# Public-Key Encryption Resilient Against Linear Related-Key Attacks Revisited

Hui Cui, Yi Mu, and Man Ho Au

School of Computer Science and Software Engineering,
University of Wollongong, Wollongong, NSW 2522, Australia,
hc892@uowmail.edu.au,ymu@uow.edu.au,aau@uow.edu.au

**Abstract:** Wee (PKC'12) proposed a generic public-key encryption scheme in the setting of related-key attacks. Bellare, Paterson and Thomson (Asiacrypt'12) provided a framework enabling related-key attack (RKA) secure cryptographic primitives for a class of non-linear related-key derivation functions. However, in both of their constructions, the instantiations to achieve the full (strong) RKA security are given under the scenario regarding the private key composed of single element. In other words, each element of the private key shares the same modification. However, this is impractical in real world. In this paper, we concentrate on the security of public-key encryption schemes under linear related-key attacks in the setting of multi-element private keys (that is, the private key is composed of more than one element), where an adversary is allowed to tamper any part of this private key stored in a hardware device, and subsequently observe the outcome of a public-key encryption system under this targeted modified private key. We define the security model for RKA secure public-key encryption schemes as chosen-ciphertext and related-key attack (CC-RKA) security, which means that a public-key encryption scheme remains secure even when an adversary is allowed to issue the decryption oracle on linear shifts of any component of the private key. After that, we present a detailed public-key encryption schemes with the private key formed of several elements, of which the CC-RKA security is under the decisional BDH assumption in the standard model.

**Keywords:** Public-key encryption, Linear related-key attack, CC-RKA security.

## 1 Introduction

The security of cryptographic algorithms are analyzed in the black-box model, where an adversary may view the algorithm's inputs and outputs, but the private key as well as all the internal computations remain perfectly hidden. Unfortunately, this assumption does not reflect the real world, where a private key could be frequently compromised for various reasons. In this case, the adversary might get some partial information about private keys through some methods, which are not anticipated by the designer of the system and, correspondingly, not taken into account when arguing its security. Such attacks, referred to as key-leakage

attacks, come in a large variety. An important example is side-channel [25] attacks that exploit information leakage from the implementation of an algorithm, where an adversary observes some "physical output" of a computation (such as radiation, power, temperature, and running time), in addition to the "logical output" of the computation.

In modern cryptography, this requirement has been relaxed to capture security under the scenarios where some information of the keys is leaked to the adversary. When an adversary tampers the private key stored in a cryptographic hardware device and observes the result of the cryptographic primitive under this modified private key, there is a related-key attack (RKA) [4, 15]. The key here could be a signing key of a certificate authority or a decryption key of an encryption system. In related-key attacks, the adversary attempts to break an encryption scheme (or a signature scheme) by invoking it with several private keys (or signing keys) satisfying some known relations. So far, the RKA security has been done on a variety of cryptographic primitives such as identity-based encryption, public-key encryption, symmetric encryption, signature [4, 6]. Applebaum, Harnik and Ishai [2] put forward symmetric encryption schemes secure against linear related-key attacks. Wee [27] presented the public-key encryption schemes resilient against linear related-key attacks under standard assumptions in the standard model. Bellare, Paterson and Thomson [6] provided a framework enabling the RKA secure cryptographic primitives for sets of related-key derivation functions that are non-linear. [13] achieved the RKA security from the Cramer-Shoup public-key encryption scheme which fails to achieve it. We note that in [4], the strong, adaptive versions of the definitions on related-key attacks were introduced, but all the above works of the instantiations for the full RKA security (in [27], it refers to strong RKA security) are considered under a scenario where the private key shares the same modification on every component (this includes the case of single-element private key).

Motivated by the above, we ask besides the generic method proposed in [6], whether cryptographic primitives fully resilient against related-key attacks can be built beyond the single-element key setting, meaning that the private key is composed of several components as $sk = (sk_1, \ldots, sk_n)$ for $n \in Z^+$, and the modification to different elements of the private key could be different.

**Our contributions**. In this paper, we focus on the chosen-ciphertext attack (CCA) security of public-key encryption schemes with the private keys of multiple components under related-key attacks from a different point of view, avoiding the usage of the generic methodology to achieve the RKA secure cryptographic primitives proposed in [6] and the fingerprinting property to achieve the RKA secure public-key encryption suggested in [27].

To begin with, we briefly describe the framework introduced in [4]. Informally, a public-key encryption scheme is secure under related-key attacks if it is secure against chosen-ciphertext attacks even when the adversary obtains partial information of the message in the public-key encryption scheme under the modified private keys of the adversary. This is modeled by providing the adversary

with access to a related-key attack decryption oracle: the adversary can query the oracle with any function $(\phi, C)$, where $\phi$ is the relate-key deriving function and $C$ is ciphertext, and then receive the decryption of $C$ under $\phi(sk)$, where $sk$ is the private key. Note that the related-key deriving functions can be chosen depending on the public key, which is known to the adversary. The adversary can query the related-key attack oracle adaptively, with only one restriction that the decryption of a ciphertext $C$ with the private key $\phi(sk)$ cannot equal the decryption of the challenge ciphertext $C^*$ with the original private key $sk$.

In [27], the constructions exploit some existing public-key encryption schemes that are susceptible to linear related-key attacks to obtain public-key encryption schemes that are secure against linear related-key attacks from adaptive trapdoor relations via strong one-time signatures, which generate a *tag* in the ciphertext of the concrete scheme. The security of this realization is analogous to those for obtaining chosen-ciphertext attack (CCA) security from extractable hash proofs [26], and trapdoor functions [21]. Following the technical methods in [27], we work with a tag-based notion of CCA security [20], where the tag is derived using a strong one-time signature scheme and a signature is added to the ciphertext. In this case, if the ciphertext $C$ queried by the adversary shared the same *tag* as the challenge ciphertext $C^*$, then because of the security of one-time signature, $C$ should equal $C^*$. On the other hand, if $C$ has a different *tag* from $C^*$, then $C$ can be decrypted using the deriving private key $\phi(sk)$, where $\phi$ denoted a linear shift. However, to achieve the full chosen-ciphertext and related-key attack (CC-RKA) security, we need to address another problem where $C = C^*$ but $\phi(sk) \neq sk$. Here, in our public-key encryption schemes, we guarantee that the adversary procures nothing about the plaintext $M^*$ of $C^*$ from the decryption of $C^*$ under any $\phi(sk) \neq sk$ because of the additive randomness hidden in the response rather than the $\Phi$-fingerprinting suggested by [27].

