instance of $\mathsf{params} = (\mathsf{group}, \Lambda, \mathcal{C}, \mathcal{V}, SK, PK, \mu)$, where group contains some additional structural parameters and Λ is a projective hash function family associated with $(\mathcal{C}, \mathcal{V}, SK, PK, \mu)$ as defined above. The deterministic public evaluation algorithm $\mathsf{HPS.pub}$ takes as input $pk = \mu(sk)$, $C \in \mathcal{V}$ and a witness w such that $C \in \mathcal{V}$ and returns $\Lambda_{sk}(C)$. The deterministic private evaluation algorithm takes $sk \in SK$ and returns $\Lambda_{sk}(C)$, without taking withness w for C (if it exists). A hash proof system HPS as above is said to have a hard subset membership problem if two random elements $C \in \mathcal{C}$ and $C' \in \mathcal{C} \backslash \mathcal{V}$ are computationally indistinguishable, that is, $\{C \mid C \overset{\mathsf{U}}{\leftarrow} \mathcal{C}\}_{\lambda \in \mathbb{N}} \overset{\mathsf{c}}{\approx} \{C' \mid C' \overset{\mathsf{U}}{\leftarrow} \mathcal{C} \backslash \mathcal{V}\}_{\lambda \in \mathbb{N}}$.

5.2 Halevi-Lin Scheme

In the Halevi-Lin scheme, the key generation is done by picking up $sk_1, sk_2 \overset{\mathsf{U}}{\leftarrow} SK$ independently and setting $pk_1 = \mu(sk_1)$ and $pk_2 = \mu(sk_2)$. To encrypt message m, the sender picks random x_1, x_2, computes $c_i = x_i \oplus \mathsf{HPS.pub}(pk_i, C_i, w_i)$ for $i \in \{1,2\}$ and computes $c = M \oplus \mathsf{Ext2}(x_1, x_2)$. The sender then sends ciphertext (C_1, C_2, c_1, c_2, c) to the receiver. To decrypt them, the decryptor computes $x'_i = c_i \oplus \mathsf{HPS.priv}(sk_i, C_i)$ for $i \in \{1,2\}$ and obtains $c \oplus \mathsf{Ext2}(x'_1, x'_2)$. Here, for each i, x_i and (C_i, c_i) are a message/ciphertext pair of a hash proof system based PKE scheme. This PKE is replaced with any entropic leakage-resilient CPA PKE scheme in the Halevi-Lin scheme, although the only known construction so far is the hash proof system based.

(5) Leakage Rate.

It follows from HPS [24] that for every $C_i \in \mathcal{C}\backslash\mathcal{V}$, letting $x_i = \mathsf{HPS.priv}(sk_i, C_i)$, $\widetilde{H}_\infty(x_i|(pk_i, L)) \leq \widetilde{H}_\infty(sk_i|(pk_i, L))$, where L is a leakage variable. By construction (of the two-source extractor), we require $\widetilde{H}_\infty(x_i|(pk_i, L)) \geq (1/2-\gamma)|\mathcal{K}|$, which implies that the leakage amount of sk_i is at most $(1/2+\gamma)|SK|$. Hence, the leakage amount of secret keys of [17] is at most $(1/2+\gamma)\ell_{sk}/2$, where ℓ_{sk} denotes the total length of secret key and γ is a universal constant of post-challenge leakage-resilient two-source extractors. This amount is essentially equivalent to that of our proposed scheme.

(6) Single-Message Security.

Halevi-Lin scheme is not proven CPA secure against multiple challenge ciphertexts. Let $\boldsymbol{c} = \{(C_1^{(i)}, C_2^{(i)}, c_i)\}$ be the challenge ciphertexts, where $c_i = M_b^{(i)} \oplus \mathsf{Ext2}(x_1^{(i)}, x_2^{(i)})$. Here, sk_j ($j \in \{0,1\}$) is related to all $x_j^{(i)} = \mathsf{HPS.priv}(sk_j, C_j^{(i)})$ where $i \in \{1,\dots,\ell\}$. By applying the trick in Theorem 3.2 (originally appeared in Claim 8 in [17]), $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Post}}(\lambda) = 2^{\ell m}\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Pre}}(\lambda)$, where $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Pre}}$ is the advantage of $\mathcal{A}$ in the presence of pre-challenge leakage and $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Post}}$ is the advantage of $\mathcal{A}$ in the presence of post-challenge leakage. Here, since ℓ is a-priori unbounded polynomial, $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{Post}}(\lambda)$ turns out non-negligible for some ℓ. Therefore, the Halevi-Lin scheme [17] is not proven CPA secure against multiple challenges. We note that we do not say that the Halevi-Lin scheme [17] is insecure against multiple challenges. We only say that the current proof strategy in [17] is not applicable to multiple challenges.

