IEICE TRANS. FUNDAMENTALS, VOL.E96–A, NO.6 JUNE 2013

PAPER Special Section on Discrete Mathematics and Its Applications

Leakage-Resilience of Stateless/Stateful Public-Key Encryption from Hash Proofs**

Manh Ha NGUYEN[†a)], *Nonmember*, **Kenji YASUNAGA**[†*b)], *and* **Keisuke TANAKA**[†c)], *Members*

SUMMARY We consider the problem of constructing public-key encryption (PKE) schemes that are resilient to a-posteriori chosen-ciphertext and key-leakage attacks (LR-CCA2). In CTYPTO'09, Naor and Segev proved that the Naor-Yung generic construction of PKE which is secure against chosen-ciphertext attack (CCA2) is also secure against key-leakage attacks. They also presented a variant of the Cramer-Shoup cryptosystem, and showed that this PKE scheme is LR-CCA2-secure under the decisional Diffie-Hellman assumption. In this paper, we apply the generic construction of "Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption" (EUROCRYPT'02) to generalize the above work of Naor-Segev. In comparing to the first construction of Naor-Segev, ours is more efficient because of not using simulation-sound NIZK. We also extend it to stateful PKE schemes. Concretely, we present the notion of LR-CCA2 attack in the case of stateful PKE, and a generic construction of stateful PKE that is secure against this attack.

key words: *public key encryption, hash proof system, key-leakage, chosen-ciphertext attack*

1. Introduction

Key-leakage attacks. Traditionally, the security of cryptographic schemes has been analyzed in an idealized setting, where anadversary only sees the specified input/output behavior of a scheme, but has no other access to its internal secret state. Unfortunately, in the real world, an adversary may often learn some partial information about secret state via various *key-leakage attacks* such as *side-channel attacks* or *cold-boot attack*. Schemes that are proven secure in an idealized setting, without key leakage, may become completely insecure if the adversary learns even a small amount of information about the secret key.

In 2009, Akavia, Goldwasser, and Vaikuntanathan [1] introduced the notion of *leakage resiliency* and the first leakage-resilient chosen-plaintext secure scheme under the LWE assumption. Then, Naor and Segev [12] extended this framework to the setting of chosen-ciphertext attacks (LR-CCA2). On the theoretical side, they proved that the Naor-Yung paradigm is applicable in this setting as well, and obtained as a corollary encryption schemes that are LR-CCA2-secure with the leakage rate of $1 - o(1)$ of the secret-key length. On the practical side, they proved that variants of the Cramer-Shoup cryptosystem are LR-CCA1-secure with the leakage rate of $1/4$, and LR-CCA2-secure with the leakage rate of $1/6$.

In 2010, Dodis et al. [9] proposed an efficient encryption scheme that is LR-CCA2-secure with the leakage rate of $1 - o(1)$ of the secret-key length. Their scheme relies on regular non-interactive zero-knowledge, which can be instantiated using the powerful Groth-Sahai techniques.

Stateful public-key encryption. In 2006, Bellare, Kohno, and Shoup [4] proposed the first model of stateful public-key encryption (StPE). The main goal of the StPE schemes is to reduce the cost of PKE by allowing a sender to maintain *state* that is reused across different encryptions. For example, one can obtain a stateful version of the ElGamal encryption in which a message M is encrypted to $(g^r, g^{rx}M)$ for public key g^x by maintaining the random value r and its corresponding value g^r as state so that g^r does not need to be computed each time.

Reducing the computational cost of PKE is of particular importance for low-power mobile devices where computational resources are constrained (such as PDA and mobile phones) or sensors communicating with the relatively powerful servers or base stations [10], [13]. Due to the efficiency gained from maintaining state, StPE schemes have potential to be employed in these settings. But, even in the environments that provide reasonable amount of computational resources, it is preferable to speed up public key operation.

In 2008, Baek, Zhou, and Bao [2] presented generic constructions of StPE, built several new StPE schemes and explained existing ones using their generic constructions. Some of them are built by using "identity-based technique" whereby one can construct PKE schemes secure against chosen-ciphertext attack in the standard model from identity-based encryption schemes.

Our contributions. In the paper [12], Naor and Segev proved that a variant of the Cramer-Shoup cryptosystem [7] is secure against LR-CCA2 attack. This LR-CCA2-secure scheme is based on the hardness of the decisional Diffie-Hellman problem. From this idea, we make the following contributions in this paper:

1. We present a generic construction of a stateless PKE

Manuscript received September 21, 2012.
Manuscript revised December 21, 2012.

[†]The authors are with the Department of Mathematical and Computing Sciences, Tokyo Institute of Techonology, Tokyo, 152-8552 Japan.
[*]Presently, with Institute of Systems, Information Technologies and Nanotechnologies (ISIT), Fukuoka-shi, 814-0001 Japan.

[**]A preliminary version of this paper was presented at the 17th Australasian Conference on Information Security and Privacy (ACISP 2012), LNCS, vol.7372, pp.208–222, 2012.

a) E-mail: nguyen9@is.titech.ac.jp
b) E-mail: yasunaga@isit.or.jp
c) E-mail: keisuke@is.titech.ac.jp

DOI: 10.1587/transfun.E96.A.1100

that is resilient to LR-CCA2 attack. This is the construction which generalizes Naor-Segev's result [12] using hash proof system (HPS). In this construction, we use the combination of any 1-universal HPS that satisfies the condition of a key-leakage extractor and any 2-universal HPS with some condition on the length of proof. −See Sect. 4.2.

2. We present the notion of LR-CCA2 attack in the case of StPE. Essentially, this notion is the same as that in the case of stateless PKE. We also present a generic construction of StPE that is secure against this attack. In this construction, we use the combination of two HPSs as in the case of stateless PKE and an IND-CCA-secure symmetric encryption scheme. This is also a new approach to achieve CCA2-secure StPE. −See Sect. 4.3.

These constructions do not rely on additional computational assumptions, and the resulting schemes are as efficient as the underlying HPS. Existing constructions of HPS (see, for example, [7], [11]) imply that our construction can be based on a variety of number-theoretic assumptions, including the decisional Diffie-Hellman assumption and its progressively weaker d-Linear variants, the quadratic residuosity assumption, and Paillier's composite residuosity assumption.

In comparing to the existing LR-CCA2-secure stateless PKE schemes [9], [12], ours is more efficient because of not using non-interactive zero-knowledge (NIZK). The scheme of Naor-Segev [12] is based on the Naor-Yung paradigm which makes use of the so-called simulation-sound NIZK (ss-NIZK). Informally, ss-NIZK proofs remain sound even if the attacker can see simulated proofs of arbitrary (even false) statements. It is well-known that this is a very inefficient primitive. Whereas, the scheme of Dodis et al. [9] uses Groth-Sahai's NIZKs which are only known for a pretty restrictive class or languages in bilinear groups. In spite of the recent advances in implementation technique, compared with standard operations such as modular exponentiation in finite fields, the bilinear map computation is still considered as a rather expensive operation.

Our construction of stateless PKE is LR-CCA2-secure with the leakage rate depending on the parameters of the underlying HPS. The Naor-Segev scheme [12] is an efficient instantiation which is LR-CCA2-secure with the leakage rate of 1/6 of the secret-key length. This rate is not as good as the result proposed in [9], which is LR-CCA2-secure with the leakage rate of $1 - o(1)$, but it is an important result for us to construct the generic construction of StPE that is resilient to LR-CCA2 attack. To the best of our knowledge, this is the first generic construction of StPE that is secure against this attack.

Road-map. We present notations, definitions, and tools in Sect. 2. The definitions of LR-CCA2 security in both cases of stateless and stateful PKE appear in Sect. 3. In Sects. 4 and 5, we describe our generic constructions and their concrete example, respectively. Finally, we conclude in Sect. 6.

2. Preliminaries

In this section we present notions, definitions, and tools that are used in our constructions. Let n be the security parameter of the schemes, U_t the uniform distribution of $\{0,1\}^t$ (where $t \in N$), and $\mathrm{U}(\mathbf{S})$ the uniform distribution of the set $\mathbf{S}$. We denote by $s \xleftarrow{\$} \mathbf{S}$ the assignment of a uniformly distributed random element from the set $\mathbf{S}$ to the variable s. We use $negl(n)$ to denote a negligible function in n.

The *statistical distance* between two random variables X and Y over a finite domain Ω is $\Delta(X,Y) = \frac{1}{2}\Sigma_{\omega\in\Omega}|\Pr[X = \omega] - \Pr[Y = \omega]|$. We also write $\Delta(x,y)$ instead of $\Delta(X,Y)$. We say that two variables are ϵ-*close* if their statistical distance is at most ϵ. The *min-entropy* of a random variable X is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X = x])$.