At this point, it is sufficient for us to present a specific construction of CC-RKA secure public-key encryption systems under the setting of multi-component private keys. Our scheme is from bilinear pairings based on the identity-based encryption scheme in [9], of which the CC-RKA security can be proved under the decisional BDH assumption.

### 1.1 Related Work

In 2004, Micali and Reyzin [24] put forward a comprehensive framework for modeling security against side-channel attacks, which relies on the assumption that there is no leakage of information in the absence of computation. Later, Halderman et al. [19] described a set of attacks violating the assumption of the framework of Micali and Reyzin [24]. Specially speaking, their "cold boot" attacks showed that a significant fraction of the bits of a cryptographic key can be recovered if the key is ever stored in memory, of which the framework was modeled by Akavia, Goldwasser and Vaikuntanathan [1]. Similarly, fault injection techniques can be used to falsify, inducing the internal state of the devices being modified, if given physical access to the hardware devices [8]. Bellare and Kohno [5] investigated related-key attacks from a theoretical point of view and

presented an approach to formally handle the notion of related-key attacks. Following the approach in [5], Lucks [22] presented some constructions for block ciphers and pseudorandom function generators. To solve the open problem in related-secret security whether or not related-key secure blockciphers exist, Bellare and Cash [3] provided the first constructions to create related-secret pseudorandom bits. Based on the work in [3], Applebaum, Harnik, and Ishai [2] gave RKA secure symmetric encryption schemes, which can be used in garbled circuits in secure computation. Later, Bellare, Cash and Miller [4] proposed approaches to build high-level primitives secure against related-key attacks like signatures, CCA secure public-key encryption, identity-based encryption, based on RKA secure pseudorandom functions. Other work about cryptographic systems with RKA security includes signatures [6, 17], CCA secure public-key encryption [6, 27], identity-based encryption [6].

The remainder of this paper is organized as follows. In Section 2, we briefly review the properties of bilinear maps and the complexity assumptions, and the concepts associated to this work. In Section 3, we elaborate the security model of RKA secure public-key encryption schemes. In Section 4, we point out a simple related-key attack on an existing scheme. In Section 5, we present a public-key encryption scheme with RKA security, and analyze its CC-RKA security based on the decisional BDH assumption. In Section 6, we talk about RKA security in a further step under the random oracle model, where the related-key deriving function could be non-linear such as affine, polynomial. Finally, we conclude this paper in Section 7.

## 2 Preliminaries

In this section, we review some basic notions, definitions, and tools that are used in our constructions. We formally state the complexity assumptions, and the technical definitions that will be used repeatedly in our analysis.

### 2.1 Complexity Assumptions

Suppose that Groupgen is a probabilistic polynomial-time algorithm that takes input a security parameter $1^\lambda$ and outputs a triplet $(G, G_T, p, g, \hat{e})$ where $G$, $G_T$ are two groups of order $p$, $g$ is generator of $G$, $p$ is a $\lambda$-bit prime number, and $\hat{e} : G \times G \to G_T$ is a bilinear map with the following properties [12]:

- Bilinear: for all $g \in G$ and $a, b \in Z_p^*$, we have $\hat{e}(g^a, g^b) = \hat{e}(g, g)^{ab}$.
- Non-degenerate: $\hat{e}(g, g) \neq 1$.

**The decisional Bilinear Diffie-Hellman (BDH) assumption.** The decisional Bilinear Diffie-Hellman assumption is that the ensembles $\{G, g, g^{x_1}, g^{x_2}, g^{x_3}, \hat{e}(g, g)^{x_1 x_2 x_3}\}$ and $\{G, g, g^{x_1}, g^{x_2}, g^{x_3}, Z\}$ are computationally indistinguishable, where $(G, G_T, p, g, \hat{e}) \leftarrow \text{Groupgen}(1^\lambda)$, and the elements $g \in G$, $Z \in G_T$, $x_1, x_2, x_3 \in Z_p^*$ are chosen independently and uniformly at random.

### 2.2 Public-Key Encryption

A public-key encryption scheme is composed of the following three randomized algorithms [16]: Keygen, Encrypt, and Decrypt.

- $\text{Keygen}(1^\lambda) \to (sk, pk)$: Taking a security parameter $\lambda$ as input, this algorithm outputs a private key and a public key pair $(sk, pk)$.
- $\text{Encrypt}_{pk}(m) \to C$: Taking a plaintext $m$ (in some implicit message space), and a public key $pk$ as input, this algorithm outputs a ciphertext $C$.
- $\text{Decrypt}_{sk}(C) \to m$: Taking a ciphertext $C$, and a private key $sk$ as input, this algorithm outputs $m$ for a valid ciphertext or $\perp$ for an invalid ciphertext.

We require that a public-key encryption system is correct, meaning that if $(sk, pk) \leftarrow \text{Keygen}(1^\lambda)$, and $C \leftarrow \text{Encrypt}_{pk}(m)$, then $\text{Decrypt}_{sk}(C) = m$.

### 2.3 One-Time Signature

A signature scheme is composed of three randomized algorithms (Gen, Sign, Verify) [7] as follows.

- $(SK, VK) \leftarrow \text{Gen}(1^\lambda)$: Taking a security parameter $\lambda$ as input, this algorithm outputs a signing key and a verification key pair $(SK, VK)$.
- $\sigma \leftarrow \text{Sign}_{SK}(m)$: Taking a signing key $SK$ and a message $m$ (in some implicit message space) as input, this algorithm outputs a signatures $\sigma$.
- $\text{Verify}_{VK}(m, \sigma) = 1/0$: Taking a verification key $VK$ and a signature $\sigma$ as input, this algorithm outputs 1 for "acceptance" or 0 for "rejection".

We require that for all $(SK, VK)$ output by Gen, all $m$ in the message space, and all $\sigma$ output by $\text{Sign}_{SK}(m)$, we have $\text{Verify}_{VK}(m, \sigma) = 1$.

A signature scheme is a strong one-time signature (OTS) scheme [11] if for any probabilistic polynomial-time adversary $\mathcal{A}$ such that the quantity

$$\Pr\left[\begin{array}{l|l} \text{Verify}_{VK}(m', \sigma') = 1 & (SK, Vk) \leftarrow \text{Gen}(1^\lambda) \\ (m', \sigma') \neq (m, \sigma) & m \leftarrow \mathcal{A}(VK) \\ & \sigma \leftarrow \text{Sign}_{SK}(m) \\ & (m', \sigma') \leftarrow \mathcal{A}(\sigma) \end{array}\right]$$

is negligible in $\lambda$.