(7) In the Multiple-Split State Model

As already mentioned, Halevi-Lin scheme can be extended in the multiple-split state model. When using C-source extractor of [2], it has $\frac{\ell_{sk}}{C}(1-o(1))$ of total leakage amount of secret key, similarly as ours, where $k = n^{1/d}$ for any $d > 1$.

6. Concluding Remarks

We have provided a very simple way to construct a (multiple-challenge) CCA2 secure PKE scheme that is simultaneously resilient to post-challenge secret-key and sender-randomness leakage from any CCA2 secure PKE scheme in the two-split state model, which was stated open in [17]. Our scheme has the same leakage rate as that of [17] and secure against multiple challenges unlike [17]. By construction, it is obvious that our method can be applied to other cryptographic primitives. For instance, when applying the two-source extractor to randomness behind a master-key and sender-randomness behind an encryption of an identity based encryption scheme (IBE), we can obtain a post-challenge master-key and sender-randomness leakage resilient IBE scheme from any IBE scheme.

We conclude our paper with the following remarks.

- *Did It Really Solve Open Issues (about CCA instantiations and randomness leakage)?:* We think the answer is YES. Although the model of the decryption algorithm is syntactically different from the original definition, it can be fit, as shown in Remark 2.10.
- *Is The Current Split-State Model Adequate?:* As discussed above, it is impossible to achieve post-challenge secret-key or sender-randomness leakage for PKE in the plain model. Therefore, searching a weaker appropriate model is a right direction. The split-state model by Halevi and Lin is one attempt of going toward this goal. We have found a simple solution in this model. Some may think that our solution is artificial. Although a half of our motivation for this work is to solve open issues, the remaining half is to raise a question of whether the current split-state model is reasonable. We hope our solution inspires the reader to discuss an appropriate model.

Acknowledgements

We would like to thank anonymous reviewers for useful comments.

References

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan, "Simultaneous hardcore bits and cryptography against memory attacks," TCC, Lect. Notes Comput. Sci., vol.5444, pp.474–495, Springer, 2009.

[2] B. Barak, A. Rao, R. Shaltiel, and A. Wigderson, "2-source dispersers for $n^{o(1)}$ entropy and ramsey graphs beating the frankl-wilson construction," Annals of Mathematics, vol.176, no.3, pp.1483–1543, 2012.

[3] M. Bellare, Z. Brakerski, M. Naor, T. Ristenpart, G. Segev, H. Shacham, and S. Yilek, "Hedged public-key encryption: How to protect against bad randomness," ASIACRYPT, Lect. Notes Comput. Sci., vol.5912, pp.232–249, Springer, 2009.

[4] M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, "Relations among notions of security for public-key encryption schemes," CRYPTO'98, Lect. Notes Comput. Sci., vol.1462, pp.26–45, 1998.

[5] E. Birrell, K.-M. Chung, R. Pass, and S. Telang, "Randomness-dependent message security," TCC'13, pp.700–720, 2013.

[6] J. Bourgain, "More on the sum-product phenomenon in prime fields and its applications," International Journal of Number Theory, vol.1, pp.1–32, 2005.

[7] Z. Brakerski, Y.T. Kalai, J. Katz, and V. Vaikuntanathan, "Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage," FOCS, pp.501–510, IEEE Computer Society, 2010. Full version available from http://eprint.iacr.org/2010/278

[8] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure aganist adaptive chosen ciphertext attacks," CRYPTO'98, Lect. Notes Comput. Sci., vol.1462, pp.13–25, 1998.

[9] R. Cramer and V. Shoup, "Universal hash proofs and a paradigm

for adaptive chosen ciphertext secure public-key encryption," Eurocrypt'02, Lect. Notes Comput. Sci., vol.2332, pp.45–64, 2002.

[10] Y. Dodis, A.B. Lewko, B. Waters, and D. Wichs, "Storing secrets on continually leaky devices," in Rafail Ostrovsky, editor, FOCS, pp.688–697, IEEE, 2011.

[11] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data," SIAM J. Computing, vol.38, no.1, pp.97–139, 2008.

[12] S. Dziembowski and K. Pietrzak, "Intrusion-resilient secret sharing," FOCS, pp.227–237, IEEE Computer Society, 2007.

[13] S. Dziembowski and K. Pietrzak, "Leakage-resilient cryptography," FOCS, pp.293–302, IEEE Computer Society, 2008.

[14] O. Goldreich, "Foundations of Cryptography: Basic Applications," vol.2, Cambridge Press, 2004.

[15] S. Goldwasser and G.N. Rothblum, "Securing computation against continuous leakage," CRYPTO, Lect. Notes Comput. Sci., vol.6223, pp.59–79, Springer, 2010.

