Parallel Key-Insulated Public Key Encryption Without Random Oracles

Benoît Libert[1]*, Jean-Jacques Quisquater[1], and Moti Yung[2]

[1] UCL, Microelectronics Laboratory, Crypto Group (Belgium)
[2] RSA Labs and Columbia University (USA)

Abstract. Key-insulated cryptography is a crucial technique for protecting private keys. To strengthen the security of key-insulated protocols, Hanaoka, Hanaoka and Imai recently introduced the idea of parallel key-insulated encryption (PKIE) where distinct physically-secure devices (called helpers) are independently used in key updates. Their motivation was to reduce the risk of exposure for helpers by decreasing the frequency of their connections to insecure environments. Hanaoka *et al.* showed that it was non-trivial to achieve a PKIE scheme fitting their model and proposed a construction based on the Boneh-Franklin identity-based encryption (IBE) scheme. The security of their system was only analyzed in the idealized random oracle model. In this paper, we provide a fairly efficient scheme which is secure in the standard model (i.e. without random oracles). To do so, we first show the existence of a relation between PKIE and the notion of aggregate signatures (AS) suggested by Boneh *et al.* We then describe our random oracle-free construction using bilinear maps. Thus, our contributions are both on the concrete side, namely the first realization of parallel key-insulated encryption without the random oracle idealization, and on the conceptual side revealing the relationships between two seemingly unrelated primitives.

Keywords. parallel key-insulated encryption, standard model, pairings.

1 Introduction

Nowadays, protecting cryptographic keys is an issue of huge importance. Hazards of key exposure are indeed ever-increasing due the growing use of mobile devices allowing remote unprotected access. This problem has been major concern to the research community for a decade. It is certainly a more serious threat for security customers than algorithms attempting to solve number theoretic problems by brute force.

To mitigate its potential damages, key-evolving protocols were studied in various flavors: forward-security [1, 3, 14], intrusion-resilience [27, 19] and key-insulation [20, 21]. The latter paradigm was introduced in 2002 by Dodis, Katz, Xu and Yung [20]. It was motivated by the upcoming setting of "Ubiquitous

* Work done during a visit at the CS Dept. of the Columbia University. This author also acknowledges the DGTRE's First Europe Program of the Walloon Region in Belgium for his financial support.

Computing" where each user will possess more than one computer (e.g. a computer at the office and a mobile phone which is also a computer) where not all computers have the same availability and/or security. The general idea of key-insulated security was to store long-term keys in a physically-secure but computationally-limited device called *base* or *helper*. Short-term secret keys are kept by users on a powerful but insecure device where cryptographic computations take place. Short term secrets are then refreshed at discrete time periods via interaction between the user and the base while the public key remains unchanged throughout the lifetime of the system. For a total of N time periods, such a mechanism is said to be (t, N)-key-insulated if a compromise of up to t periods leaves the remaining $N - t$ time periods unharmed. A scheme is additionally said *strongly* key-insulated when adversaries corrupting the base remain unable to perform private key operations on behalf of the user.

An increased tolerance against key exposures is thus allowed by frequent updates of private keys. This unfortunately implies frequent connections between the helper device and the network and thereby an increased risk of helper key exposure. A theft of the helper's base key is quite damaging as it typically requires to restart the system with a new public key. It even jeopardizes strongly key-insulated protocols where the additional exposure of a single time period at the user definitely crashes the system. This recently motivated Hanaoka, Hanaoka and Imai [26] to make a significant step forward and introduce the concept of parallel key-insulated encryption (PKIE for short) where distinct independent helpers are alternatively used in key update operations. As argued in [26], the involvement of two helpers may simultaneously increase the security of helpers and users by allowing for frequent updates without incurring a higher risk of helper exposure. In [26], Hanaoka *et al.* provided a PKIE constuction which will be referred to as the HHI scheme in this paper. It was obtained from the Boneh-Franklin identity-based encryption (IBE) scheme [8] and provably fits properly defined security requirements in the random oracle model [4]. However, a proof in the random oracle model can only serve as a heuristic argument as it is known (see e.g. [13]) not to imply the security in the real world.

It is natural to wonder if secure PKIE schemes exist in the standard model and if they can be generically built from IBE. Indeed, equivalence relations are known [5, 20] between $(N - 1, N)$-key-insulated systems and identity-based schemes [36], hierarchical extensions of which [24, 6, 7] also allowed for the design of other key-evolving protocols [29, 14, 19, 16, 38, 7].

To answer those questions, we first point out an intuitive relation between PKIE and IBE systems that involve a signature scheme supporting aggregation as in the aggregate signing (AS) protocol put forth by Boneh *et al.* [9]. We then describe a PKIE scheme which, although non-generic, is demonstrably secure in the sense of [26] without resorting to the random oracle methodology. Our construction uses a selective-ID secure IBE due to Boneh and Boyen [6] as a building block. It is fairly efficient and enjoys a security resting on the (by now well-studied) Decisional Bilinear Diffie-Hellman assumption.

In the upcoming sections, we first recall functional definitions and security notions for PKIE schemes. Section 3 discusses necessary conditions for building such a primitive from identity-based protocols. Our system and its security are respectively analyzed in sections 4 and 5. Section 6 then explains how to further secure our protocol against chosen-ciphertext attacks.

2 Preliminaries

2.1 Model and Security Notions

A PKIE scheme over N stages consists of the following five algorithms.

Key generation: takes a security parameter λ and returns helpers' private keys mst_1, mst_2, a user's private key usk_0 and a public key pk.

Helper-Update: takes as input helper j's private key mst_j and a period number i to return an update key hsk_i if $i = j \bmod 2$ and $\perp$ otherwise.

User-Update: is given user's private key usk_{i-1} for period $i-1$, an update key hsk_i and computes the private key usk_i for period i.

Encrypt: is given a message m, a public key pk and a period number $i \in \{1, \ldots, N\}$ and returns a ciphertext σ.

Decrypt: given a ciphertext σ, a period number i and the matching private key usk_i, returns either a plaintext m or $\perp$.

The usual completeness requirement imposes $\mathbf{Decrypt}(\mathsf{pk}, \mathsf{usk}_i, \sigma) = m$ whenever $\sigma = \mathbf{Encrypt}(m, i, \mathsf{pk})$ for any $i \in \{1, \ldots, N\}$.