*A Strong One-Time Signature Scheme from DDH.* We present the one-time signature scheme in [18, 27], which is secure under the difficulty of computing discrete log and $H$ is collision resistant function.

- Key generation. Choose a random $g \in G$, random $a$, $b$, $c \in Z_p^*$, and set $u_1 = g^a$, $u_2 = g^b$, $u_3 = g^c$. Also, choose a collision resistant hash function $H : G \to Z_p^*$. The verification key is $VK = (u_1, u_2, u_3)$, and the signing key is $SK = (a, b, c)$.
- Sign. To sign a message $M \in G$,
  1. choose a random $e \in Z_p^*$, and compute $w = c + e \cdot a + (H(M) + e) \cdot b$.
  2. output signature $\sigma = (e, w)$.
- Verify. To verify signature $\sigma = (e, w)$, check $g^w = u_3 {u_1}^e {u_2}^{H(M)+e}$. If the equation holds, output 1; otherwise, output 0.

## 3 Modeling Related-Key Attacks

In this section, we define the notion of security under chosen-ciphertext attacks; in addition, we present an extension of this notion to the setting of related-key attacks, as introduced by Bellare, Cash and Miller [4].

### 3.1 Chosen-Ciphertext Attacks

A public-key encryption scheme (Keygen, Encrypt, Decrypt) is secure against chosen-ciphertext attacks (CCA security) if for a stateful adversary algorithm $\mathcal{A}$, it holds that

$$\Pr\left[d = d' \;\middle|\; \begin{array}{l} (sk, pk) \leftarrow \text{Keygen}(1^\lambda). \\ (m_0, m_1) \leftarrow \mathcal{A}^{\text{Decrypt}_{sk}(\cdot)}(pk), |m_0| = |m_1|. \\ d \in \{0,1\}. \\ C^* \leftarrow \text{Encrypt}_{pk}(m_d). \\ d' \leftarrow \mathcal{A}^{\text{Decrypt}_{sk}(\cdot)}(C^*). \end{array}\right] - 1/2$$

is negligible in the security parameter $\lambda$, where $\text{Decrypt}_{sk}(\cdot)$ is an oracle that on an input $C$, it returns $\text{Decrypt}_{sk}(C)$.

The weaker security notion of security against chosen-plaintext attacks (CPA security) is obtained in the above security game when depriving adversary $\mathcal{A}$ of the the access to the decryption oracle.

### 3.2 RKA Security

**Related-key deriving functions.** Our definition follows the notion of related-key deriving functions given in [5]. Briefly speaking, a class $\Phi$ of related-key deriving functions $\phi$: $sk \rightarrow sk$ is a finite set of functions with the same domain and range, which map a key to a related key. Additionally, $\Phi$ should allow an efficient membership test, and $\phi$ should be efficiently computable. Note that in our concrete constructions, we only consider the class $\Phi^+$ as linear shifts.

*The family* $\Phi^+$. Any function $\phi : Z_p^* \rightarrow Z_p^*$ in this class is indexed by $\triangle \in Z_p^*$, where $\phi_\triangle(sk) = sk + \triangle$. Note that if $sk$ is composed of several elements as ($sk_1$, ..., $sk_n$) with $n \in Z^+$, for any $sk_i$ where $i \in \{1, \ldots, n\}$, $\triangle_i \in Z_p^*$.

**CC-RKA Security.** A public-key encryption scheme (Keygen, Encrypt, Decrypt) is $\Phi$-CC-RKA secure if for a stateful adversary algorithm $\mathcal{A}$, it holds that

$$\Pr\left[d = d' \;\middle|\; \begin{array}{l} (sk, pk) \leftarrow \text{Keygen}(1^\lambda). \\ (m_0, m_1) \leftarrow \mathcal{A}^{\text{RKA.Decrypt}_{sk}(\cdot,\cdot)}(pk), |m_0| = |m_1|. \\ d \in \{0,1\}. \\ C^* \leftarrow \text{Encrypt}_{pk}(m_d). \\ d' \leftarrow \mathcal{A}^{\text{RKA.Decrypt}_{sk}(\cdot,\cdot)}(C^*). \end{array}\right] - 1/2$$

is negligible in the security parameter $\lambda$, where $\text{RKA.Decrypt}_{sk}(\cdot, \cdot)$ is an oracle that on an input ($\phi$, $C$), it returns $\text{Decrypt}_{\phi(sk)}(C)$. We constraint algorithm $\mathcal{A}$ to only make queries ($\phi$, $C$) such that $\phi \in \Phi$ and ($\phi(sk)$, $C$) $\neq$ ($sk$, $C^*$).

## 4 RKA Attacks on An Existing Scheme

The algorithms of the CCA secure public-key encryption scheme in [11] are given as follows, which is claimed to be CCA secure under the related-key attacks where each component of the private key shares the same linear related-key deriving function in [4]. Let (Mac, Vrfy) denote the message authentication code [11]. Let $H : \{0,1\}^* \to \{0,1\}^*$ represent a hash function to be second-preimage resistant [11]. Let $G$ be a group of order $p$, and let $\hat{e} : G \times G \to G_T$ be a bilinear group.

- Key generation. To generate the public key and secret key of the system,
  1. choose random $g \in G$, $x_1$, $x_2$, $y \in Z_p^*$, and set $g_1 = g^{x_1}$, $g_2 = g^{x_2}$, $g_3 = g^y$, $\Omega = \hat{e}(g_1, g_3)$.
  2. choose a hash function $h$ from a family of pairwise-independent hash functions [11].

  The public key is $PK = (g,\ h,\ g_1,\ g_2,\ g_3,\ \Omega)$, and the secret key is $SK = (x_1,\ x_2,\ y)$.
- Encryption. To encrypt a message $M \in G_T$,
  1. choose a random $r \in \{0,1\}^*$, and set $k_1 = h(r)$, $ID = H(r)$.
  2. choose a random $s \in Z_p^*$, and set $C = (g^s,\ {g_2}^s {g_3}^{s \cdot ID},\ \Omega^s \oplus (M \circ r))$.
  3. output ciphertext $(ID,\ C,\ \mathrm{Mac}_{k_1}(C))$.
- Decryption. To decrypt a ciphertext $(ID,\ C,\ \mathrm{Mac}_{k_1}(C))$,
  1. parse $C$ as $(C_1,\ C_2,\ C_3)$, choose a random $t \in Z_p^*$, and computes $(M \circ r) = C_3 \oplus \hat{e}({C_1}^{x_1 \cdot y + t \cdot (x_2 + y \cdot ID)} {C_2}^{-t}, g)$.
  2. set $k_1 = h(r)$. If $\mathrm{Vrfy}_{k_1}(C, tag) = 1$ and $H(r) = ID$, output $M$; otherwise, output $\perp$.