Dodis et al. [8] formalized the notion of *average min-entropy* that captures the remaining unpredictability of a random variable X conditioned on the value of a random variable Y, formally defined as follows:

$$\widetilde{\mathrm{H}}_\infty(X|Y) = -\log\left(E_{y\leftarrow Y}\left[2^{-\mathrm{H}_\infty(X|Y=y)}\right]\right).$$

The average min-entropy corresponds exactly to the optimal probability of guessing X, given knowledge of Y. The following bound on average min-entropy was proved in [8]:

Lemma 1 ([8]): If Y has 2^r possible values and Z is any random variable, then $\widetilde{\mathrm{H}}_\infty(X|Y,Z) \geq \mathrm{H}_\infty(X|Z) - r$.

One of the main tools in our constructions is a strong randomness extractor. The following definition naturally generalizes the standard definition of a strong extractor to the setting of average min-entropy:

Definition 1 ([8]): A function $\mathbf{Ext} : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ is an average-case (k,ϵ)-strong extractor if for all pairs of random variables (X,I) such that $X \in \{0,1\}^n$ and $\widetilde{\mathrm{H}}_\infty(X|I) \geq k$ it holds that

$$\Delta((\mathbf{Ext}(X,U_t),U_t,I),(U_m,U_t,I)) \leq \epsilon.$$

Hash proof systems. We review the framework of HPS, introduced by Cramer and Shoup [7], where HPS is considered as key-encapsulation mechanisms (using the notation of Naor and Segev [12]).

Let X, L, W be non-empty sets, such that L is a proper subset of X, and $R_L \subset X \times W$ be a binary relation. For $x \in X$ and $w \in W$ with $(x,w) \in R_L$, we say that w is a *witness* for x. Note that it would be quite natural to require that for all $x \in X$, we have $(x,w) \in R_L$ for some $w \in W$ if and only if $x \in L$, and that the relation R_L is efficiently computable. We can also view X as the set of all ciphertexts, L as the set of all valid ciphertexts (i.e., those generated appropriately with the corresponding witness). We denote by $(x,w) \xleftarrow{\$} \mathcal{R}_L$ the *instance sampling algorithm* of L, i.e. choose a random pair (x,w) such that $x \in X, w \in W$, and $(x,w) \in R_L$.

A hash proof system $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ consists of four algorithms that run in polynomial time. The

algorithm $Param(1^n)$ generates system parameter sp. We denote by $\mathcal{SK}_n$ and $\mathcal{PK}_n$ the sets of secret keys and public keys that are produced by $KGen(sp)$. That is, $KGen(sp) : \{0,1\}^* \to \mathcal{SK}_n \times \mathcal{PK}_n$ for every $n \in N$. The algorithm Pub receives as input a public key $pk \in \mathcal{PK}_n$, a valid ciphertext $x \in L$, and a witness w of the fact that $x \in L$, then outputs the encapsulated key $\pi \in \mathbf{\Pi}$, where $\mathbf{\Pi}$ denotes the set of encapsulated symmetric keys. The algorithm $Priv$ receives as input a secret key $sk \in \mathcal{SK}_n$ and a ciphertext $x \in X$, then outputs the encapsulated key π.

Consider the probability space defined by choosing sk randomly from the set of secret keys. We say that an HPS is *1-universal* for language L if for all $x \in X \setminus L$ and $\pi \in \mathbf{\Pi}$, it holds that

$$\Pr[Priv(sk, x) = \pi] = \frac{1}{|\mathbf{\Pi}|}.$$

We say that an HPS is *2-universal* for language L if for all $x, x^* \in X$ and $\pi, \pi^* \in \mathbf{\Pi}$, with $x \notin L \cup \{x^*\}$, it holds that

$$\Pr[Priv(sk, x) = \pi \mid Priv(sk, x^*) = \pi^*] = \frac{1}{|\mathbf{\Pi}|}.$$

It is easy to see that, if $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ is 1-universal, then

$$\Delta((pk, x, Priv(sk, x)), (pk, x, \mathrm{U}(\mathbf{\Pi}))) \leq negl(n),$$

and if **HPS** is 2-universal, then

$$\Delta((pk, x, x^*, \pi^*, Priv(sk, x)), (pk, x, x^*, \pi^*, \mathrm{U}(\mathbf{\Pi}))) \leq negl(n),$$

where $\pi^* = Priv(sk, x^*)$ and $\mathrm{U}(\mathbf{\Pi}) \in \mathbf{\Pi}$ is sampled uniformly at random.

We also need an extension of this notion. The definition of *extended* HPS is the same as that of ordinary HPS, except that the proof system HPS accepts an extra input from a finite set E. In this setting, the algorithm Pub takes as input $pk \in \mathcal{PK}_n$, $x \in L$, $e \in E$, and a witness w of the fact that $x \in L$, and the algorithm $Priv$ takes as input $sk \in \mathcal{SK}_n$, $x \in X$, and $e \in E$. We shall also require that elements of E are uniquely encoded as bit strings of length bounded by a polynomial in n, and that HPS provides an algorithm that efficiently determines whether a bit string is a valid encoding of an element of E.

We can modify in the obvious way to define *extended 1(2)-universal* HPS.

Next, we define a new property for HPS that is useful in our construction.

Definition 2: We say that a hash proof system $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ for a language L is a 1-universal (λ, ϵ)-key-leakage extractor if for any function $f : \{0,1\}^* \to \{0,1\}^\lambda$ we have

$$\Delta((pk, x, f(sk), Priv(sk, x)), (pk, x, f(sk), U(\mathbf{\Pi}))) \leq \epsilon,$$

where $x \in_R X \setminus L$. If $\epsilon = negl(n)$ we say that **HPS** is a 1-universal λ-key-leakage extractor for L.

Note that, we can obtain a 1-universal λ-key-leakage extractor HPS by combining any 1-universal HPS with any strong extractor (see Sect. 4.1).

Subset membership problem. Roughly, we require that L and $X \setminus L$ are computationally indistinguishable. The formal definition is as follows.

Definition 3: A subset membership problem is called *hard* if for any probabilistic polynomial-time adversary $\mathcal{A}$ it holds that

$$\mathrm{Adv}^{\mathrm{SM}}_{\mathrm{HPS},\mathcal{A}}(n) \stackrel{def}{=} \left|\Pr_{x_0 \leftarrow L}[\mathcal{A}(X, L, x_0) = 1] - \Pr_{x_1 \leftarrow X \setminus L}[\mathcal{A}(X, L, x_1) = 1]\right|.$$

is negligible in n.

3. Models

3.1 Leakage-Resilient CCA2 Stateless PKE

In this section we review the notion of a-posteriori chosen-ciphertext key-leakage attack, introduced by Naor and Segev [12].

Let $\Pi = (KGen, Enc, Dec)$ be a PKE scheme. The leakage oracle, denoted by $\mathrm{Leak}(sk)$, takes as input a function f and outputs $f(sk)$. We say that $\mathcal{A}$ is a λ-key-leakage adversary if the sum of output lengths of all the functions that $\mathcal{A}$ submits to the leakage oracle is at most λ.

In this game, the adversary is allowed to adaptively access a decryption oracle $Dec(sk, \cdot)$ that receives as input a ciphertext CT and outputs $Dec(sk, CT)$. We denote by $Dec_{\neq C}(sk, \cdot)$ a decryption oracle that decrypts any ciphertext other than C.

Definition 4 (LR-CCA2 security [12]): A public-key encryption scheme $\Pi = (KGen, Enc, Dec)$ is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attack if for any probabilistic polynomial-time λ-key-leakage adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ it holds that

$$\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(n) \stackrel{def}{=} \frac{1}{2}\left|\Pr\left[\mathbf{Expt}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(0) = 1\right] - \Pr\left[\mathbf{Expt}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(1) = 1\right]\right|$$

is negligible in n, where $\mathbf{Expt}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(b)$ is defined as follows

1. $(pk, sk) \leftarrow KGen(1^n)$.
2. $(M_0, M_1, st) \leftarrow \mathcal{A}_1^{\mathrm{Leak}(sk), Dec(sk,\cdot)}(pk)$ such that $|M_0| = |M_1|$.
3. $C \leftarrow Enc_{pk}(M_b)$.
4. $b' \leftarrow \mathcal{A}_2^{Dec_{\neq C}(sk,\cdot)}(C, st)$.
5. Output b'.

3.2 Leakage-Resilient CCA2 Stateful PKE

In this section we review the definition of StPE and its security as given in [2], then present the notion of LR-CCA2

attack in the case of StPE.

Definition 5 (StPE [2]): A StPE scheme consists of the following algorithms:

- **StPE.Setup**$(1^n) \to sp$: Taking 1^n for a security parameter $n \in \mathbb{Z}_{\geq 0}$ as input, this algorithm generates a system parameter sp which includes n.
- **StPE.KGen**$(sp) \to (sk, pk)$: Taking sp as input, this algorithm generates a secret/public key pair (sk, pk).
- **StPE.PKCk**$(sp, pk) \to \delta$: Taking sp and pk as input, this algorithm returns 1 if the public key pk is valid or returns 0 otherwise (i.e., $\delta \in \{0, 1\}$).
- **StPE.NwSt**$(sp) \to st$: Taking sp as input, this algorithm generates a new state.
- **StPE.Enc**$(sp, pk, st, M) \to (C, st)$: Taking sp, pk, st and a plaintext M as input, this algorithm outputs a ciphertext C and state st which may be different from the state provided as input to this algorithm.
- **StPE.Dec**$(sp, sk, C) \to (M)$: Taking sp, sk, and C as input, this deterministic algorithm outputs M which is either a plaintext or $\perp$ (meaning reject) message.