In the basic (i.e. non-strong) key-insulation security, if no helper is compromised, the exposure of any short-term secret leaves other periods safe as in [20, 21]. Besides, if a single helper is broken into while some stage i is exposed, only one other stage adjacent to i is also exposed (recall that even strongly key-insulated traditional schemes collapse in this scenario).

Definition 1. *A PKIE scheme is (t, ϵ)-secure against chosen-ciphertext attacks if no adversary has better advantage than ϵ in the following game within running time t.*

1. *The challenger $\mathcal{C}$ runs the key generation algorithm, hands pk to the adversary $\mathcal{A}$ and keeps mst_0, mst_1 and uks_0 to itself.*
2. *$\mathcal{A}$ adaptively issues queries which are either:*
   - *Exposure queries $\langle j, \mathtt{class} \rangle$: if $\mathtt{class} =$ "user", $\mathcal{C}$ runs helper and user update algorithms to generate usk_j and send it to $\mathcal{A}$. If $\mathtt{class} =$ "helper", $\mathcal{A}$ obtains mst_j.*
   - *Decryption queries $\langle j, \sigma \rangle$: $\mathcal{C}$ responds by generating usk_j (via calls to update algorithms) to decrypt σ and pass the result to $\mathcal{A}$.*
3. *At some point, $\mathcal{A}$ comes up with messages M_0, M_1 and a period number $j^\star \in \{1, \ldots, N\}$. She obtains a challenge $\sigma^\star = \mathbf{Encrypt}(M_{b^\star}, j^\star, \mathsf{pk})$ for a random bit $b^\star \overset{R}{\leftarrow} \{0, 1\}$ selected by $\mathcal{C}$.*
4. *$\mathcal{A}$ issues new queries as in stage 2. She finally outputs $b \in \{0, 1\}$ and wins if $b = b^\star$ provided*
   - *$\langle j^\star, \sigma^\star \rangle$ does not appear in the list of decryption queries,*

- *$\langle j^\star$, "user"$\rangle$ is not in the list of exposure queries,*
- *$\langle j^\star - 1$, "user"$\rangle$ and $\langle 2 - (j^\star \bmod 2)$, "helper"$\rangle$ do not simultaneously appear in the list of exposure queries and neither does the pair $\langle j^\star + 1$, "user"$\rangle$, $\langle (j^\star \bmod 2) + 1$, "helper"$\rangle$,*
- *$\langle 1$, "helper"$\rangle$ and $\langle 2$, "helper"$\rangle$ were not both queried.*

As usual, $\mathcal{A}$'s advantage is measured by $Adv^{PKIE}(\mathcal{A}) = |\Pr[b = b^\star] - 1/2|$.

This definition considers two kinds of adversaries: Type I attackers do not corrupt helpers during the game. In contrast, Type II adversaries corrupt exactly one helper without requesting a private key that would trivially expose the target period $j^\star$. For example, if $j^\star$ is odd, $\mathcal{A}$ may not obtain $\mathsf{usk}_{j^\star-1}$ if she ever learns mst_1 as the latter is involved in all updates from even to odd time periods. Besides, she is disallowed to query $\mathsf{usk}_{j^\star+1}$ if she also receives mst_2 since $\mathsf{usk}_{j^\star}$ could be trivially retrieved from $\mathsf{usk}_{j^\star+1}$ and the helper's key that allowed the update from period $j^\star$ to $j^\star + 1$.

According to [20, 21], a parallel key-insulated scheme is said *strongly* key-insulated if breaking into all helpers does not help the adversary as long as she does not also obtain any user secret for any period. Unlike [26], we follow [20, 21] and address this problem in a separate game where $\mathcal{A}$ is provided with *both* base keys and may not request usk_i for any i.

2.2 Bilinear Maps and Related Problems

Groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order p are called *bilinear map groups* if there is a mapping $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with the following properties:

1. bilinearity: $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G} \times \mathbb{G}$ and $a, b \in \mathbb{Z}$;
2. efficient computability for any input pair;
3. non-degeneracy: $e(g, h) \neq 1_{\mathbb{G}_T}$ whenever $g, h \neq 1_{\mathbb{G}}$.

We require the intractability of these problems in bilinear map groups.

Definition 2. *Let $(\mathbb{G}, \mathbb{G}_T)$ be bilinear map groups of order p and $g \in \mathbb{G}$.*

1. *the* **Bilinear Diffie-Hellman Problem** *(BDH) [28, 8] is to compute $e(g, g)^{abc} \in \mathbb{G}_T$ given (g^a, g^b, g^c);*
2. *the* **Decision Bilinear Diffie-Hellman Problem** *(DBDH) is to distinguish the distributions $(g^a, g^b, g^c, e(g, g)^{abc})$ and $(g^a, g^b, g^c, e(g, g)^z)$. A distinguisher $\mathcal{B}$ (t, ε)-solves it if it runs in time at most t and*

$$\Big| Pr[\mathcal{B}(g^a, g^b, g^c, e(g, g)^{abc}) = 1 | a, b, c \xleftarrow{R} \mathbb{Z}_p^*] \\ - Pr[\mathcal{B}(g^a, g^b, g^c, e(g, g)^z) = 1 | a, b, c, z \xleftarrow{R} \mathbb{Z}_p^*] \Big| \geq \varepsilon.$$

3 On Obtaining PKIE from IBE

The HHI scheme [26] (recalled in appendix A) uses the Boneh-Franklin IBE [8] as a building block. It is pointed out in [26] that constructing parallel key-insulated schemes from identity-based ones is non-trivial. For instance, one cannot settle for simply combining two IBE schemes by letting two Private Key Generators (PKGs) alternatively act as helpers for even and odd time periods. This naive approach indeed leaves half of the stages exposed after the compromise of only one helper. Another idea that falls short is to doubly encrypt (using the techniques of [22]) messages for period i under identities i and $i-1$ and let private key usk_i consist of IBE private keys for identities i and $i-1$. Unfortunately, the key $\mathsf{usk}_{i^\star}$ may be exposed by merely corrupting periods $i^\star - 1$ and $i^\star + 1$.