**The attack**. We point out a related-key attack on the above scheme where each element of the private key has its own linear related-key deriving function. Note that this does not contradict to the result in [11], since we consider a different related-key deriving function from theirs. Suppose we are given a valid ciphertext $(ID,\ C,\ \mathrm{Mac}_{k_1}(C))$ of some message $M$. We can recover $M$ by making decryption queries to RKA.Decrypt oracle on a related secret key via the following attack. For any $\triangle_i \in Z_p^*$, $i \in \{1, 2, 3\}$, we change the secret key $(x_1,\ x_2,\ y)$ to $(x_1,\ x_2 + \triangle_2,\ y + \triangle_3)$ where $\triangle_2 = -ID \cdot \triangle_3$, then $(ID,\ C,\ \mathrm{Mac}_{k_1}(C))$ can be decrypted to $M$ directly.

## 5 An Instantiation from Bilinear Pairings

In this section, we propose a public-key encryption scheme in the setting of related-key attacks from bilinear pairings, and reduce its CC-RKA security based on the decisional BDH assumption.

### 5.1 The Construction

Let $G$ be a group of order $p$, and let $\hat{e} : G \times G \to G_T$ be a bilinear group. Based on the selective-identity secure identity-based encryption scheme in [9], we construct a CC-RKA secure public-key encrypiton scheme as follows.

- Key generation. To generate the public key and secret key of the system,
  1. choose random $g$, $h \in G$, $x_1$, $x_2$, $y \in Z_p^*$, and set $g_1 = g^{x_1}$, $g_2 = g^{x_2}$, $h_1 = h^{y_1}$, $h_2 = h^{y_2}$, $\Omega = \hat{e}(g, g)^{x_1 \cdot x_2}$.
  2. choose two collision resistant hash functions $H_1 : G^3 \to Z_p^*$, $H_2 : G^3 \times G_T \to Z_p^*$.

  The public key is $PK = (g, h, g_1, g_2, h_1, h_2, \Omega, H_1, H_2)$, and the secret key is $SK = (x_1, x_2, y_1, y_2)$.
- Encryption. To encrypt a message $M \in G_T$,
  1. choose random $a$, $b$, $c \in Z_p^*$ independently, and set $u_1 = g^a$, $u_2 = g^b$, $u_3 = g^c$.
  2. choose a random $r \in Z_p^*$, compute $t = H_1(u_1, u_2, u_3)$, and set $C = (C_1, C_2, C_3, C_4, C_5)$, where
  $$C_1 = g^r, \quad C_2 = h^r, \quad C_3 = {h_1}^r {g_2}^{r \cdot t}, \quad C_4 = {h_2}^r {g_1}^{r \cdot t}, \quad C_5 = \Omega^r \cdot M.$$
  3. choose a random $e \in Z_p^*$, and compute $\sigma = c + e \cdot a + (H_2(C) + e) \cdot b$.
  4. output ciphertext $(u, e, \sigma, C)$, where $u = (u_1, u_2, u_3)$.
- Decryption. To decrypt a ciphertext $(u, e, \sigma, C)$,
  1. parse $u$ as $(u_1, u_2, u_3)$, and compute $t = H_1(u_1, u_2, u_3)$.
  2. parse $C$ as $(C_1, C_2, C_3, C_4, C_5)$, and output $\perp$ if $g^\sigma \neq u_3 {u_1}^e {u_2}^{H_2(C)+e}$.
  3. if $\hat{e}(C_3, g) = \hat{e}(C_1, h_1) \cdot \hat{e}(C_1, g_2)^t$, $\hat{e}(C_3, h) = \hat{e}(C_2, h_1) \cdot \hat{e}(C_2, g_2)^t$, and $\hat{e}(C_4, g) = \hat{e}(C_1, h_2) \cdot \hat{e}(C_1, g_1)^t$, $\hat{e}(C_4, h) = \hat{e}(C_2, h_2) \cdot \hat{e}(C_2, g_1)^t$, choose random $s_1$, $s_2 \in Z_p^*$, and compute
  $$M = \frac{C_5}{\hat{e}({C_1}^{x_1 \cdot x_2 + s_1 \cdot x_2 \cdot t + s_2 \cdot x_1 \cdot t} {C_2}^{s_1 \cdot y_1 + s_2 \cdot y_2} {C_3}^{-s_1} {C_4}^{-s_2}, g)}.$$
  Note that the pairing computation of $\hat{e}(C_3, g) = \hat{e}(C_1, h_1) \cdot \hat{e}(C_1, g_2)^t$, $\hat{e}(C_3, h) = \hat{e}(C_2, h_1) \cdot \hat{e}(C_2, g_2)^t$, and $\hat{e}(C_4, g) = \hat{e}(C_1, h_2) \cdot \hat{e}(C_1, g_1)^t$, $\hat{e}(C_4, h) = \hat{e}(C_2, h_2) \cdot \hat{e}(C_2, g_1)^t$ to check the correctness of the ciphertext can be done by a third party [23], and we do not count them in the computation cost.

**Correctness.** For any sequence of the key generation and encryption algorithms, it holds that

$$\begin{aligned} M &= \frac{C_5}{\hat{e}({C_1}^{x_1 \cdot x_2 + s_1 \cdot x_2 \cdot t + s_2 \cdot x_1 \cdot t} {C_2}^{s_1 \cdot y_1 + s_2 \cdot y_2} {C_3}^{-s_1} {C_4}^{-s_2}, g)} \\ &= \frac{\Omega^r \cdot M}{\hat{e}((g^r)^{x_1 \cdot x_2 + s_1 \cdot x_2 \cdot t + s_2 \cdot x_1 \cdot t} (h^r)^{s_1 \cdot y_1 + s_2 \cdot y_2} ({h_1}^r {g_2}^{r \cdot t})^{-s_1} ({h_2}^r {g_1}^{r \cdot t})^{-s_2}, g)} \\ &= \frac{\Omega^r \cdot M}{\hat{e}(g^{x_1 \cdot x_2 \cdot r}, g)}, \end{aligned}$$

and therefore the decryption algorithm is always correct.

### 5.2 Security

**Theorem 1.** *The above public-key encryption scheme is secure in the CC-RKA security game regarding linear related-key deriving function $\phi^+$ under the decisional BDH assumption.*

**Proof.** The intuition of the proof is as follows. We show that given an algorithm $\mathcal{A}$ that breaks the security of the CC-RKA secure public-key encryption scheme, we can build an algorithm $\mathcal{B}$ that solves the decisional BDH problem, which is given a random tuple $(g,\ g^{x_1},\ g^{x_2},\ g^{x_3},\ Z)$ as input. Algorithm $\mathcal{B}$'s goal is to determine whether $Z = \hat{e}(g^{x_1}, g^{x_2})^{x_3}$ or $Z$ is a random element in $G_T$.