We impose a consistency condition on **StPE**: For any sp output by **StPE.Setup**, (sk, pk) generated by **StPE.KG** and st output by either **StPE.NwSt** or **StPE.Enc**, if (C, st) is an output of **StPE.Enc**(sp, pk, st, M), then **StPE.Dec** $(sp, sk, C) = M$.

We now define the notion of LR-CCA2 attack in the case of StPE, which naturally extends that of CCA2 security for StPE of [2] by giving an adversary the leakage oracle and only allowing him to query this oracle before challenge query.

Note that in the framework of this type of StPE, we can consider the secret key of the receiver and the state issued by the sender as possibly leaking information. Therefore, it seems natural to discuss also the security with state-leakage, in addition to the secret key. However, in this paper, we only focus on the security with (secret) key-leakage.

Definition 6 (LR-CCA2 security of StPE): Let **StPE** be a StPE scheme. Consider a game played with an attacker $\mathcal{A}$:

Phase 1: The game computes $sp \leftarrow$ **StPE.Setup**(1^n), $(pk_1, sk_1) \leftarrow$ **StPE.KGen**(sp) and $st \leftarrow$ **StPE.NwSt** (sp). Note that (sk_1, pk_1) is the secret/public key pair of the honest receiver R_1. The game sends (sp, pk_1) to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ outputs public keys $pk_2, \ldots, pk_t$ of receivers $R_2, \ldots, R_t$ respectively, all of which are in the range of the second element of **StPE.KGen**(sp). Note that $\mathcal{A}$ may or may not know the secret keys corresponding to the public keys $pk_2, \ldots, pk_t$.

Phase 3: $\mathcal{A}$ issues a number of (but polynomially many) queries, each of which is responded by the game. The type of each query and the action taken by the game are described as follows:

- Leakage queries, each of which is denoted by f_j: The game computes $f_j(sk_1)$ and sends this result to $\mathcal{A}$. Note that, the sum of output lengths of all leakage functions is at most λ, and these queries are requested before the challenge query.
- A challenge query (m_0, m_1) such that $|m_0| = |m_1|$: The game picks $b \xleftarrow{\$} \{0, 1\}$, computes $(C^*, st) \leftarrow$ **StPE.Enc**(sp, pk_1, st, m_b), where st denotes the current state, and sends C^* to $\mathcal{A}$.
- Encryption queries, each of which is denoted by (i, M) where $i \in \{1, ..., t\}$: The game computes $(C, st) \leftarrow$ **StPE.Enc**(sp, pk_i, st, M), where st denotes the current state, and sends C to $\mathcal{A}$.
- Decryption queries, each of which is denoted by $C \neq C^*$: The game computes **StPE.Dec**(sp, sk_1, C) and sends the resulting decryption message or $\perp$ (Reject) to $\mathcal{A}$.

Phase 4: $\mathcal{A}$ outputs its guess $b' \in \{0, 1\}$.

We define $\mathcal{A}$'s advantage by

$$\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\mathrm{StPE},\mathcal{A}}(n) \stackrel{\mathrm{def}}{=} \left|\Pr[b' = b] - \frac{1}{2}\right|.$$

The above StPE is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attack if for any probabilistic polynomial-time λ-key-leakage adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ is negligible in n.

The LR-CCA2 security of **StPE** defined above can be considered in the *KSK* (Known Secret Key) or the *USK* (Unknown Secret Key) models [4]. In the *KSK* model, we assume that the attacker $\mathcal{A}$ possesses the corresponding secret keys $sk_2, \ldots, sk_t$ of the public keys output in Phase 2 of the attack game.

3.3 Symmetric Encryption

First, we review the formal definition of symmetric encryption as follows.

Definition 7 (SYM [2]): Let $\mathcal{K}_D$ be the key space. A symmetric encryption scheme, denoted by **SYM**, consists of the following algorithms:

- **SYM.Enc**$(K, M) \to e$: Taking a key $K \in \mathcal{K}_D$ and a plaintext M as input, this algorithm encrypts M into a ciphertext e.
- **SYM.Dec**$(K, e) \to M$: Taking $K \in \mathcal{K}_D$ and e as input, this algorithm decrypts e into M.

To construct LR-CCA2-secure StPE schemes, we need a SYM scheme secure against CCA attack in which the attacker does issue encryption queries. Now, a formal definition follows.

Definition 8 (IND-CCA of SYM [2]): Let **SYM** be a symmetric encryption scheme as defined in Definition 7. Consider a game played with an attacker $\mathcal{A}$:

Phase 1: The game chooses $K \xleftarrow{\$} \mathcal{K}_D$.

Phase 2: $\mathcal{A}$ issues encryption queries, each of which is denoted by M. On receiving this, the game computes

$e \xleftarrow{\$} \mathbf{Enc}(K, M)$ and gives e to $\mathcal{A}$. $\mathcal{A}$ also issues decryption queries, each of which is denoted by e. On receiving this, the game computes $M \leftarrow \mathbf{Dec}(K, e)$ and gives M to $\mathcal{A}$.

Phase 3: $\mathcal{A}$ issues a challenge query (a pair of plaintexts) (m_0, m_1) such that $|m_0| = |m_1|$. On receiving this, the game picks $b \xleftarrow{\$} \{0,1\}$, computes $e^* \xleftarrow{\$} \mathbf{SYM.Enc}(K, m_b)$ and gives e^* to $\mathcal{A}$.

Phase 4: $\mathcal{A}$ continues to issue encryption and decryption queries as in Phase 2. However, a restriction here is that $\mathcal{A}$ is not allowed to issue e^* as decryption query. The game responds to $\mathcal{A}$'s queries in the same way as it did in Phase 2.

Phase 5: $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$.

We define $\mathcal{A}$'s advantage by

$$\mathbf{Adv}^{\text{IND-CCA}}_{\text{SYM},\mathcal{A}}(n) \stackrel{\text{def}}{=} \left|\Pr[b' = b] - \frac{1}{2}\right|.$$

We remark that as mentioned in [4], the SYM schemes meeting the IND-CCA definition can in fact be easily constructed, eg. using the encrypt-then-mac composition [5] with an AES mode of operation (such as CBC) and a MAC (such as CBC-MAC or HMAC [3]).

4. Generic Constructions from Hash Proof Systems

In this section, we give a generic construction of a 1-universal λ-key-leakage extractor that will be used in our constructions. We then present generic constructions of both stateless and stateful PKE scheme that are resilient to LR-CCA2 attack. In the case of stateless PKE, we apply the generic construction of Cramer-Shoup [6] to generalize the work of Naor-Segev. The stateful scheme is an extension from the above stateless scheme.

4.1 The Construction of a 1-Universal λ-Key-Leakage Extractor HPS

Assume that L is a membership indistinguishable language, $\mathbf{HPS} = (Param, KGen, Pub, Priv)$ be a 1-universal HPS for L. Let $\mathbf{Ext}$: $\mathbf{\Pi} \times \{0,1\}^t \rightarrow \{0,1\}^m$ be an average-case $(\log|\mathbf{\Pi}| - \lambda, \epsilon)$-strong extractor. The following describes the 1-universal λ-key-leakage extractor HPS $\mathbf{HPS^{ext}} = (Param^{ext}, KGen^{ext}, Pub^{ext}, Priv^{ext})$ for language $L' = \{(x,s)|x \in L, s \in \{0,1\}^t\}$ (it is easy to see that if L is a membership indistinguishable language, then so is L'):

$Param^{ext}$**:** the same as the algorithm $Param$ of **HPS**.

$KGen^{ext}$**:** On input sp generated by $Param^{ext}$, choose $(pk, sk) \leftarrow KGen(sp)$, and return (pk, sk).

Pub^{ext}**:** On input a public key pk, a pair of variables $((x,s), w)$ with $(x,w) \xleftarrow{\$} \mathcal{R}_L$, and $s \xleftarrow{\$} \{0,1\}^t$, compute $\pi = Pub(pk, x, w)$, $\pi' = \mathbf{Ext}(\pi, s)$. Output π'.

$Priv^{ext}$**:** On input a secret key sk, and (x,s), compute $\pi = Priv(sk, x)$, $\pi' = \mathbf{Ext}(\pi, s)$. Output π'.

The correctness of the scheme follows from the property that $Priv(sk, x) = Pub(pk, x, w)$ for any valid ciphertext $x \in L$ with witness w. Thus, the output of $Priv^{ext}$ is always the key encapsulated by Pub^{ext}.

The following theorem shows that the above HPS is a 1-universal λ-key-leakage extractor. The main idea of the proof was implicit in [12] (see the proof of Theorem 4.1 of [12] for details).