Nevertheless, the HHI construction stems from a careful double application of the Boneh-Franklin IBE. Although it was not explicitly mentioned in [26], the key idea is to let a private key for period i be an aggregation of identity-based private keys for periods i and $i-1$. We recall that identity-based cryptosystems traditionally involve private keys that are authorities' signatures on identities. As noted in [9], the signature algorithm [11] that derives private keys from identifiers in [8] is compatible with a signature aggregation. This means that n signatures generated by distinct signers on possibly distinct messages may be merged into a single signature in such a way that verifiers can still ensure that all signers each signed their original message. In [26], a private key for period i is the aggregation of both helpers' signatures on messages i and $i-1$. The security in the sense of definition 1 relies on the intractability of extracting individual signatures from an aggregate of even only two signatures. In Boneh *et al.*'s scheme [9], this problem was shown equivalent to the Diffie-Hellman problem in [17].

An intuitive connection thus turns out to exist between parallel key-insulated cryptosystems and identity-based encryption schemes extracting private keys using a signature scheme supporting aggregation. The infeasibility of extracting individual signatures from a 2-aggregate appears as a necessary condition for the underlying AS to provide a secure PKIE system. In the next section, we take advantage of this connection to devise a PKIE in the standard model. Our construction uses the selective-ID secure[3] [14] scheme of Boneh-Boyen [6] as a starting point. As previously mentioned by Canetti, Halevi and Katz [14], selective-ID secure schemes are sufficient as building blocks for key-evolving protocols. Indeed, in security proofs, simulators have to guess in advance which time period will be the prey of attacks. This degrades security bounds by a factor that remains acceptable for any realistic number of time periods (such as $N \leq 2^{30}$). We emphasize that constructing a PKIE scheme using Waters's fully secure IBE [37] would be overkill and would not yield a tighter reduction here[4].

In the selective-ID secure IBE of [6], private keys are computed using a signature scheme which bears similarities with Waters's signature [37] but is only selective-message secure against chosen-message attacks. Unlike a recent variant [33] where signatures are sequentially computed, it only supports a limited aggregation as the size of an aggregate remains linear in the number of signers (as

[3] i.e. secure in a model where attackers are required to announce the identity they intend to attack ahead of time, even before seeing the system parameters.

[4] Indeed, the condition for the simulator of [37] not to abort in the challenge phase would have to be satisfied for both "identities" i and $i-1$.

in a similar method suggested in [34]). However, this scheme is sufficient for the pursued goal here as, in our construction, private keys are aggregates of only 2 individual signatures.

4 A Scheme with Chosen-Plaintext Security

As in [26], private keys stored by users at period i are 2-aggregate signatures computed by helpers on "messages" i and $i-1$.

Key generation: given a security parameter $\lambda \in \mathbb{N}$, this algorithm

1. chooses bilinear map groups $(\mathbb{G}, \mathbb{G}_T)$ of order $p > 2^{\lambda}$, a generator $g \in \mathbb{G}$, a collision-resistant hash function $H : \mathbb{N} \to \mathbb{Z}_p^*$ and a pseudorandom function $f : \{0,1\}^{1+\log_2 p} \times \mathbb{N} \to \mathbb{Z}_p^*$,
2. picks $\alpha_1, \alpha_2 \xleftarrow{R} \mathbb{Z}_p^*$ and computes $g_1 = g^{\alpha_1}$, $g_2 = g^{\alpha_2}$,
3. selects elements $h, g_1', g_2' \xleftarrow{R} \mathbb{G}$ and defines functions $F_1, F_2 : \mathbb{N} \to \mathbb{G}$ as $F_1(i) = g_1^{I_i} g_1'$ and $F_2(i) = g_2^{I_i} g_2'$ where $I_i = H(i) \in \mathbb{Z}_p^*$,
4. initializes the user's private key to
$$\mathsf{usk}_0 = \big(h^{\alpha_1+\alpha_2} F_1(-1)^{r_{-1}} F_2(0)^{r_0}, g^{r_0}, g^{r_{-1}}\big)$$
where $r_{-1} = f(\mathsf{sd}_1, -1)$ and $r_0 = f(\mathsf{sd}_2, 0)$ are respectively derived from randomly chosen seeds $\mathsf{sd}_1, \mathsf{sd}_2 \in \{0,1\}^{1+\log_2 p}$,
6. Helpers' private keys are set to $\mathsf{mst}_1 = \mathsf{sd}_1$ and $\mathsf{mst}_2 = \mathsf{sd}_2$ and the public key is
$$\mathsf{pk} := \{p, \mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, h, g_1', g_2', H, f\}.$$
Elements α_1, α_2 are *erased.*

Helper-Update: given $\mathsf{mst}_j = \mathsf{sd}_j$ and a period number $i \in \{1, 2, \ldots, N\}$, helper $j \in \{1, 2\}$

1. returns $\perp$ if $i \neq j \bmod 2$,
2. computes $r_{i-2} = f(\mathsf{sd}_j, i-2)$, $r_i = f(\mathsf{sd}_j, i)$
3. outputs an update key $\mathsf{hsk}_i = \big(F_j(i)^{r_i} / F_j(i-2)^{r_{i-2}}, g^{r_i - r_{i-2}}\big)$

User-Update: given usk_{i-1}, hsk_i and i,

1. parse hsk_i into (h_i, h_i') and usk_{i-1} into $(u_{i-1,0}, u_{i-1,1}, u_{i-1,2})$,
2. set $\mathsf{usk}_i = (u_{i-1,0} \cdot h_i, u_{i-1,2} \cdot h_i', u_{i-1,1})$,
3. return usk_i, discard usk_{i-1} and hsk_i.

At time period i, user's private key is always set to
$$\mathsf{usk}_i = \big(h^{\alpha_1+\alpha_2} F_j(i)^{r_i} F_{j-1}(i-1)^{r_{i-1}}, g^{r_i}, g^{r_{i-1}}\big),$$
with $j = 2 - (i \bmod 2)$ and for uniformly distributed exponents $r_i = f(\mathsf{sd}_j, i)$, $r_{i-1} = f(\mathsf{sd}_{j-1}, i-1) \in \mathbb{Z}_p^*$ determined by helpers.