**Setup.** Algorithm $\mathcal{B}$ chooses a random $\gamma \in Z_p^*$ such that $h = g^\gamma$, two hash functions $H_1 : G^3 \to Z_p^*$, $H_2 : G^4 \to Z_p^*$, and sets $g_1 = g^{x_1}$, $g_2 = g^{x_2}$, and $\Omega = \hat{e}(g_1, g_2)$. Also, algorithm $\mathcal{B}$ chooses random $a,\ b,\ c \in Z_p^*$, and sets $t^* = H_1(u_1^*, u_2^*, u_3^*)$, where $u_1^* = g^a$, $u_2^* = g^b$, $u_3^* = g^c$. Then algorithm $\mathcal{B}$ chooses random $y_1,\ y_2 \in Z_p^*$, and sets $h_1 = {g_2}^{-t^*} h^{y_1}$, $h_2 = {g_1}^{-t^*} h^{y_2}$. Algorithm $\mathcal{B}$ sends the public key $PK = (g,\ h,\ g_1,\ g_2,\ h_1,\ h_2,\ \Omega,\ H_1,\ H_2)$ to algorithm $\mathcal{A}$.

**Phase 1.** Algorithm $\mathcal{A}$ queries $(\phi,\ (u,\ e,\ \sigma,\ C))$ to RKA.Decrypt oracle. Algorithm $\mathcal{B}$ proceeds as follows.

- Parses $C$ as $(C_1,\ C_2,\ C_3,\ C_4,\ C_5)$. If $\hat{e}(C_3, g) \neq \hat{e}(C_2, h_1) \cdot \hat{e}(C_1, g_1)^t$, or $\hat{e}(C_4, g) \neq \hat{e}(C_2, h_2) \cdot \hat{e}(C_1, g_2)^t$, algorithm $\mathcal{B}$ outputs $\perp$.
- Parses $u$ as $(u_1,\ u_2,\ u_3)$, and checks the signature $(e,\ \sigma)$ on $C$. If $g^\sigma \neq {u_3 {u_1}^e {u_2}^{H_2(C)+e}}$, algorithm $\mathcal{B}$ outputs $\perp$.
- Computes $t = H_1(u_1, u_2, u_3)$. If $t = t^*$, algorithm $\mathcal{B}$ aborts the simulation. Otherwise, algorithm $\mathcal{B}$ runs in the following steps.
  1. Chooses random $s_1,\ s_2 \in Z_p^*$, and sets $d_1 = {g_1}^{\frac{-\gamma \cdot y_1}{t-t^*}} ({g_2}^t h_1)^{s_1}$, $d_2 = {g_2}^{\frac{-\gamma \cdot y_2}{t-t^*}} ({g_1}^t h_2)^{s_2}$.
     Let $s_1' = s_1 - x_1/(t-t^*)$, $s_2' = s_2 - x_2/(t-t^*)$, we have that

$$d_1 = {g_1}^{\frac{-\gamma \cdot y_1}{t-t^*}} ({g_2}^t h_1)^{s_1} = {g_1}^{\frac{-\gamma \cdot y_1}{t-t^*}} ({g_2}^t h_1)^{s_1' + \frac{x_1}{t-t^*}} = g^{x_1 \cdot x_2} ({g_2}^t h_1)^{s_1'},$$
$$d_2 = {g_2}^{\frac{-\gamma \cdot y_2}{t-t^*}} ({g_1}^t h_2)^{s_2} = {g_2}^{\frac{-\gamma \cdot y_2}{t-t^*}} ({g_1}^t h_2)^{s_2' + \frac{x_2}{t-t^*}} = g^{x_1 \cdot x_2} ({g_1}^t h_2)^{s_2'}.$$

  2. Computes $M$ as $M = C_5 / [\hat{e}(d_1, C_1) \cdot \hat{e}({C_3}^{-s_1'}, g) \cdot \hat{e}(d_2, C_1) \cdot \hat{e}({C_4}^{-s_2'}, g)]^{\frac{1}{2}}$.
     To see this, we rewrite

$$\begin{aligned} M &= \frac{C_5}{[\hat{e}(d_1, C_1) \cdot \hat{e}({C_3}^{-s_1'}, g) \cdot \hat{e}(d_2, C_1) \cdot \hat{e}({C_4}^{-s_2'}, g)]^{1/2}} = \frac{C_5}{\hat{e}(g^{x_1}, g^{x_2})^r} \\ &= \frac{C_5}{[\hat{e}(d_1, C_1) \cdot \hat{e}(C_3, g^{-s_1'}) \cdot \hat{e}(d_2, C_1) \cdot \hat{e}(C_4, g^{-s_2'})]^{1/2}} \\ &= \frac{C_5}{[\hat{e}(d_1, C_1) \cdot \hat{e}(C_3, (g^{x_1})^{\frac{1}{t-t^*}} \cdot g^{-s_1}) \cdot \hat{e}(d_2, C_1) \cdot \hat{e}(C_4, (g^{x_2})^{\frac{1}{t-t^*}} \cdot g^{-s_2}]^{\frac{1}{2}}}. \end{aligned}$$

3. Outputs $M'$ as

$$M' = \frac{M}{\hat{e}(C_1, {g_1}^{\triangle_{x_2}} {g_2}^{\triangle_{x_1}} g^{\triangle_{x_1} \cdot \triangle_{x_2}} g^{(s_1 \cdot t \cdot \triangle_{x_2} + s_2 \cdot t \cdot \triangle_{x_1}) + \gamma \cdot (s_1 \cdot \triangle_{y_1} + s_2 \cdot \triangle_{y_2})})}.$$