Theorem 1: Assuming that **HPS** is a 1-universal HPS for the language L, and **Ext** is an average-case $(\log|\mathbf{\Pi}| - \lambda(n), \epsilon)$-strong extractor, the hash proof system $\mathbf{HPS^{ext}}$ is a 1-universal $\lambda(n)$-key-leakage extractor for the language L' for any $\lambda(n) \leq \log|\mathbf{\Pi}| - \omega(\log n) - m$, where n is the security parameter and m is the proof size of $\mathbf{HPS^{ext}}$.

Proof. Let $f : \{0,1\}^* \rightarrow \{0,1\}^{\lambda(n)}$ be the function that the adversary used to learn $\lambda(n)$ bits from the secret key.

From the property that $Priv(sk, x) = Pub(pk, x, w)$ for any valid ciphertext $x \in L$ with witness w. We have $Priv^{ext}(sk, x, s) = \mathbf{Ext}(\pi, s) = Pub^{ext}(pk, x, w, s)$. Thus, $\mathbf{HPS^{ext}}$ is an HPS for language $L' = \{(x,s)|x \in L, s \in \{0,1\}^t\}$. To complete the proof, next we show that for any fixed $(x,s) \in X' \setminus L'$ (where $X' = \{(x,s)|x \in X, s \in \{0,1\}^t\}$), it holds that

$$\Delta((Priv^{ext}((x,s), sk), s, pk, f(sk)), (U(\mathbf{\Pi}'), s, pk, f(sk))) \leq \epsilon.$$

The adversary learns at most $\lambda(n) (= \log(f(sk)))$ bits of leakage, therefore from Lemma 1 we have

$$\widetilde{\mathrm{H}}_\infty(\pi \mid pk, f(sk)) \geq \mathrm{H}_\infty(\pi \mid pk) - \log(f(sk)) = \log|\mathbf{\Pi}| - \lambda(n).$$

This is derived from the 1-universality of **HPS**, that is,

$$\widetilde{\mathrm{H}}_\infty(\pi \mid pk) = \Pr[Priv(sk, x) = \pi] = \frac{1}{|\mathbf{\Pi}|}.$$

Now, from the assumption that **Ext** is an average-case $(\log|\mathbf{\Pi}| - \lambda(n), \epsilon)$-strong extractor and the definition of the strong extractor (Definition 1), it holds that

$$\Delta((\mathbf{Ext}(\pi, s), s, pk, f(sk)), (U(\mathbf{\Pi}'), s, pk, f(sk))) \leq \epsilon.$$

By replacing $\mathbf{Ext}(\pi, s)$ with $Priv^{ext}((x,s), sk)$, we have

$$\Delta((Priv^{ext}((x,s), sk), s, pk, f(sk)), (U(\mathbf{\Pi}'), s, pk, f(sk))) \leq \epsilon.$$

Therefore, $\mathbf{HPS^{ext}}$ satisfies the condition of a 1-universal $\lambda(n)$-key-leakage extractor.

Now, from the fact that strong extractor can extract at most $m \leq \log|\mathbf{\Pi}| - \lambda(n) - \omega(\log n)$ nearly random bits, we find out the bound of $\lambda(n)$ as follow: $\lambda(n) \leq \log|\mathbf{\Pi}| - \omega(\log n) - m$. □

4.2 The Construction of Stateless PKE

Assume that L is a membership indistinguishable language,

$\mathbf{HPS_1} = (Param_1, KGen_1, Pub_1, Priv_1)$ be a 1-universal λ-key-leakage extractor HPS for a language L, and $\mathbf{HPS_2} = (Param_2, KGen_2, Pub_2, Priv_2)$ be an extended 2-universal HPS for the same language L. We define an encryption scheme $\Pi = (KGen, Enc, Dec)$ as follows:

Key Generation: On input 1^n for $n \in \mathbb{Z}_{\geq 0}$, generate system parameter $sp = (sp_1, sp_2)$, where $sp_1 \leftarrow Param_1(1^n)$, $sp_2 \leftarrow Param_2(1^n)$. Choose $(pk_1, sk_1) \leftarrow KGen_1(sp_1)$, $(pk_2, sk_2) \leftarrow KGen_2(sp_2)$, and return $pk = (pk_1, pk_2)$, $sk = (sk_1, sk_2)$.

Encryption: Given 1^n for $n \in \mathbb{Z}_{\geq 0}$, a public key $pk = (pk_1, pk_2)$, along with a message $M \in \mathbf{\Pi}_1$ (where $\mathbf{\Pi}_1$ is the domain of Pub_1; it may be, for example, $\{0,1\}^m$), do as follows.

E0: Choose a pair $(x, w) \xleftarrow{\$} \mathcal{R}_L$.
E1: $\pi_1 = Pub_1(pk_1, x, w)$.
E2: $e = M \oplus \pi_1$.
E3: $\pi_2 = Pub_2(pk_2, x, w, e)$.
E4: Output $c = (x, e, \pi_2)$.

Decryption: Given 1^n for $n \in \mathbb{Z}_{\geq 0}$, a secret key $sk = (sk_1, sk_2)$, along with a ciphertext c, do the following.

D0: Parse c as a 3-tuple (x, e, π_2); output $\perp$ if c is not of this form.
D1: Compute $\pi_2' = Priv_2(sk_2, x, e)$.
D2: Test if $\pi_2' = \pi_2$; output $\perp$ and halt if this is not the case.
D3: Compute $\pi_1 = Priv_1(sk_1, x)$.
D4: Output $M = e \oplus \pi_1$.

Next, we show that this PKE scheme is LR-CCA2-secure.

Theorem 2: Assume that L is a membership indistinguishable language, $\mathbf{HPS_1}$ is a 1-universal λ-key-leakage extractor for L, and $\mathbf{HPS_2}$ is an extended 2-universal HPS for L, with proofs π_2 of size $|\pi_2| = p \geq \lambda + \omega(\log n)$. Then the encryption scheme constructed from $\mathbf{HPS_1}, \mathbf{HPS_2}$ is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attacks, where n denotes the security parameter.

Proof. Before starting to prove, we state the following lemma, which explicitly appeared in [7].

Lemma 2 (Lemma 4 of [7]): *Let X_1, X_2 and F be events defined on some probability space. Suppose that the event $X_1 \wedge \neg F$ occurs if and only if $X_2 \wedge \neg F$ occurs. Then* $|\Pr[X_1] - \Pr[X_2]| \leq \Pr[F]$.

Let $f : \{0,1\}^* \rightarrow \{0,1\}^\lambda$ be the function that the adversary used to learn λ bits from the secret key.

Let $\mathcal{A}$ be an adversary on the LR-CCA2 security of the scheme. We will consider a sequence of games, Game $\mathbf{G_0}$, Game $\mathbf{G_1}$,..., each game involving $\mathcal{A}$. Let T_i be the event that in Game $\mathbf{G_i}$, it holds that $b = b'$, i.e., that the adversary succeeds.

Game $\mathbf{G_0}$: This is the original LR-CCA2 security game. The adversary $\mathcal{A}$ has access to a decryption oracle $Dec(sk, \cdot)$ and can query leakage functions, gets a target ciphertext $c^* = (x^*, e^*, \pi_2^*)$ of message M_b, where c^* $(= Enc(pk, M_b))$ is computed as follows:

1. $(x^*, w^*) \xleftarrow{\$} \mathcal{R}_L$.
2. $\pi_1^* = Pub_1(pk_1, x^*, w^*)$.
3. $e^* = M_b \oplus \pi_1^*$.
4. $\pi_2^* = Pub_2(pk_2, x^*, w^*, e^*)$.

It is clear that

$$\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\Pi,\mathcal{A}}(n) = |\Pr[T_0] - 1/2|. \tag{1}$$

Game $\mathbf{G_1}$: We now modify game $\mathbf{G_0}$ to obtain a new game $\mathbf{G_1}$. These two games are identical, except for a small modification to the encryption oracle. Instead of using the encryption algorithm as given to compute the target ciphertext c^*, we use a modified encryption algorithm, in which steps **E1** and **E3** are replaced by:
E1′: $\pi_1^* = Priv_1(sk_1, x^*)$.
E3′: $\pi_2^* = Priv_2(sk_2, x^*, e)$.

The change we have made is purely conceptual: the values of π_1^* and π_2^* are exactly the same in game $\mathbf{G_1}$ as they were in $\mathbf{G_0}$. Therefore,

$$\Pr[T_1] = \Pr[T_0]. \tag{2}$$

Note that the encryption oracle now makes use of some components of the secret key, which is something the original encryption oracle does not do.

Game $\mathbf{G_2}$: We now modify game $\mathbf{G_1}$ to obtain a new game $\mathbf{G_2}$. We modify the challenge ciphertext, replacing step **E0** of the encryption algorithm by **E0′**: $x^* \xleftarrow{\$} X \setminus L$.