Encrypt: given pk, $i \in \mathbb{N}$, a message $m \in \mathbb{G}_T$ is encrypted into
$$\sigma = \big(m \cdot e(h, g_1 g_2)^s, g^s, F_j(i)^s, F_{j-1}(i-1)^s\big)$$
where $s \xleftarrow{R} \mathbb{Z}_p^*$ is randomly chosen and $j = 2 - (i \bmod 2)$.

Decrypt: given $\sigma = (A, B, C, D)$ and $\mathsf{usk}_i = (u_{i,0}, u_{i,1}, u_{i,2})$, compute

$$m = A \cdot \frac{e(C, u_{i,1}) \cdot e(D, u_{i,2})}{e(B, u_{i,0})}$$

The completeness is checked by noting that $\mathsf{usk}_i = (u_{i,0}, u_{i,1}, u_{i,2})$ satisfies

$$e(u_{i,0}, g) = e(h, g_1 g_2) \cdot e\left(F_j(i), u_{i,1}\right) \cdot e\left(F_{j-1}(i-1), u_{i,2}\right)$$

and raising both members of this equality to the power s.

The function f plays the crucial role of a "memory function" seeded by sd_j for $j = 2 - (i \bmod 2)$ and allowing helpers to remember the exponent r_{i-2} of their latest update. Those exponents must be unpredictable without the seed sd_j as an adversary obtaining usk_{i-1} could trivially compute a private key for period i without knowing hsk_i if she could find out r_{i-2}.

5 Security

Theorem 1. *If no algorithm (t, ε)-breaks the Decision Bilinear Diffie-Hellman assumption, the scheme is $(t', 4N\varepsilon)$-secure against chosen-plaintext attacks for $t' < t - O(N\tau_{exp})$ where N is the number of time periods and τ_{exp} stands for the cost of an exponentiation in $\mathbb{G}$.*

Proof. We construct an algorithm $\mathcal{B}$ that solves the DBDH problem in $(\mathbb{G}, \mathbb{G}_T)$ using an adversary $\mathcal{A}$ against the IND-CPA security of our scheme. The input of $\mathcal{B}$ is a tuple $(g^a, g^b, g^c, T) \in \mathbb{G}^3 \times \mathbb{G}_T$ and it aims at deciding whether $T = e(g,g)^{abc}$ thanks to its interaction with $\mathcal{A}$. As explained in section 2.1, two kinds of adversaries are distinguished:

Type I adversaries: do not corrupt helpers during the game.
Type II adversaries: corrupt exactly one helper without exposing a private key that would trivially compromise the attacked period $i^\star$.

At the outset of the simulation, $\mathcal{B}$ tosses a coin $\mathcal{COIN} \stackrel{R}{\leftarrow} \{0,1\}$ to guess which kind of attack $\mathcal{A}$ will produce. If $\mathcal{COIN} = 0$, it expects to face a Type I adversary. If $\mathcal{COIN} = 1$, it forecasts a Type II behaviour from $\mathcal{A}$. Our simulator $\mathcal{B}$ also chooses an index $\ell \stackrel{R}{\leftarrow} \{1, \ldots, N\}$ as a guess for the time period to be attacked by $\mathcal{A}$. W.l.o.g., we shall assume that ℓ is odd as the case of an even ℓ can be handled in a completely similar manner. In all cases, $\mathcal{B}$ generates the public key as follows. It selects $\gamma \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$, and defines $g_1 = g^a$, $g_2 = g_1^\gamma = (g^a)^\gamma$ and $h = g^b$. It also computes $I_\ell = H(\ell)$ and $I_{\ell-1} = H(\ell - 1)$ and sets $g_1' = g_1^{-I_\ell} g^{t_1}$, $g_2' = g_2^{-I_{\ell-1}} g^{t_2}$ for randomly chosen $t_1, t_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and thereby implicitly defines functions

$$F_1(i) = g_1^{I_i - I_\ell} g^{t_1}, \; F_2(i) = g_2^{I_i - I_{\ell-1}} g^{t_2}.$$

$\mathcal{A}$ is initialized on input of $(g, g_1, g_2, h, g_1', g_2')$ and issues exposure queries that are handled differently when $\mathcal{COIN} = 0$ and $\mathcal{COIN} = 1$. Informally, $\mathcal{B}$ uses the fact that $\mathcal{A}$ has no information on exponents r_i ($i \in \{1,\ldots,N\}$) unless she breaks into helper $j = 2-(i \bmod 2)$. When dealing with exposure queries, these exponents may be freely chosen for both even and odd stages when facing Type I attacks. In Type II attacks, they are constrained for either odd or even stages and $\mathcal{B}$ has to guess in advance the parity of constrained indexes.

• $\mathcal{COIN} = 0$: $\mathcal{B}$ aborts if $\mathcal{A}$ issues a query $\langle i, \texttt{class}\rangle$ with $\texttt{class}$ = "helper" or with $i = \ell$. Otherwise, it can answer the query as private keys usk_i are computable for $i \neq \ell$. To do so, $\mathcal{B}$ selects $r_0 \overset{R}{\leftarrow} \mathbb{Z}_p^*$. For $i = 1,2,\ldots,(\ell-1)/2$, it picks $r_{2i-1}, r_{2i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and computes

$$\mathsf{usk}_{2i} = \Big(h^{-\frac{(1+\gamma)t_1}{I_{2i-1}-I_\ell}} F_1(2i-1)^{r_{2i-1}} F_2(2i)^{r_{2i}}, h^{-\frac{1+\gamma}{I_{2i-1}-I_\ell}} g^{r_{2i-1}}, g^{r_{2i}}\Big), \quad (1)$$

$$\mathsf{usk}_{2i-1} = \Big(h^{-\frac{(1+\gamma)t_1}{I_{2i-1}-I_\ell}} F_1(2i-1)^{r_{2i-1}} F_2(2i-2)^{r_{2i-2}}, g^{r_{2i-2}}, h^{-\frac{1+\gamma}{I_{2i-1}-I_\ell}} g^{r_{2i}}\Big) \quad (2)$$

where $I_{2i-1} = H(2i-1)$. We observe that $\mathsf{usk}_1,\ldots,\mathsf{usk}_{\ell-1}$ have the correct shape. If we define $\tilde{r}_{2i-1} = r_{2i-1} - b(1+\gamma)/(I_{2i-1}-I_\ell)$, we have