From the RKA decryption algorithm, we have

$$\begin{aligned} M' &= \frac{C_5}{\hat{e}({C_1}^{\phi(x_1) \cdot \phi(x_2) + s_1 \cdot \phi(x_2) \cdot t + s_2 \cdot \phi(x_1) \cdot t} {C_2}^{s_1 \cdot \phi(y_1) + s_2 \cdot \phi(y_2)} {C_3}^{-s_1} {C_4}^{-s_2}, g)} \\ &= \frac{M}{\hat{e}({C_1}^{x_1 \cdot \triangle_{x_2} + x_2 \cdot \triangle_{x_1} + \triangle_{x_1} \cdot \triangle_{x_2} + (s_1 \cdot t \cdot \triangle_{x_2} + s_2 \cdot t \cdot \triangle_{x_1})} {C_2}^{s_1 \cdot \triangle_{y_1} + s_2 \cdot \triangle_{y_2}}, g)} \\ &= \frac{M}{\hat{e}(C_1, {g_1}^{\triangle_{x_2}} {g_2}^{\triangle_{x_1}} g^{\triangle_{x_1} \cdot \triangle_{x_2}} g^{(s_1 \cdot t \cdot \triangle_{x_2} + s_2 \cdot t \cdot \triangle_{x_1}) + \gamma \cdot (s_1 \cdot \triangle_{y_1} + s_2 \cdot \triangle_{y_2})})}, \end{aligned}$$

where $\phi(x_1) = x_1 + \triangle_{x_1}$, $\phi(x_2) = x_2 + \triangle_{x_2}$, $\phi(y_1) = y_1 + \triangle_{y_1}$, and $\phi(y_2) = y_2 + \triangle_{y_2}$.
Thus, as long as the ciphertext from algorithm $\mathcal{A}$ is correctly formulated, algorithm $\mathcal{B}$'s response is always identical to the RKA decryption algorithm as required.

Note that algorithm $\mathcal{B}$ responds the RKA decryption queries without using the secret key $SK$.

**Challenge.** Algorithm $\mathcal{A}$ outputs two messages $M_0, M_1 \in G_T$ on which it wishes to be challenged. Algorithm $\mathcal{B}$ chooses a random $d \in \{0, 1\}$, and responds with the ciphertext $C^* = (C_1^*, C_2^*, C_3^*, C_4^*, C_5^*)$, where

$$C_1^* = g^{x_3}, \quad C_2^* = (g^{x_3})^{\gamma}, \quad C_3^* = (g^{x_3})^{\gamma \cdot y_1}, \quad C_4^* = (g^{x_3})^{\gamma \cdot y_2}, \quad C_5^* = Z \cdot M_d.$$

Hence, if $Z = \hat{e}(g, g)^{x_1 x_2 x_3} = \hat{e}(g_1, g_2)^{x_3}$. then $C^*$ is a valid encryption of $M_d$ with respect to $t^*$. On the other hand, when $Z$ is a random in $G_T$, then $C^*$ is independent of $d$ in algorithm $\mathcal{A}$'s view.

**Phase 2.** Algorithm $\mathcal{A}$ continues to adaptively issue queries $(\phi, (u, e, \sigma, C))$ to RKA.Decrypt oracle.

- If $\phi(SK) = SK$ and $C = C^*$, algorithm $\mathcal{B}$ responds with $\perp$.
- Otherwise, algorithm $\mathcal{B}$ responds as in Phase 1 except when $H_1(u_1, u_2, u_3) = t^*$, algorithm $\mathcal{B}$ outputs

$$M' = \frac{M_d}{\hat{e}(C_1, {g_1}^{\triangle_{x_2}} {g_2}^{\triangle_{x_1}} g^{\triangle_{x_1} \cdot \triangle_{x_2}} g^{(s_1 \cdot t^* \cdot \triangle_{x_2} + s_2 \cdot t^* \cdot \triangle_{x_1}) + \gamma \cdot (s_1 \cdot \triangle_{y_1} + s_2 \cdot \triangle_{y_2})})}.$$

Note that without the randomness chosen by algorithm $\mathcal{B}$, algorithm $\mathcal{A}$ has a negligible probability in outputting $M_d$. In other words, $M'$ is independent of $d$ from the view of algorithm $\mathcal{A}$.

**Output.** Algorithm $\mathcal{A}$ output a guess $d' \in \{0, 1\}$. If $d' = d$, algorithm $\mathcal{B}$ outputs 1; otherwise, algorithm $\mathcal{B}$ outputs 0.

Let abort be the event that algorithm $\mathcal{B}$ aborts during the simulation. To conclude the proof of Theorem 1, it remains to bound the probability that algorithm $\mathcal{B}$ aborts the simulation for one of algorithm $\mathcal{A}$'s RKA decryption queries. We claim that Pr[abort] is negligible, or one can use algorithm $\mathcal{A}$ to forge signatures with at least the same probability. Briefly, we can construct another algorithm $\mathcal{B}'$ to simulate the above security game that is given the private key, but is given the signing key as a challenge in an existential forgery game. Algorithm $\mathcal{A}$ causes an abort by querying a ciphertext including an existential forgery under the given signing key. Algorithm $\mathcal{B}'$ is able to use this forgery to win the existential forgery game. Note that during the game algorithm $\mathcal{A}$ makes only one chosen message query to generate the signature needed for the challenger ciphertext. Thus, Pr[abort] is negligible.

From the analysis of Phase 2, definitely, algorithm $\mathcal{A}$ has negligible probability in outputting $d' = d$ via the RKA decryption queries, as $M'$ is always consistent with two random $s_1$, $s_2 \in Z_p^*$ chosen by algorithm $\mathcal{B}$ such that $M'$ is independent of $d$ from the view of algorithm $\mathcal{A}$. Let $\epsilon$ be the advantage that algorithm $\mathcal{A}$ breaks the CC-RKA security of the above game. Therefore, we can see that if algorithm $\mathcal{B}$'s input tuple is $(g,\ g^a,\ g^b,\ g^c,\ Z)$ where $Z = \hat{e}(g,g)^{abc}$, then algorithm $\mathcal{A}$'s view of this simulation is identical to the real attack, thus algorithm $\mathcal{A}$'s probability in outputting $d' = d$ must satisfy $\Pr[d = d'] = 1/2 + \epsilon$. On the other hand, if algorithm $\mathcal{B}$'s input tuple is $(g,\ g^a,\ g^b,\ g^c,\ Z)$ where $Z \in G_T$, then algorithm $\mathcal{A}$'s advantage is nil, and algorithm $\mathcal{A}$'s view of the challenge ciphertext is independent of $d$, thus algorithm $\mathcal{A}$'s probability in outputting $d' = d$ is $\Pr[d' = d] = 1/2$. To sum up, algorithm $\mathcal{B}$'s probability in solving the decisional BDH problem is

$$\Pr[\mathcal{B}(g, g^a, g^b, g^c, Z)] = 1/2 \cdot (1/2 + \epsilon) + 1/2 \cdot 1/2 = 1/2 + \epsilon/2.$$

This completes the proof of Theorem 1.