By the membership indistinguishability property of the language L, Games $\mathbf{G_1}$ and $\mathbf{G_2}$ are indistinguishable. Indeed, we show that the existence of an adversary that is able to distinguish the above Games with non-negligible advantage implies the existence of an efficient distinguishing algorithm that contradicts the hardness assumption for L.

We define the following game between a *simulator* and an adversary that plays one of the above Games ($\mathbf{G_1}$ or $\mathbf{G_2}$). The simulator takes as input 1^n, for $n > 0$, along with $\mathbf{HPS_1}, \mathbf{HPS_2}$, and $x^* \in X$.

The simulator provides a "simulated environment" for the adversary the same as Game $\mathbf{G_1}$ using $\mathbf{HPS_1}, \mathbf{HPS_2}$.

In the challenge phase, the attacker presents messages M_0 and M_1 of his choice to the simulator. The simulator flips a random coin σ, and computes the target ciphertext $c^* = (x^*, e^*, \pi_2^*)$ of message M_σ where x^* is the value input to the simulator. Note that, in Game $\mathbf{G_1}$ the values π_1^*, π_2^* are computed by using $Priv_1$, $Priv_2$, and sk_1, sk_2, so they are correctly computed whether or not x is in L.

Finally, the adversary outputs a bit β to show the Game ($\mathbf{G}_\beta$) which he has played with. The simulator outputs 1 if $\beta = 1$ (mean that he has played with Game $\mathbf{G_1}$), and 0 otherwise, after which, the simulator halts.

It is easy to see that the simulation is perfect. Therefore, we have

$$|\Pr[T_2] - \Pr[T_1]| \leq negl(n). \tag{3}$$

Game $\mathbf{G_3}$: In this game, we modify the decryption oracle in game $\mathbf{G_2}$ to obtain a new game $\mathbf{G_3}$. Instead of using the original decryption algorithm, we modify the decryption algorithm, replacing step **D2** with:

D2′: Test if $x \in L$; output $\perp$ and halt if this is not the case. If $x \in L$, the oracle runs as previously.

Now, let R_3 be the event that in game $\mathbf{G_3}$, some ciphertext C is submitted to the decryption oracle that is rejected in step **D2′** but that would have passed the test in step **D2**.

Note that if a ciphertext passes the test in **D2′**, it would also have passed the test in **D2**.

It is clear that games $\mathbf{G_2}$ and $\mathbf{G_3}$ proceed identically until the event R_3 occurs. In particular, the event $T_2 \wedge \neg R_3$ and $T_3 \wedge \neg R_3$ are identical. So by Lemma 2, we have

$$| \Pr[T_3] - \Pr[T_2] | \leq \Pr[R_3],$$

and therefore it suffices to bound $\Pr[R_3]$.

Claim 1: If $\mathbf{HPS_2}$ is an extended 2-universal HPS for L, with proofs π_2 of size $|\pi_2| = p \geq \lambda + \omega(\log n)$, then the event R_3 occurs with only a negligible probability.

Proof of Claim 1. Denote by $c_1 = (x_1, e_1, \pi_{2_1})$ the first invalid ciphertext submitted by the adversary. Denote by **view** the view of the adversary prior to submitting the invalid ciphertext. Then, **view** $= \{pk, x_1, e_1, f(sk), x^*, e^*, \pi_2^* = Priv_2(sk_2, x^*, e^*)\}$. The adversary learns at most λ bits of leakage, and therefore (see Lemma 1)

$$\begin{aligned}\widetilde{\mathrm{H}}_\infty(\pi_{2_1} \mid \mathbf{view}) &\geq \mathrm{H}_\infty(\pi_{2_1} \mid \mathbf{view} \setminus \{f(sk)\}) - \log(f(sk)) \\ &= \mathrm{H}_\infty(\pi_{2_1} \mid \mathbf{view} \setminus \{f(sk)\}) - \lambda \\ &= p - \lambda\end{aligned}$$

This is derived from the 2-universality of $\mathbf{HPS_2}$, that is

$$\Pr[Priv_2(sk_2, x_1, e_1) = \pi_{2_1} \mid Priv_2(sk_2, x^*, e^*) = \pi_2^*] = \frac{1}{|\pi_2|} = \frac{1}{p}.$$

In particular, the definition of average min-entropy implies that prior to submitting the invalid ciphertext the probability of $\mathcal{A}$ in guessing π_{2_1} is at most $2^{-\widetilde{\mathrm{H}}_\infty(\pi_{2_1} | \mathbf{view})} \leq 2^{\lambda - p}$. Thus, the probability that the decryption algorithm accepts the first invalid ciphertext is at most $2^{\lambda-p} = 2^\lambda / 2^p$.

An almost identical argument holds for all the subsequent invalid decryption queries. The only difference is that each time the decryption oracle rejects an invalid ciphertext the adversary can rule out one more value of π_2. This shows that the decryption algorithm accepts the i-th invalid ciphertext with probability at most $2^\lambda / (2^p - i + 1)$.

Assume that $\mathcal{A}$ made at most q decryption queries, where $q = q(n)$ is polynomial in n.

We have

$$\Pr[R_3] \leq \frac{2^\lambda}{2^p} + \ldots + \frac{2^\lambda}{2^p - q + 1} \leq \frac{q \cdot 2^\lambda}{2^p - q + 1}.$$

From the assumption that $p \geq \lambda + \omega(\log n)$, we have

$$\Pr[R_3] \leq \frac{q \cdot 2^{p - \omega(\log n)}}{2^p - q + 1} = \frac{q \cdot 2^{-\omega(\log n)}}{1 - \frac{q-1}{2^p}} < \frac{q\epsilon}{1 - 1/2} = 2q\epsilon.$$

The claim follows. □

Therefore, we have

$$| \Pr[T_3] - \Pr[T_2] | \leq \Pr[R_3] \leq negl(n). \quad (4)$$

Game $\mathbf{G_4}$: This game is identical to game $\mathbf{G_3}$, except for a small modification to the encryption oracle. In the challenge phase, replacing step **E1′** by **E1″**: $\pi_1^* \stackrel{\$}{\leftarrow} \mathbf{\Pi}_1$

It is clear by construction that

$$\Pr[T_4] = 1/2, \quad (5)$$

since in game $\mathbf{G_4}$, the variable b is never used at all, and so the adversary's output is independent of b.

Claim 2:

$$| \Pr[T_4] - \Pr[T_3] | \leq negl(n). \quad (6)$$

Proof of Claim 2. Now, let us condition on fixed values of sk_2, b, and the adversary's coins. In this conditional probability space, since the simulator rejects all ciphertexts (x, e, π_2) with $x \notin L$, it follows that the output of the simulator in game $\mathbf{G_3}$ is completely determined as a function of $pk_1, x^*, f(sk_1)$, and $Priv_1(sk_1, x^*)$, while the output in game $\mathbf{G_4}$ is determined as the same function of $pk_1, x^*, f(sk_1)$, and $\pi_1^* \in_R \mathbf{\Pi}_1$. Moreover, by independence, the joint distribution of $(pk_1, x^*, f(sk_1), \pi_1^*)$ does not change in passing from the original probability space to the conditional probability space. It now follows directly from the assumption $\mathbf{HPS_1}$ is a 1-universal λ-key-leakage extractor for L that

$$\begin{aligned}&\Delta((pk_1, x^*, f(sk_1), Priv_1(x^*, sk_1)), \\ &\quad (pk_1, x^*, f(sk_1), \mathrm{U}(\mathbf{\Pi}_1))) \leq negl(n).\end{aligned}$$

The claim follows. □

The theorem now follows immediately from (1)–(6). □

Remark 1. Although our construction is a generalization of the Naor-Segev's scheme, it is not straightforward extension. The main technical difference is that to bound $\Pr[R_3]$ (Claim 1) we analyze the probability of the adversary in guessing $\pi_{2_i} = Priv_2(sk_2, x_i, e_i)$, where $x_i \leftarrow X \setminus L$, (it is the reason why we need the condition on the length of π_2), and hence our analysis holds in the general case. Whereas, Naor and Segev analyze the probability of the adversary in guessing some component parts of the secret key (see Claim 6.12 of [12] for more details) and hence they analyze a very specific case.

4.3 The Construction of StPE

Assume that $L, \mathbf{HPS_1}, \mathbf{HPS_2}$ are components as in Sect. 4.2, and **SYM** be a SYM scheme. We assume that the HPS scheme $\mathbf{HPS_1}$ and the SYM scheme **SYM** are "compatible" meaning that the key space $\mathcal{K}_K$ of $\mathbf{HPS_1}$ is the same as the key space $\mathcal{K}_D$ of **SYM**. We define a StPE scheme **StPE** as

follows:

StPE.Setup: On input 1^n for $n \in \mathbb{Z}_{\geq 0}$, return (system parameter) $sp = (sp_1, sp_2)$, where $sp_1 \leftarrow Param_1(1^n)$, $sp_2 \leftarrow Param_2(1^n)$.