$$\begin{aligned} h^{-\frac{(1+\gamma)t_1}{I_{2i-1}-I_\ell}} F_1(2i-1)^{r_{2i-1}} &= h^{-\frac{(1+\gamma)t_1}{I_{2i-1}-I_\ell}} \big(g_1^{I_{2i-1}-I_\ell} g^{t_1}\big)^{\tilde{r}_{2i-1}+\frac{b(1+\gamma)}{I_{2i-1}-I_\ell}} && (3)\\ &= {g_1}^{b(1+\gamma)} (g_1^{I_{2i-1}-I_\ell} g^{t_1})^{\tilde{r}_{2i-1}} && (4)\\ &= g^{(a\gamma+a)b} F_1(2i-1)^{\tilde{r}_{2i-1}} && (5)\\ &= h^{\alpha_1+\alpha_2} F_1(2i-1)^{\tilde{r}_{2i-1}} && (6)\end{aligned}$$

and $h^{-\frac{1+\gamma}{I_{2i-1}-I_\ell}} g^{r_{2i-1}} = g^{-\frac{b(1+\gamma)}{I_{2i-1}-I_\ell}} g^{r_{2i-1}} = g^{\tilde{r}_{2i-1}}$ for $i = 1,\ldots,(\ell-1)/2$.

Next, $\mathcal{B}$ repeats a similar procedure to generate $\mathsf{usk}_{\ell+1},\ldots,\mathsf{usk}_N$. It picks $r_\ell \overset{R}{\leftarrow} \mathbb{Z}_p^*$. For $i = (\ell-1)/2+1,\ldots,N/2$ (we assume that N is even), it chooses $r_{2i}, r_{2i+1} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and computes

$$\mathsf{usk}_{2i} = \Big(h^{-\frac{(1+\gamma^{-1})t_2}{I_{2i}-I_{\ell-1}}} F_2(2i)^{r_{2i}} F_1(2i-1)^{r_{2i-1}}, h^{-\frac{1+\gamma^{-1}}{I_{2i}-I_{\ell-1}}} g^{r_{2i}}, g^{r_{2i-1}}\Big), \quad (7)$$

$$\mathsf{usk}_{2i+1} = \Big(h^{-\frac{(1+\gamma^{-1})t_2}{I_{2i}-I_{\ell-1}}} F_2(2i)^{r_{2i}} F_1(2i+1)^{r_{2i+1}}, g^{r_{2i+1}}, h^{-\frac{1+\gamma^{-1}}{I_{2i}-I_{\ell-1}}} g^{r_{2i}}\Big) \quad (8)$$

where $I_{2i} = H(2i)$. We check that $\mathsf{usk}_{\ell+1},\ldots,\mathsf{usk}_N$ are correct keys as, if we define $\tilde{r}_{2i} = r_{2i} - b(1+\gamma^{-1})/(I_{2i}-I_{\ell-1})$ for $i = (\ell-1)/2+1,\ldots,N/2$,

$$\begin{aligned} h^{-\frac{(1+\gamma^{-1})t_2}{I_{2i}-I_{\ell-1}}} F_2(2i)^{r_{2i}} &= h^{-\frac{(1+\gamma^{-1})t_2}{I_{2i}-I_{\ell-1}}} \big(g_2^{I_{2i}-I_{\ell-1}} g^{t_2}\big)^{\tilde{r}_{2i}+\frac{b(1+\gamma^{-1})}{I_{2i}-I_{\ell-1}}} && (9)\\ &= {g_2}^{b(1+\gamma^{-1})} (g_2^{I_{2i}-I_{\ell-1}} g^{t_2})^{\tilde{r}_{2i}} && (10)\\ &= g^{(a\gamma+a)b} F_2(2i)^{\tilde{r}_{2i}} = h^{\alpha_1+\alpha_2} F_2(2i)^{\tilde{r}_{2i}} && (11)\end{aligned}$$

and $h^{-\frac{1+\gamma^{-1}}{I_{2i}-I_{\ell-1}}} g^{r_{2i}} = g^{-\frac{b(1+\gamma^{-1})}{I_{2i}-I_{\ell-1}}} g^{r_{2i}} = g^{\tilde{r}_{2i}}$.

• $\mathcal{COIN} = 1$: $\mathcal{B}$ expects $\mathcal{A}$ to corrupt either $\mathsf{mst}_1 = \mathsf{sd}_1$ or $\mathsf{mst}_2 = \mathsf{sd}_2$. It picks random values $\mathbf{b} \overset{R}{\leftarrow} \{1,2\}$ and $\mathsf{sd}_\mathbf{b} \overset{R}{\leftarrow} \{0,1\}^{1+\log_2 p}$ and bets on an exposure query involving $\mathsf{mst}_\mathbf{b} = \mathsf{sd}_\mathbf{b}$. As ℓ is odd, if $\mathcal{A}$ indeed attacks stage ℓ, she is restricted not to request $\mathsf{usk}_{\ell-1}$ (resp. $\mathsf{usk}_{\ell+1}$) if $\mathbf{b}=1$ (resp. $\mathbf{b}=2$). When $\mathcal{A}$ issues a query $\langle i, \texttt{class}\rangle$, $\mathcal{B}$ returns $\mathsf{sd}_{\mathbf{b}}$ if $\texttt{class}$ = "helper" and $i=\mathbf{b}$. It aborts if $\texttt{class}$ = "helper" with $i=\overline{\mathbf{b}}$ (where $\overline{\mathbf{b}}=2$ if $\mathbf{b}=1$ and vice versa). When $\texttt{class}$ = "user", it also aborts if $i=\ell$, if $i=\ell-1$ while $\mathbf{b}=1$ and if $i=\ell+1$ while $\mathbf{b}=2$. Otherwise, two cases are distinguished:

$\mathbf{b}=1$: we have $i \neq \ell-1, \ell$. For all $i=1,\ldots,N/2$, exponents r_{2i-1} are imposed by the relation $r_{2i-1}=f(\mathsf{sd}_1, 2i-1)$ but exponents r_{2i} can be freely chosen. For $i=0,\ldots,(\ell+1)/2-2,(\ell+1)/2,\ldots,N/2$, $\mathcal{B}$ chooses $r_{2i} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and generates $\mathsf{usk}_{2i}, \mathsf{usk}_{2i+1}$ following equations (7)-(11). Therefore, it obtains all private keys but $\mathsf{usk}_{\ell-1}, \mathsf{usk}_{\ell}$ (though usk_0 and usk_{N+1} are never requested by the adversary, they are computable). Those private keys have the correct shape for uniformly distributed (unknown) elements $\tilde{r}_{2i} \in \mathbb{Z}_p^*$.