### 5.3 Comparison

We compare the example of CC-RKA secure public-key encryption scheme referred to in [4], the strongly CC-RKA secure public-key encryption scheme from BDH in [27], the transformation from the Cramer-Shoup public-key encryption scheme to achieve CC-RKA security in [13]. and ours from BDH in Table 1.

In this table, "Pairing-E" means the sum of paring computation executed during the encryption phase, and "Pairing-D" means the sum of paring computation executed during the decryption phase. "Ex-E" means the the sum of exponentiation computation executed during the encryption phase, "Ex-D" means the the sum of exponentiation computation executed during the decryption phase.

## 6 Discussion

We only consider the linear related-key deriving function in the above public-key encryption scheme. Actually, the related-key deriving function could be

**Table 1.** Comparison between public-key encryption schemes with CC-RKA security

| Scheme | Ciphertext Size Size | Pairing-E | Pairing-D | Ex-E | Ex-D | Full (Strong) RKA Security | Muti-Element Private Key |
|---|---|---|---|---|---|---|---|
| [4] | 4 | 0 | 1 | 6 | 4 | No | Yes |
| [27] | 6 | 1 | 3 | 7 | 5 | Yes | No |
| [13] | 5 | 0 | 0 | 7 | 5 | No | Yes |
| Ours | 6 | 0 | 1 | 10 | 7 | Yes | Yes |

non-linear such as constant, affine, polynomial. To achieve the CC-RKA security beyond the linear barrier, we can simply change $C_4 = \Omega^r \cdot M$ to $C_4 = H(\Omega^r||{g_1}^r||{g_2}^r||{h_1}^r||{h_2}^r) \cdot M$, where $H : G_T \times G^4 \rightarrow G_T$ is a collision resistant hash function.

**Theorem 2.** *The above public-key encryption scheme is secure in the CC-RKA security game beyond linear related-key deriving function $\phi^+$ under the decisional BDH assumption in the random oracle model.*

**Proof.** The process of this proof roughly follows that of Theorem 1 except that we add a random oracle $H$ to the public parameter, so the responses to the RKA decryption queries will be different from those in Theorem 1.

At any time, algorithm $\mathcal{A}$ can query $Y||Y_1||Y_2||Y_3||Y_4$ to the random oracle $H$. Algorithm $\mathcal{B}$ keeps a list $L_H$ of tuples ($Y||Y_1||Y_2||Y_3||Y_4$, $H(Y||Y_1||Y_2||Y_3||Y_4)$) which is initially empty. For a query to the random oracle $H$ from algorithm $\mathcal{A}$,

- if $Y||Y_1||Y_2||Y_3||Y_4$ already appears in list $L_H$, algorithm $\mathcal{B}$ responds with $H(Z||Y_1||Y_2||Y_3||Y_4)$.
- if either $\hat{e}(Y_1, g) = \hat{e}(g_1, g^{x_3})$, $\hat{e}(Y_2, g) = \hat{e}(g_2, g^{x_3})$, or $\hat{e}(Y_3, g) = \hat{e}(h_1, g^{x_3})$, $\hat{e}(Y_4, g) = \hat{e}(h_2, g^{x_3})$, algorithm $\mathcal{B}$ solves the decisional BDH problem immediately via $\hat{e}(Y_1, g_2)$, $\hat{e}(Y_2, g_1)$, or

$$\begin{aligned}
\hat{e}(Y_3, g_2) &= \hat{e}((g_1{}^{-t^*} h^{y_2})^{x_3}, g_2) = \hat{e}({g_1}^{x_3}, g_2)^{-t^*} \cdot \hat{e}((g^{x_3})^{\gamma \cdot y_2}, g_2) \\
&\Rightarrow \hat{e}({g_1}^{x_3}, g_2) = (\frac{\hat{e}(Y_3, g_2)}{\hat{e}((g^{x_3})^{\gamma \cdot y_2}, g_2)})^{-\frac{1}{t^*}}, \\
\hat{e}(Y_4, g_1) &= \hat{e}((g_2{}^{-t^*} h^{y_1})^{x_3}, g_1) = \hat{e}({g_2}^{x_3}, g_1)^{-t^*} \cdot \hat{e}((g^{x_3})^{\gamma \cdot y_1}, g_1) \\
&\Rightarrow \hat{e}({g_2}^{x_3}, g_1) = (\frac{\hat{e}(Y_4, g_1)}{\hat{e}((g^{x_3})^{\gamma \cdot y_1}, g_1)})^{-\frac{1}{t^*}}.
\end{aligned}$$

- otherwise, algorithm $\mathcal{B}$ chooses a random $Z_i \in G_T$ to answer, and adds ($Y||Y_1||Y_2||Y_3||Y_4$, $Z_i$) to list $L_H$.

In Phase 1, when algorithm $\mathcal{A}$ queries ($\phi$, ($u$, $e$, $\sigma$, $C$)) to RKA.Decrypt oracle, where $\phi$ can be any kind of related-key deriving functions,

- algorithm $\mathcal{B}$ parses $C$ as ($C_1$, $C_2$, $C_3$, $C_4$, $C_5$), if either $\hat{e}(C_3, g) \neq \hat{e}(C_2, h_1) \cdot \hat{e}(C_1, g_1)^t$, or $\hat{e}(C_4, g) \neq \hat{e}(C_2, h_2) \cdot \hat{e}(C_1, g_2)^t$, algorithm $\mathcal{B}$ outputs $\perp$.
- algorithm $\mathcal{B}$ parses $u$ as ($u_1$, $u_2$, $u_2$), if $g^\sigma \neq u_3 {u_1}^e {u_2}^{H_2(C)+e}$, algorithm $\mathcal{B}$ outputs $\perp$.

- algorithm $\mathcal{B}$ computes $t = H_1(u_1, u_2, u_3)$, if $t = t^*$, algorithm $\mathcal{B}$ aborts the simulation.
- if either $\hat{e}(Y_1, g) = \hat{e}(g_1, C_1)$, $\hat{e}(Y_2, g) = \hat{e}(g_2, C_1)$, or $\hat{e}(Y_3, g) = \hat{e}(h_1, C_2)$, $\hat{e}(Y_4, g) = \hat{e}(h_2, C_2)$ algorithm $\mathcal{B}$ can answer with $M' = C_5/Z_i$ directly, where $Z_i$ is consistent to some existed $Y||Y_1||Y_2||Y_3||Y_4$ in list $L_H$.
- otherwise, algorithm $\mathcal{B}$ chooses a random $Z_i \in G_T$, compute $M' = C_5/Z_i$ to respond, and adds $(Y||Y_1||Y_2||Y_3||Y_4,\ Z_i)$ to list $L_H$.