StPE.KGen: On input sp, choose $(pk_1, sk_1) \leftarrow KGen_1(sp_1)$ and $(pk_2, sk_2) \leftarrow KGen_2(sp_2)$. Then return $PK = (pk_1, pk_2)$, $SK = (sk_1, sk_2)$.

StPE.PKCk: On input sp and $PK = (pk_1, pk_2)$, output 1.

StPE.NwSt: On input sp, execute the instance sampling algorithm of L by **E0:**$(x, w) \xleftarrow{\$} \mathcal{R}_L$, and return $st = (x, w)$.

StPE.Enc: On input sp, a public key $PK = (pk_1, pk_2)$, a state st, along with a message M, do the following.
If st is of the form (x, w) then compute **E1:** $\pi_1 = Pub_1(pk_1, x, w)$; else, parse st as (x, w, PK, π_1). Next, do as follows.

- **E2:** $e =$ **SYM.Enc**(π_1, M).
- **E3:** $\pi_2 = Pub_2(pk_2, x, w, e)$.
- **E4:** Output $c = (x, e, \pi_2)$ and the new state $st = (x, w, PK, \pi_1)$.

StPE.Dec: On input sp, a secret key $SK = (sk_1, sk_2)$, along with a ciphertext c, do the following.

- **D0:** Parse c as a 3-tuple (x, e, π_2); output $\perp$ if c is not of this form.
- **D1:** Compute $\pi_2' = Priv_2(sk_2, x, e)$.
- **D2:** Test if $\pi_2' = \pi_2$; output $\perp$ and halt if this is not the case.
- **D3:** Compute $\pi_1 = Priv_1(sk_1, x)$.
- **D4:** Output $M =$ **SYM.Dec**(π_1, e).

Note that, **StPE.PKCk** returns 1 (and does nothing else) as the *KSK* model implies that any public keys in this system are generated correctly following the algorithm **StPE.KGen**. (Namely the entity that has generated a public key must know the corresponding secret key.)

In our construction of StPE, there are two types of state: 1) (x, w) which is output by **StPE.NwSt**; 2)(x, w, PK, π_1) which is produced by the algorithm **StPE.Enc**. Note also that for state $st = (x, w)$ generated by the algorithm **StPE.NwSt**, $[$**StPE.Enc**$(PK, st, M)]_C = [$**StPE.Enc**$(PK, st', M)]_C$ for any st' output by **StPE.Enc** before **StPE.NwSt** is invoked to generate new state (different from st). Here "$[$**StPE.Enc**$(\cdots)]_C$" denotes the ciphertext part of an output of **StPE.Enc**.

We remark that the algorithm **StPE.Enc** becomes *highly efficient* when a sender sends encryptions to a single receiver: If the sender wants to send encryptions of M_1, ..., M_n to the same receiver whose public key is PK, he does not have to run Pub_1 for each plaintext M_i for $i = 1, \ldots, n$ but just runs them once at the beginning and then only does steps **E2, E3, E4** (in **StPE.Enc**).

Next, we show that in the *KSK* model, the above scheme is LR-CCA2-secure.

Theorem 3: Assume that L is a membership indistinguishable language, $\mathbf{HPS_1}$ is a 1-universal λ-key-leakage extractor for L, $\mathbf{HPS_2}$ is an extended 2-universal HPS for L, with proofs π_2 of size $|\pi_2| = p \geq \lambda + \omega(\log n)$, and the underlying symmetric encryption **SYM** is IND-CCA-secure. Then in the *KSK* model, the proposed generic StPE scheme **StPE** is semantically secure against a-posteriori chosen-ciphertext λ-key-leakage attacks. More precisely, we have

$$\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\mathrm{StPE},\mathcal{A}}(n) \leq \mathbf{Adv}^{\mathrm{IND\text{-}CCA}}_{\mathrm{SYM},\mathcal{B}}(n),$$

where n denotes the security parameter.

Proof. Let $f : \{0,1\}^* \rightarrow \{0,1\}^\lambda$ be the function that the adversary used to learn λ bits from the secret key.

Let $\mathcal{A}$ be an adversary on the LR-CCA2 security of the scheme. We will consider a sequence of games, Game $\mathbf{G_0}$, Game $\mathbf{G_1}$,..., each game involving $\mathcal{A}$. Let T_i be the event that in Game $\mathbf{G_i}$, it holds that $b = b'$, i.e., that the adversary succeeds.

Game $\mathbf{G_0}$: This is the original LR-CCA2 security game. We repeat this game to clean up the notations. Let sp be a system parameter. Let PK_1 and SK_1 be public and secret keys of the honest receiver respectively. Let $PK_2, \ldots, PK_t$ be the public keys output by $\mathcal{A}$. Let $st = (x^*, w^*)$, where $(x^*, w^*) \xleftarrow{\$} \mathcal{R}_L$, be the sender's state generated by **StPE.NwSt**, fixed throughout each game. We denote a challenge ciphertext by $c^* = (x^*, e^*, \pi_2^*)$. Where c^* (= **StPE.Enc**(pk, st, M_b)) is computed as follows:

1. $\pi_1^* = Pub_1(pk_1, x^*, w^*)$.
2. $e^* =$ **SYM.Enc**(π_1, M_b).
3. $\pi_2^* = Pub_2(pk_2, x^*, w^*, e^*)$.

Now, observe that we can assume that $\mathcal{A}$ does not make encryption queries of the form (i, M) for $i = 2, \ldots, t$. The reason is that since $\mathcal{A}$ is assumed to know SK_i corresponding to its public key PK_i following the *KSK* model, it, given (x^*, PK_1), can also compute $c_i = (x^*, e_i, \pi_{2_i})$ (for all $i = 2, \ldots, t$) as follows:

1. $\pi_{1_i} = Priv_1(sk_{i1}, x^*)$.
2. $e_i =$ **SYM.Enc**(π_{1_i}, M_b).
3. $\pi_{2_i} = Priv_2(sk_{i2}, x^*, e_i)$.

Since $\mathbf{G_0}$ is the original LR-CCA2 game of **StPE**, we have

$$\mathbf{Adv}^{\mathrm{KL,CCA2}}_{\mathrm{StPE},A}(n) = |\Pr[T_0] - 1/2|. \tag{7}$$

Game $\mathbf{G_1}$: We now modify game $\mathbf{G_0}$ to obtain a new game $\mathbf{G_1}$. These two games are identical, except for a small modification to the encryption oracle. Instead of using the encryption algorithm as given to compute the target ciphertext c^*, we use a modified encryption algorithm, in which steps **E1** and **E3** are replaced by:

- **E1′:** $\pi_1^* = Priv_1(sk_1, x)$.
- **E3′:** $\pi_2^* = Priv_2(sk_2, x, e)$.

The change we have made is purely conceptual: the values of π_1^* and π_2^* are exactly the same in game $\mathbf{G_1}$ as they were in $\mathbf{G_0}$. Therefore,

$$\Pr[T_1] = \Pr[T_0]. \tag{8}$$

Note that the encryption oracle now makes use of some components of the secret key, which is something the original encryption oracle does not do.

Game $\mathbf{G_2}$: We now modify game $\mathbf{G_1}$ to obtain a new game $\mathbf{G_2}$. We modify the encryption oracle, replacing step **E0** by **E0′**: $x^* \xleftarrow{\$} X \setminus L$.

The same as proof of Theorem 2, by the membership indistinguishability property of the language L, Games $\mathbf{G_1}$ and $\mathbf{G_2}$ are indistinguishable. Therefore,

$$|\Pr[T_2] - \Pr[T_1]| \leq negl(n). \tag{9}$$

Game $\mathbf{G_3}$: In this game, we modify the decryption oracle in game $\mathbf{G_2}$ to obtain a new game $\mathbf{G_3}$. Instead of using the original decryption algorithm, we modify the decryption algorithm, replacing step **D2** with:

D2′: Test if $x \in L$; output $\perp$ and halt if this is not the case. If $x \in L$, the oracle runs as previously.

Similar to the case of stateless PKE (Theorem 2), we proved that in this game $\mathbf{G_3}$, it holds that

$$|\Pr[T_3] - \Pr[T_2]| \leq negl(n). \tag{10}$$

Game $\mathbf{G_4}$: This game is identical to game $\mathbf{G}_3$, except for a small modification to the encryption oracle. We again modify the algorithm used by the encryption oracle, replacing step **E1′** by **E1″**: $\pi_1^* \xleftarrow{\$} \mathbf{\Pi}_1$.