$\mathbf{b}=2$: $\mathcal{B}$ has to compute private keys usk_i with $i \neq \ell, \ell+1$ as $\mathcal{A}$ is assumed not to request $\mathsf{usk}_{\ell+1}$. This time, exponents of even time periods have to comply with the constraint $r_{2i}=f(\mathsf{sd}_2, 2i)$ for all i but exponents r_{2i-1} are free. For $i=1,\ldots,(\ell-1)/2,(\ell-1)/2+2,\ldots,N/2$, $\mathcal{B}$ chooses $r_{2i-1} \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and computes usk_{2i}, usk_{2i-1} according to equations (1)-(6) and thereby obtains well-formed $\mathsf{usk}_1,\ldots,\mathsf{usk}_{\ell-1},\mathsf{usk}_{\ell+2},\ldots,\mathsf{usk}_N$ for random (unknown) implicitly defined $\tilde{r}_{2i-1}$.

Challenge: when $\mathcal{A}$ decides that phase 1 is over, she comes up with messages M_0, M_1 and a target time period $i^\star$, $\mathcal{B}$ halts and reports "failure" if $i^\star \neq \ell$. Otherwise, it flips a fair coin $b^\star \overset{R}{\leftarrow} \{0,1\}$ a returns the challenge

$$\sigma^\star = \Big(M_{b^\star}\cdot T^{1+\gamma}, g^c, (g^c)^{t_1}, (g^c)^{t_2}\Big).$$

Since $F_1(\ell)=g^{t_1}$ and $F_2(\ell-1)=g^{t_2}$, $\sigma^\star$ has the same distribution as the output of the encryption algorithm if $T=e(g,g)^{abc}$. In contrast, if T is random in $\mathbb{G}_T$, $\sigma^\star$ is independent of $b^\star$ and $\mathcal{A}$ cannot guess $b^\star$ with a higher probability than 1/2. Hence, $\mathcal{B}$ deduces that $T=e(g,g)^{abc}$ if $\mathcal{A}$'s final output equals $b^\star$. Otherwise, it bets that $T \in_R \mathbb{G}_T$.

When assessing $\mathcal{B}$'s success probability, we note that it may fail to provide $\mathcal{A}$ with a consistent view because of the following events:

E_1 : a key exposure is made for period ℓ
E_2 : a helper key exposure occurs and $\mathcal{COIN}=0$
E_3 : helper $\overline{\mathbf{b}}$'s private key is exposed while $\mathcal{COIN}=1$
E_4 : a key exposure on $\mathsf{usk}_{\ell-1}$ occurs while $\mathbf{b}=1$ and $\mathcal{COIN}=1$
E_5 : a key exposure on $\mathsf{usk}_{\ell+1}$ occurs while $\mathbf{b}=2$ and $\mathcal{COIN}=1$

We also consider the following events:

H_0 : $\mathcal{B}$ correctly guesses $i^\star=\ell$
H_1 : $\mathcal{B}$ successfully foresees the kind of attack produced by $\mathcal{A}$
H_2 : $\mathcal{B}$ luckily predicts which helper's key is exposed when $\mathcal{COIN}=1$

Clearly $\Pr[H_0]=1/N$ and $\Pr[H_1]=1/2$. Also, we have $H_0 \Rightarrow \neg E_1$, $H_1 \Rightarrow \neg E_2$, $H_2 \Rightarrow \neg E_3$ and $H_2 \wedge H_0 \Rightarrow \neg E_4 \wedge \neg E_5$. The conjunction of events H_0, H_1 and H_2 is readily seen to occur with probability greater than $1/4N$ and it suffices to prevent a failure of the simulation. $\square$

6 Chosen-Ciphertext Security

Chosen-ciphertext security in the standard model can be achieved using ideas from [15, 10] but it is more directly obtained following the techniques of Boyen, Mei and Waters [12] which require to turn our scheme into a key-encapsulation mechanism (KEM) [35].

A KEM [35] is a public key algorithm that, instead of encrypting messages as a regular public key cryptosystem, takes only a public key as input and returns pairs (K, σ) made of a randomly distributed key K and an encapsulation σ of it. The reverse operation is achieved by a decapsulation algorithm which, on input of a private key and an encapsulation σ, either outputs a key K or a rejection message $\perp$. It is well-known [35] that a KEM immediately provides a public key encryption scheme when combined with a suitable symmetric cryptosystem.

The methods of [12] involve a piece of ciphertext acting as a checksum treated as part of an identity-based system by the simulator handling decryption queries.

In order to optimize the decapsulation algorithm, we use a trick suggested in [30, 32] to minimize the number of pairing calculations and render the consistency checking implicit in the computation of the key.

Key generation: is unchanged except that it additionally chooses a function $H' : \mathbb{G} \to \mathbb{Z}_p$ which is either a collision-resistant hash function or a suitable injective encoding (see [12] for details on how to define such an encoding). The algorithm also picks another element $g' \overset{R}{\leftarrow} \mathbb{G}$ to define the "checksum function" $F_3 : \mathbb{Z}_p \to \mathbb{G} : x \to F_3(x) = (g_1 g_2)^x g'$.

$$\mathsf{pk} := \{p, \mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, h, g_1', g_2', g', H, H', f\}.$$

Helper-Update and **User-Update** do not change. At period i, user's private key is still

$$\mathsf{usk}_i = \big(h^{\alpha_1+\alpha_2} F_j(i)^{r_i} F_{j-1}(i-1)^{r_{i-1}}, g^{r_i}, g^{r_{i-1}}\big),$$

with $j = 2 - (i \bmod 2)$ and $r_i = f(\mathsf{sd}_j, i)$, $r_{i-1} = f(\mathsf{sd}_{j-1}, i-1) \in \mathbb{Z}_p^*$.