[16] J.A. Halderman, S.D. Schoen, N. Heninger, W. Clarkson, W. Paul, J.A. Calandrino, A.J. Feldman, J. Appelbaum, and E.W. Felten, "Lest we remember: Cold-boot attacks on encryption keys," Commun. ACM, vol.52, no.5, pp.91–98, 2009.

[17] S. Halevi and H. Lin, "After-the-fact leakage in public-key encryption," TCC, Lect. Notes Comput. Sci., vol.6597, pp.107–124, Springer, 2011.

[18] D. Hofheinz and E. Kiltz, "Secure hybrid encryption from weakened key encapsulation," CRYPTO, pp.553–571, 2007.

[19] Y. Ishai, A. Sahai, and D. Wagner, "Private circuits: Securing hardware against probing attacks," CRYPTO, Lect. Notes Comput. Sci., vol.2729, pp.463–481, Springer, 2003.

[20] A. Juma and Y. Vahlis, "Protecting cryptographic keys against continual leakage," CRYPTO, Lect. Notes Comput. Sci., vol.6223, pp.41–58, Springer, 2010.

[21] F.-H. Liu and A. Lysyanskaya, "Tamper and leakage resilience in the split-state model," Advances in Cryptology — Crypto 2012, Lect. Notes Comput. Sci., vol.7417, pp.517–532. Springer, 2012.

[22] S. Micali and L. Reyzin, "Physically observable cryptography (extended abstract)," TCC, Lect. Notes Comput. Sci., vol.2951, pp.278–296, Springer, 2004.

[23] H. Namiki, K. Tanaka, and K. Yasunaga, "Randomness leakage in the KEM/DEM framework," ProvSec, Lect. Notes Comput. Sci., vol.6980, pp.309–323, Springer, 2011.

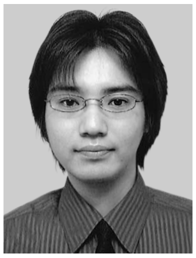
[24] M. Naor and G. Segev, "Public-key cryptosystems resilient to key leakage," CRYPTO, Lect. Notes Comput. Sci., vol.5677, pp.18–35, Springer, 2009.

[25] A. Rao, "An exposition of Bourgain's 2-source extractor," Electronic Colloquium on Computational Complexity (ECCC), TR07-034, 2007.

[26] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," J. ACM, vol.56, no.6, pp.84–93, 2009.

[27] Z. Zhang, S.S. M. Chow, and Z. Cao, "Post-challenge leakage in public-key encryption," SCIS 2013, 2A-4(2), 2013.

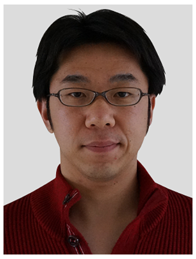
Eiichiro Fujisaki received the B.S. and Ph.D. degrees from Tokyo Institute of Technology in 1991 and 2005, respectively. He is currently engaged in research on cryptography and information security at NTT Secure Platform Laboratories. He is a member of IEICE, IPSJ, and IACR.



Akinori Kawachi is an assistant professor of Department of Mathematical and Computing Sciences, Tokyo Institute of Technology. He received B.E., M.Info., and Ph.D. degrees from Kyoto University in 2000, 2002, and 2004, respectively. His research interests are computational complexity, quantum computing, and foundations of cryptography.

Ryo Nishimaki received his B.E. and M.I. degrees from Kyoto University, Kyoto Japan in 2005 and 2007, and Dr.Sci degree from Tokyo Institute of Technology, Tokyo Japan in 2010. Currently he is engaged in research on public-key cryptography and cryptographic protocols at NTT Secure Platform Laboratories. He is a member of IEICE and IACR.

Keisuke Tanaka is Associate Professor of Department of Mathematical and Computing Sciences at Tokyo Institute of Technology. He received his B.S. from Yamanashi University in 1992 and his M.S. and Ph.D. from Japan Advanced Institute of Science and Technology in 1994 and 1997, respectively. For each degree, he majored in computer science. Before joining Tokyo Institute of Technology, he was Research Engineer at NTT Information Platform Labs.



Kenji Yasunaga received his B.E. degree in information and computer sciences in 2003, and his M.S. and Ph.D. degrees in information science and technology in 2005 and 2008, from Osaka University, Japan. He was a Postdoctoral Fellow at Kwansei Gakuin University in 2008, was an Assistant Professor at Tokyo Institute of Technology from 2008 to 2011, and was a Researcher at Institute of Systems, Information Technologies and Nanotechnologies (ISIT) from 2011 to 2012. He is currently an Assistant Professor at Kanazawa University. His research interests are in Coding Theory, Cryptography, and Computational Complexity.