In Phase 2, when algorithm $\mathcal{A}$ queries $(\phi,\ (u,\ e,\ \sigma,\ C))$ to RKA.Decrypt oracle with the restriction that $\phi(SK) \neq SK$, algorithm $\mathcal{B}$ responds as in Phase 1 except that in the case $t = t^*$, it chooses a random $Z_i \in G_T$, compute $M' = C_5^*/Z_i$ to respond, and adds $(Y||Y_1||Y_2||Y_3||Y_4,\ Z_i)$ to list $L_H$.

Note that $M'$ is independent of $d$, and algorithm $\mathcal{B}$'s response is similar to the RKA decryption algorithm as required without using the secret key $SK$.

For other parts of this proof, as it follows that in Theorem 1, we do not repeat them here. This completes the proof of Theorem 2.

*Analysis.* However, the above scheme is proved to be CC-RKA secure in the random oracle which assumes that all parties have access to an ideal hash function and is extremely useful for designing simple, efficient and highly practical solutions for many problems but from a theoretical view, a security proof in the random oracle model is only heuristic indication of the system's security. As a result, from a perspective of security, a proof in the standard model is preferable [14] to a proof in the random oracle model which does not assume idealized oracle access and proves security with only standard complexity assumptions.

## 7 Conclusions

Wee [27] presented the first public-key encryption schemes against linear related-key attacks under the standard assumptions in the standard model, and then Bellare, Paterson and Thomson [6] provided a framework enabling RKA secure cryptographic primitives for sets of related-key derivation functions that are non-linear, but the instantiations achieving the full (strong) RKA security given in both of them are in the context of regarding the (single-element or multi-element) private key as a whole where each part of the private key shares the same modification. Motivated by this, we focused on the construction of a fully CC-RKA secure public-key encryption scheme in the setting where the private key is made up of multiple components. Following the work in [27], in this paper, we made use of a tag-based notion of CCA security [20] to achieve the CC-RKA security, where the tag is derived using a strong one-time signature scheme. In this way, as well as the gadget of randomness, we reduced the CC-RKA security of the scheme to its CCA security with an addition property that any query $(\phi,\ C^*)$, where $\phi$ is a related-key deriving function, and $C^*$ is a challenge ciphertext, to RKA.Decrypt oracle, gives no information about the plaintext $M^*$ of $C^*$ to the adversary. Specifically, we obtained a public-key encryption scheme with multi-element private key secure against linear related-key attacks from bilinear

pairings, based on an efficient selective-identity secure identity-based encryption scheme without random oracles in [9], and proved its CC-RKA security under the decisional BDH assumption. We discussed the CC-RKA security beyond linear barrier where related-key deriving function could be non-linear such as affine, polynomial in the random oracle model.

## References


1. A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.
2. B. Applebaum, D. Harnik, and Y. Ishai. Semantic security under related-key attacks and applications. In *ICS*. Tsinghua University Press, 2011.
3. M. Bellare and D. Cash. Pseudorandom functions and permutations provably secure against related-key attacks. In *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 666–684. Springer, 2010.
4. M. Bellare, D. Cash, and R. Miller. Cryptography secure against related-key attacks and tampering. In *ASIACRYPT*, volume 7073 of *Lecture Notes in Computer Science*, pages 486–503. Springer, 2011.
5. M. Bellare and T. Kohno. A theoretical treatment of related-key attacks: Rka-prps, rka-prfs, and applications. In *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 491–506. Springer, 2003.
6. M. Bellare, K. G. Paterson, and S. Thomson. Rka security beyond the linear barrier: Ibe, encryption and signatures. In *ASIACRYPT*, volume 7658 of *Lecture Notes in Computer Science*, pages 331–348. Springer, 2012.
7. M. Bellare and P. Rogaway. Optimal asymmetric encryption. In *EUROCRYPT*, volume 950 of *Lecture Notes in Computer Science*, pages 92–111. Springer, 1994.
8. E. Biham. New types of cryptoanalytic attacks using related keys (extended abstract). In *EUROCRYPT*, volume 765 of *Lecture Notes in Computer Science*, pages 398–409. Springer, 1993.
9. D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.
10. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.
11. D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM J. Comput.*, 36(5):1301–1328, 2007.
12. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–219. Springer-Verlag, 2001.
13. H. Cui, Y. Mu and M. H. Au. Public-Key Encryption Resilient to Linear Related-Key Attacks. In *SecureComm*, volume 127 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 182–196. Springer-Verlag, 2013.
14. D. Naccache. Secure and *Practical* Identity-Based Encryption. *CoRR*, abs/cs/0510042, 2005.
15. R. Gennaro, A. Lysyanskaya, T. Malkin, S. Micali, and T. Rabin. Algorithmic tamper-proof (atp) security: Theoretical foundations for security against hardware tampering. In *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 258–277. Springer, 2004.

16. S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.
17. V. Goyal, A. O'Neill, and V. Rao. Correlated-input secure hash functions. In *TCC*, volume 6597 of *Lecture Notes in Computer Science*, pages 182–200. Springer, 2011.
18. J. Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. In *ASIACRYPT*, volume 4284 of *Lecture Notes in Computer Science*, pages 444–459. Springer, 2006.
19. 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. In *USENIX Security Symposium*, pages 45–60. USENIX Association, 2008.
20. E. Kiltz. Chosen-ciphertext security from tag-based encryption. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 581–600. Springer, 2006.
21. E. Kiltz, P. Mohassel, and A. O'Neill. Adaptive trapdoor functions and chosen-ciphertext security. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 673–692. Springer, 2010.
22. S. Lucks. Ciphers secure against related-key attacks. In *FSE*, volume 3017 of *Lecture Notes in Computer Science*, pages 359–370. Springer, 2004.
23. T. Matsumoto, K. Kato, and H. Imai. Speeding up secret computations with insecure auxiliary devices. In *CRYPTO*, volume 403 of *Lecture Notes in Computer Science*, pages 497–506. Springer, 1990.
24. S. Micali and L. Reyzin. Physically observable cryptography (extended abstract). In *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 278–296. Springer, 2004.
25. J.-J. Quisquater and D. Samyde. Electromagnetic analysis (ema): Measures and counter-measures for smart cards. In *E-smart*, volume 2140 of *Lecture Notes in Computer Science*, pages 200–210. Springer, 2001.
26. H. Wee. Efficient chosen-ciphertext security via extractable hash proofs. In *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 314–332. Springer, 2010.
27. H. Wee. Public key encryption against related key attacks. In *Public Key Cryptography*, volume 7293 of *Lecture Notes in Computer Science*, pages 262–279. Springer, 2012.