Encapsulate: given i and pk, let $j = 2 - (i \bmod 2)$, pick $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and compute $A = g^s$, $\omega = H'(g^s) \in \mathbb{Z}_p$. Set $B = F_j(i)^s$, $C = F_{j-1}(i-1)^s$ and $D = F_3(\omega)^s$ to get

$$\sigma = (A, B, C, D) = \big(g^s, F_j(i)^s, F_{j-1}(i-1)^s, F_3(\omega)^s\big)$$

which encapsulates the key $K = e(h, g_1 g_2)^s$.

Decapsulate: given $\sigma = (A, B, C, D)$ and $\mathsf{usk}_i = (u_{i,0}, u_{i,1}, u_{i,2})$, the receiver sets $\omega = H'(A) \in \mathbb{Z}_p$, picks $z_1, z_2, z_3 \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and computes

$$K = \frac{e\big(A, u_{i,0}\ F_j(i)^{z_1} F_{j-1}(i-1)^{z_2} F_3(\omega)^{z_3}\big)}{e(B, u_{i,1}\ g^{z_1}) \cdot e(C, u_{i,2}\ g^{z_2}) \cdot e(D, g^{z_3})} \tag{12}$$

To explain the decapsulation mechanism, we note that any properly formed encapsulation satisfies the (publicly verifiable) conditions

$$\tau_1 = \frac{e(A, F_j(i))}{e(B, g)} = 1, \ \tau_2 = \frac{e(A, F_{j-1}(i-1))}{e(C, g)} = 1, \ \tau_3 = \frac{e(A, F_3(\omega))}{e(D, g)} = 1_{\mathbb{G}_T}.$$

The naive approach is to return $K = e(A, u_{i,0})/(e(B, u_{i,1}) \cdot e(C, u_{i,2}))$ if they hold and $\perp$ (or a random $K \xleftarrow{R} \mathbb{G}_T$ from the key space) otherwise. This approach is perfectly equivalent to choose $z_1, z_2, z_3 \xleftarrow{R} \mathbb{Z}_p^*$ and return

$$K = \tau_1^{z_1} \cdot \tau_2^{z_2} \cdot \tau_3^{z_3} \cdot \frac{e(A, u_{i,0})}{e(B, u_{i,1}) \cdot e(C, u_{i,2})}$$

which is the actual decapsulated key if the encapsulation was correct and a random key otherwise. This alternative decapsulation mechanism is easily seen to be exactly the one suggested by relation (12).

Overall, the cost of the decapsulation operation amounts to a product of four pairings (which is much faster to compute than a naive evaluation of four pairings as discussed in [25]) plus a few exponentiations in $\mathbb{G}$.

In appendix B, we formally define the KEM counterpart of parallel key-insulated security. We then prove theorem 2 which claims the chosen-ciphertext security of our key-insulated KEM under the Decision BDH assumption.

Borrowing ideas from [31], we can construct a CCA-secure KEM with as short ciphertexts and almost as efficient decryption as in section 4. As in [31], this is obtained at the expense of longer private keys and a security resting on a slightly stronger assumption.

We also mention that a regular CCA-secure PKIE scheme can be directly achieved (without using the KEM-DEM framework) by implementing the checksum function F_3 using Waters's "hashing" technique [37], much in the fashion of the cryptosystem described in section 3 of [12]. It unfortunately entails a much longer public key and a looser reduction.

7 Strong Key-Insulation

The scheme inherently provides strong key-insulation thanks to the erasure of discrete logarithms α_1, α_2 of g_1, g_2 after generation of the initial key usk_0. Indeed, base keys $\mathsf{sd}_1, \mathsf{sd}_2$ (that uniquely determine $r_1, \ldots, r_N$) are useless to adversaries as long as they do not additionally obtain any local secret usk_i for any period.

To formally prove this fact (in a distinct game from the one of definition 1), we proceed as in the proof of theorem 1 with the sole difference that no key exposure query has to be tackled with. Hence, it does not matter if $\mathcal{A}$ knows exponents r_i.

8 Key-Insulated Encryption with Auxiliary Helper

In [2], Anh *et al.* generalized the notion of PKIE into a new primitive called key-insulated public key encryption *with auxiliary helper*. Such a scheme also involves two independent helpers but one of them is auxiliary and used in updates much less frequently (say every ℓ time periods) than the main helper. In practice, the latter can be a laptop performing updates every day while the auxiliary helper (e.g. a smart card) can be kept in a much safer location most of the time in order to decrease the chance of compromise of both helpers.

This results in noticeable enhancements since, when the main helper is compromised, another exposure at the user only harms ℓ time periods: the next update carried out by the auxiliary helper restores the security. Furthermore, simultaneous break-ins at the user and the auxiliary helper compromise at most two adjacent periods as long as the main helper is not also exposed.

In [2], the HHI system [26] was extended into a key-insulated scheme with auxiliary helper (implicitly using aggregate signatures). Our constructions can be similarly extended to fit security definitions of [2] without using random oracles.

9 Conclusion

We pinpointed connections between the concept of parallel key-insulated encryption and certain identity-based cryptosystems using signatures supporting aggregation. This observation allowed for the design of a secure system in the standard model.

This motivates the open problem (with or without random oracles) of increasing the number of helpers without paying an important loss of efficiency. Our scheme and the one of [26] can both be extended to involve more than two helpers but this entails a significant computational penalty.

References

1. R. Anderson. Two Remarks on Public Key Cryptology. Invited lecture, *ACM Conference on Computer and Communications Security*, 1997.
2. P.T.L. Anh, Y. Hanaoka, G. Hanaoka, K. Matsuura, H. Imai. Reducing the Spread of Damage of Key Exposures in Key-Insulated Encryption. In *Vietcrypt'06*, to appear in *LNCS series.*
3. M. Bellare, S. Miner. A Forward-Secure Digital Signature Scheme. In *Crypto'99*, *LNCS* 1666, pp. 431–448. Springer, 1999.
4. M. Bellare, P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *1st ACM Conference on Computer and Communications Security*, pages 62–73, ACM Press, 1993.
5. M. Bellare, A. Palacio. Protecting against Key Exposure: Strongly Key-Insulated Encryption with Optimal Threshold. Cryptology ePrint Archive: Report 2002/064, 2002.
6. D. Boneh, X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In *Eurocrypt'04*, *LNCS* 3027, pp. 223–238. Springer, 2004.
7. D. Boneh, X. Boyen, E.-J. Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In *Eurocrypt'05*, *LNCS* 3494, pp. 440–456. Springer, 2005.
8. D. Boneh, M. Franklin. Identity-based encryption from the Weil pairing. In *Crypto'01*, *LNCS* 2139, pp. 213–229. Springer, 2001.
9. D. Boneh, C. Gentry, B. Lynn, H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In *Eurocrypt'03*, volume 2656 of *LNCS*, pages 416–432. Springer, 2003.
10. D. Boneh, J. Katz. Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. In *CT-RSA'05*, volume 3376 of *LNCS*, pages 87–103. Springer, 2005.