Claim 3:

$$|\Pr[T_4] - \Pr[T_3]| \leq negl(n). \tag{11}$$

Proof of Claim 3. Now, let us condition on fixed values of sk_2, b, the simulator's coins (i.e., the coins which are used in the algorithm **SYM**.**Enc**) and the adversary's coins. In this conditional probability space, since the actions of Pub_1 and $Priv_1$ on L are determined by pk_1, and since the simulator rejects all ciphertexts (x, e, π_2) with $x \notin L$, it follows that the output of the simulator in game $\mathbf{G_3}$ is completely determined as a function of $pk_1, x^*, f(sk_1)$, and $Priv_1(sk_1, x^*)$, while the output in game $\mathbf{G_4}$ is determined as the same function of $pk_1, x^*, f(sk_1)$, and $\pi_1^* \in_R \mathbf{\Pi}_1$. Moreover, by independence, the joint distribution of $(pk_1, x^*, f(sk_1), \pi_1^*)$ does not change in passing from the original probability space to the conditional probability space. It now follows directly from the assumption $\mathbf{HPS_1}$ is a 1-universal λ-key-leakage extractor for L that

$$\Delta((pk_1, x^*, f(sk_1), Priv_1(x^*, sk_1)), \\ (pk_1, x^*, f(sk_1), \mathrm{U}(\mathbf{\Pi}_1))) \leq negl(n).$$

The claim follows. □

Claim 4:

$$|\Pr[T_4] - \frac{1}{2}| \leq \mathbf{Adv}^{\text{IND-CCA}}_{\text{SYM},B}(n). \tag{12}$$

Where B denote the attacker against the underlying symmetric encryption **SYM** .

Proof of Claim 4. To prove Claim 4 we construct an oracle machine B that breaks IND-CCA security of **SYM** using the attacker $\mathcal{A}$ as a subroutine.

Algorithm $B(n)$:

Generate sp, SK_1, and PK_1, then give (sp, PK_1) to $\mathcal{A}$, choose randomly $x^* \xleftarrow{\$} X \setminus L$.
If $\mathcal{A}$ issues a challenge query (M_0, M_1) such that $|M_0| = |M_1|$, then

Query (M_0, M_1) to the challenger to get $e^* \xleftarrow{\$}$ **SYM**.**Enc**(K_0^*, M_b) where $b \xleftarrow{\$} \{0, 1\}$; compute $\pi_2^* = Priv_2(sk_2, x^*, e^*)$; and give $c^* = (x^*, e^*, \pi_2^*)$ to $\mathcal{A}$.

If $\mathcal{A}$ issues an encryption query $(1, M)$, then

Query M to the challenger to get $e \xleftarrow{\$}$ **SYM**.**Enc**(K_0^*, M), compute $\pi_2 = Priv_2(sk_2, x^*, e)$, and give $c = (x^*, e, \pi_2)$ to $\mathcal{A}$.

If $\mathcal{A}$ issues a decryption query $c \neq c^*$, where $c = (x, e, \pi_2)$, then

If $\pi_2 \neq \pi_2^*$ then

$\pi_1 \leftarrow Priv_1(sk_1, x^*)$;
If $\pi_1 \neq \perp$, then give $M =$ **SYM**.**Dec**(π_1, e) to $\mathcal{A}$; else return $\perp$.

Else

Query e (which must be different from e^*) to the challenger to get $M \leftarrow$ **SYM**.**Enc**(K_0^*, e), and give M to $\mathcal{A}$.

If $\mathcal{A}$ outputs b', then return b'.

Observe that in the above algorithm B, $\mathcal{A}$ is essentially conducting chosen ciphertext attack on the SYM scheme **SYM**. Thus we have

$$|\Pr[T_4] - \frac{1}{2}| \leq \mathbf{Adv}^{\text{IND-CCA}}_{\text{SYM},B}(n).$$

The claim follows. □
The theorem now follows immediately from (7)–(12). □

Remark 2. In this work, we only focus on the security with key-leakage, other than state-leakage (Definition 6). Thus, we can consider that the secret key π_1 of **SYM**, which is included in the state st, is not leaked. This as well as our analysis in Claim 3 show the reason why leakage resilient symmetric encryption is not needed in the above scheme.

Remark 3. It follows from Theorems 1 and 2 (3) that the above constructions are LR-CCA2-secure with the leakage rate at most $\frac{\min\{|\pi_1|,|\pi_2|\}}{|sk_1|+|sk_2|}$ of the secret-key length, where $|a|$ denotes the size of a. An efficient instantiation of the proposed construction of stateless PKE is the encryption scheme of Naor-Segev [12], which is LR-CCA2-secure with the leakage rate of 1/6. Since the StPE scheme can also be constructed from the same HPSs, we obtained an efficient instantiation of the proposed StPE with the same leakage rate.

5. Example: The Efficient LR-CCA2-Secure Scheme of Naor-Segev

In this section, we first review the decisional Diffie-Hellman assumption (DDH). Then we present and analyze the Naor-Segev's scheme in the framework of our general construction.

The DDH assumption. The DDH assumption is that the ensembles $\{(\mathbb{G}, g_1, g_2, g_1^r, g_2^r)\}$ and $\{(\mathbb{G}, g_1, g_2, g_1^{r_1}, g_2^{r_2})\}$ are indistinguishable, where $\mathbb{G}$ is a group of order q, and the elements $g_1, g_2 \in \mathbb{G}$ and $r, r_1, r_2 \in \mathbb{Z}_q$ are chosen independently and uniformly at random.

Notations. Let $\mathbb{G}$ be a group of prime order q, g_0 and g_1 be randomly chosen elements of $\mathbb{G}$. Define $X = \mathbb{G} \times \mathbb{G}$, and L be the subgroup of X generated by $(g_1, g_2) \in X$. A witness for $(x_1, x_2) \in L$ is $w \in \mathbb{Z}_q$ such that $(x_1, x_2) = (g_1^w, g_2^w)$. Let $\lambda = \lambda(n)$ be the leakage parameter, let $\mathbf{Ext}$: $\mathbf{\Pi} \times \{0,1\}^t \to \{0,1\}^m$ be an average-case $(\log q - \lambda, \epsilon)$-strong extractor, and let $\mathcal{H}$ be a family of universal one-way hash functions $H : \mathbb{G}^3 \to \mathbb{Z}_q$. Rompel [14] showed that universal one-way hash functions can be constructed from one-way functions, and in particular, such functions exist based on the hardness of DDH.

The scheme. We now present the encryption scheme of Naor-Segev with a secret key of size essentially $6 \log q$ bits (six group elements), and it was showed in [12] that the scheme is secure for any leakage of length $\lambda \leq \log q - \omega(\log n) - m$, where m is the length of the plaintext. Now, we analyse the scheme in the framework of our general construction. First, we present $\mathbf{HPS}_1$ the 1-universal λ-key-leakage extractor for L, and $\mathbf{HPS}_2$ the extended 2-universal HPS for L. Then we present the stateless public-key encryption scheme constructed from $\mathbf{HPS}_1$, $\mathbf{HPS}_2$, which is identical to the scheme of Naor-Segev (the stateful encryption scheme can also be constructed from the same HPSs, so we do not present here). From Theorem 2, it is obviously that the constructed scheme is semantically secure against λ-key-leakage CCA2 attacks.

The following describes the 1-universal λ-key-leakage extractor $\mathbf{HPS}_1$.

$\mathbf{HPS}_1$:

$Param_1$: On input 1^n for $n \in \mathbb{Z}_{\geq 0}$, choose $g_1, g_2 \in \mathbb{G}$, and $H \in \mathcal{H}$ uniformly at random. Then output $sp_1 = (g_1, g_2, H)$.

$KGen_1$: On input a system parameter sp_1, choose $z_1, z_2 \in \mathbb{Z}_q$, uniformly at random. Compute $h = g_1^{z_1} g_2^{z_2}$, and output $pk_1 = h, sk_1 = (z_1, z_2)$.

Pub_1: On input a public key pk_1, a group of variables (u_1, u_2, r, s) with $r \in \mathbb{Z}_q$, $u_1 = g_1^r, u_2 = g_2^r$, and $s \xleftarrow{\$} \{0,1\}^t$, compute $\pi_1 = Pub_1(pk_1, u_1, u_2, r, s) = \mathbf{Ext}(h^r, s)$; then output π_1.

$Priv_1$: On input a private key sk_1, and (u_1, u_2, s), compute $\pi_1 = Priv_1(sk_1, u_1, u_2, s) = \mathbf{Ext}(u_1^{z_1} u_2^{z_2}, s)$, and output π_1.

The hash proof system underlying the above scheme is the one described in [11]. This hash proof system is 1-universal based on the DDH assumption, and as an immediate consequence we obtain the following corollary of Theorem 1:

Corollary 1: Assuming the hardness of DDH, and $\mathbf{Ext}$ is an average-case $(\log q - \lambda(n), \epsilon)$-strong extractor, the above hash proof system is a 1-universal $\lambda(n)$-key-leakage extractor for the language L for any $\lambda(n) \leq \log q - \omega(\log n) - m$, where n is the security parameter and m is the proof size of $\mathbf{HPS}_1$.

Next, we describe the extended 2-universal $\mathbf{HPS}_2$.

$\mathbf{HPS}_2$:

$Param_2$: On input 1^n for $n \in \mathbb{Z}_{\geq 0}$, choose $g_1, g_2 \in \mathbb{G}$, and $H \in \mathcal{H}$ uniformly at random. Then output $sp_2 = (g_1, g_2, H)$.

$KGen_2$: On input a system parameter sp_2, choose $x_1, x_2, y_1, y_2 \in \mathbb{Z}_q$, uniformly at random. Let $c = g_1^{x_1} g_2^{x_2}, d = g_1^{y_1} g_2^{y_2}$. Output $pk_2 = (c, d), sk_2 = (x_1, x_2, y_1, y_2)$.

Pub_2: On input a public key pk_1, a group of variables (u_1, u_2, r, e) with $r \in \mathbb{Z}_q$, $u_1 = g_1^r, u_2 = g_2^r$, and $e \in \mathbb{G}$, compute $\pi_2 = Pub_2(pk_2, u_1, u_2, r, e) = c^r d^{r\alpha}$, where $\alpha = H(u_1, u_2, e)$. Output π_2.

$Priv_2$: On input a private key sk_1, and (u_1, u_2, e), compute Compute $\alpha = H(u_1, u_2, e)$ and $\pi_2 = Priv_2(sk_2, u_1, u_2, e) = u_1^{x_1 + y_1\alpha} u_2^{x_2 + y_2\alpha}$. Output π_2.

Applying the construction in Theorem 3 of [7] with one change that instead of the injective encoding function Γ we use the universal one-way hash functions H, we can easily prove that the above hash proof system is the extended 2-universal.

The following describes the stateless PKE scheme:

Key Generation:

On input 1^n for $n \in \mathbb{Z}_{\geq 0}$, choose $x_1, x_2, y_1, y_2, z_1, z_2 \in \mathbb{Z}_q$, $g_1, g_2 \in \mathbb{G}$, and $H \in \mathcal{H}$ uniformly at random. Let $c = g_1^{x_1} g_2^{x_2}, d = g_1^{y_1} g_2^{y_2}, h = g_1^{z_1} g_2^{z_2}$, then $pk_1 = h, sk_1 = (z_1, z_2)$, and $pk_2 = (c, d), sk_2 = (x_1, x_2, y_1, y_2)$. Finally, output a public key $pk = (g_1, g_2, c, d, h, H)$, and a secret key $sk = (x_1, x_2, y_1, y_2, z_1, z_2)$.

Encryption:

Given 1^n for $n \in \mathbb{Z}_{\geq 0}$, a public key $pk = (pk_1, pk_2)$, along with a message $M \in \{0,1\}^m$, compute

- **E0:** Choose $r \in \mathbb{Z}_q$ and $s \in \{0,1\}^t$ uniformly at random, and compute $u_1 = g_1^r, u_2 = g_2^r$.
- **E1:** $\pi_1 = Pub_1(pk_1, u_1, u_2, r, s) = \mathbf{Ext}(h^r, s)$;
- **E2:** $e = M \oplus \pi_1$;
- **E3:** $\pi_2 = Pub_2(pk_2, u_1, u_2, r, e) = c^r d^{r\alpha}$, where $\alpha = H(u_1, u_2, e)$;

E4: Output $c = (u_1, u_2, e, s, \pi_2)$.

Decryption:

Given 1^n for $n \in \mathbb{Z}_{\geq 0}$, a secret key $sk = (sk_1, sk_2)$, along with a ciphertext c, do the following.

D0: Parse c as (u_1, u_2, e, s, π_2); output $\perp$ if c is not of this form.

D1: Compute $\alpha = H(u_1, u_2, e)$ and $\pi'_2 = Priv_2(sk_2, u_1, u_2, e) = u_1^{x_1+y_1\alpha} u_2^{x_2+y_2\alpha}$.

D2: Test if $\pi'_2 = \pi_2$; output $\perp$ and halt if this is not the case.

D3: Compute $\pi_1 = Priv_1(sk_1, u_1, u_2) = \mathbf{Ext}(u_1^{z_1} u_2^{z_2}, s)$.

D4: Output $M = e \oplus \pi_1$.

Correctness. For any sequence of coin tosses of the key generation and encryption algorithms it holds that $u_1^{x_1+y_1\alpha} u_2^{x_2+y_2}\alpha = c^r d^{r\alpha} = \pi_2$ and that $u_1^{z_1} u_2^{z_2} = h^r$, and therefore the decryption algorithm is always correct.

Proof of security. The following theorem is obtained from Theorem 2 and the fact that the above hash proof systems $\mathbf{HPS}_1$ and $\mathbf{HPS}_2$ are 1-universal λ-key-leakage extractor and extended 2-universal, respectively. The following theorem is shown and proven in the paper [12].

Theorem 4 ([12]): Assuming the hardness of the DDH problem, the above encryption scheme is semantically- secure against a-posteriori chosen-ciphertext $(\ell/6 - \omega(\log n) - m)$key-leakage attacks, where n denotes the security parameter, $\ell = \ell(n)$ denotes the length of the secret key and $m = m(n)$ denotes the length of the plaintext.

6. Conclusion

We have introduced the generic constructions of both stateless and stateful PKE and proved that they are LR-CCA2-secure. In these constructions, we have used the combination of any 1-universal HPS that satisfies the condition of a key-leakage extractor and any 2-universal HPS with some condition on the length of proof. In the case of StPE, we have also used IND-CCA-secure symmetric encryption.

We leave it as an open problem to identify other generic cryptography primitives (other than HPS) that are sufficient for constructing PKE schemes that are resilient to key-leakage. It is also an interesting open problem of constructing a StPE scheme secure against both state-leakage and key-leakage attacks. We might discuss the security with state-leakage in a similar way as that with randomness-leakage in our framework.

Acknowledgements

We would like to thank the anonymous reviewers for their valuable comments. This research was supported in part by a grant of I-System Co. Ltd., NTT Information Sharing Platform Laboratories, and Ministry of Education, Culture, Sports, Science and Technology.

References

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan, "Simultaneous hardcore bits and cryptography against memory attacks," Reingold, O. (ed.) TCC 2009, LNCS, vol.5444, pp 474–495, Springer, Heidelberg, 2009.

[2] J. Baek, J. Zhou, and F. Bao, "Generic construction of stateful key encryption and their applications," S.M. Bellovin, R. Gennaro, A.D. Keromytis, M. Yung, eds., ACNS 2008, LNCS, vol.5037, pp.75–93, Springer, Heidelberg, 2008.

[3] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," N. Koblitz, ed., CRYPTO 1996, LNCS, vol.1109, pp.1–15. Springer, Heidelberg, 1996.

[4] M. Bellare, T. Kohno, and V. Shoup, "Stateful public-key cryptosystems: How to encryption with one 160-bit exponentiaton," ACM CCS 2006, pp.380–389, ACM Press, 2006.

[5] M. Bellare and C. Namprepre, "Authenticated encryption: Relations among notions and analysis of the generic composition paradigm," T. Okamoto, ed., ASIACRYPT 2000, LNCS, vol.1976, pp.531–545, Springer, Heidelberg, 2000.

[6] R. Cramer and V. Shoup, "Universal Hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption," L.R. Knudsen, ed., EUROCRYPT 2002, vol.2332, pp.45–64, Springer, Heidelberg, 2002.

[7] R. Cramer and V. Shoup, "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack," SIAM J. Comput., vol.33, no.1, pp.167–226, 2003.

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

[9] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs, "Efficient public-key cryptography in the presence of key leakage," M. Abe, ed., ASIACRYPT 2010, LCNS, vol.6477, pp.613–631, Springer, Heidelberg, 2010.

[10] G. Gaubatz, J.-P. Kaps, and B. Sunar, "Public key cryptography in sensor networks revisited," 1st European Workshop on Security in Ad-Hoc and Sensor Networks (ESAS 04), 2004.

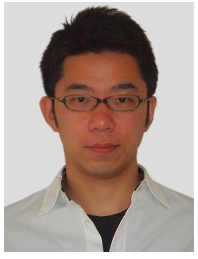
[11] E. Kiltz, K. Pietrzak, M. Stam, and M. Yung, "A new randomness extraction paradigm for hybrid encryption," A. Joux, ed., EUROCRYPT 2009, LNCS, vol.5479, pp.590–609, Springer, Heidelberg, 2009.

[12] M. Naor and V. Segev, "Public-key cryptosystems resilient to key leakage," Cryptology ePrint Archive: Report 2009/105 (2009), http://eprint.iacr.org/2009/105

[13] T. Phan, L. Huang, and C. Dulan, "Challenge: Integrating mobile wireless devices into the computational grid," MobiCom 2002, pp.271–278, ACM Press, 2002.

[14] J. Rompel, "One-way functions are necessary and sufficient for secure signatures," Proc. 22nd Annual ACM Symposium on Theory of Computing, pp.387–394, 1990.

Manh Ha Nguyen received his B.E. degree in computer science from National Defense Academy of Japan in 2006, and his M.S. degree from Tokyo Institute of Technology in 2011. He is currently a doctor course student in the Department of Mathematical and Computing Sciences, Tokyo Institute of Technology.



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. From 2008 to 2011, he was an assistant professor at Tokyo Institute of Technology. He is currently a researcher at Institute of Systems, Information Technologies and Nanotechnologies. His research interests are in coding theory, cryptography, and computational complexity.

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.