11. D. Boneh, B. Lynn, H. Shacham. Short signatures from the Weil pairing. In *Asiacrypt'01*, volume 2248 of *LNCS*, pages 514–532. Springer, 2002.
12. X. Boyen, Q. Mei, B. Waters. Direct Chosen Ciphertext Security from Identity-Based Techniques. in *ACM CCS'05*, ACM Press, pages 320–329, 2005.
13. R. Canetti, O. Goldreich, S. Halevi. The random oracle methodology, revisited. Journal of the ACM 51(4), 557–594 (2004)
14. R. Canetti, S. Halevi, J. Katz. A forward secure public key encryption scheme. In *Eurocrypt'03*, volume 2656 of *LNCS*, pages 254–271. Springer, 2003.
15. R. Canetti, S. Halevi, J. Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 207–222. Springer, 2004.
16. S. S. Chow, L. C. Kwong Hui, S. M. Yiu, K. P. Chow. Secure Hierarchical Identity Based Signature and Its Application. In *ICICS'04*, volume 3269 of *LNCS*, pages 480–494, Springer, 2004.
17. J. S. Coron, D. Naccache. Boneh *et al.*'s k-Element Aggregate Extraction Assumption Is Equivalent to the Diffie-Hellman Assumption. In *Asiacrypt'03*, volume 2894 of *LNCS*, pages 392–397. Springer, 2003.
18. R. Cramer, V. Shoup, *Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack*, in *SIAM* Journal of Computing 33, pages 167–226, 2003.
19. Y. Dodis, M. Franklin, J. Katz, A. Miyaji, M. Yung. Intrusion-Resilient Public-Key Encryption. In *CT-RSA'03*, volume 2612 of *LNCS*, pages 19–32. Springer, 2003.
20. Y. Dodis, J. Katz, S. Xu, M. Yung. Key-Insulated Public Key Cryptosystems. In *Eurocrypt'02*, volume 2332 of *LNCS*, pages 65–82. Springer, 2002.
21. Y. Dodis, J. Katz, S. Xu, M. Yung. Strong key-insulated signature schemes. In *PKC'03*, volume 2567 of *LNCS*, pages 130–144. Springer, 2003.
22. Y. Dodis, J. Katz. Chosen-Ciphertext Security of Multiple Encryption. In *TCC'05*, volume 3378 of *LNCS*, pages 188–209, Springer, 2005.
23. E. Fujisaki, T. Okamoto. How to enhance the security of public-key encryption at minimum cost. In *PKC'99*, *LNCS* 1560, pp. 53–68. Springer, 1999.
24. C. Gentry, A. Silverberg. Hierarchical ID-based cryptography. In *Asiacrypt'02*, volume 2501 of *LNCS*, pages 548–566. Springer, 2002.
25. R. Granger, N. P. Smart. On Computing Products of Pairings. Cryptology ePrint Archive: Report 2006/172, 2006.
26. G. Hanaoka, Y. Hanaoka, H. Imai. Parallel Key-Insulated Public Key Encryption. In *PKC'06*, volume 3958 of *LNCS*, pages 105–122, Springer, 2006.
27. G. Itkis, L. Reyzin. SiBIR: Signer-Base Intrusion-Resilient Signatures. In *Crypto'02*, volume 2442 of *LNCS*, pages 499–514, Springer, 2002.
28. A. Joux. A One Round Protocol for Tripartite Diffie-Hellman. In *ANTS'00*, volume 1838 of *LNCS*, pages 385–394, Springer, 2000.
29. J. Katz. A Forward-Secure Public-Key Encryption Scheme. Cryptology ePrint Archive: Report 2002/060, 2002.
30. E. Kiltz. On the Limitations of the Spread of an IBE-to-PKE Transformation. In *PKC'06*, *LNCS* 3958, pp. 274–289, Springer, 2006.
31. E. Kiltz. Chosen-Ciphertext Secure Identity-Based Encryption in the Standard Model with short Ciphertexts. Cryptology ePrint Archive: Report 2006/122, 2006.
32. E. Kiltz, D. Galindo. Direct Chosen-Ciphertext Secure Identity-Based Key Encapsulation without Random Oracles. In *ACISP'06*, volume 4058 of *LNCS*, pages 336–347 Springer, 2006.

33. S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, B. Waters. Sequential Aggregate Signatures and Multisignatures Without Random Oracles. In *Eurocrypt'06*, volume 4004 of *LNCS*, pages 465–485, Springer, 2006.
34. K. G. Paterson, J. C. N. Schuldt. Efficient Identity-based Signatures Secure in the Standard Model. In *ACISP'06*, volume 4058 of *LNCS*, pages 207–222, Springer, 2006.
35. V. Shoup. Using Hash Functions as a Hedge against Chosen Ciphertext Attack. In *Eurocrypt'00*, volume 1807 of *LNCS*, pages 275–288, Springer, 2000.
36. A. Shamir. Identity based cryptosystems and signature schemes. In *Crypto'84*, volume 196 of *LNCS*, pages 47–53. Springer, 1984.
37. B. Waters. Efficient Identity-Based Encryption Without Random Oracles. In *Eurocrypt'05*, volume 3494 of *LNCS*, pages 114–127. Springer 2005.
38. D. Yao, N. Fazio, Y. Dodis, A. Lysyanskaya. ID-based encryption for complex hierarchies with applications to forward security and broadcast encryption. In *ACM CCS'04*, ACM Press, pages 354–363, 2004.

A The HHI Construction

The original PKIE system proposed by Hanaoka, Hanaoka and Imai [26] is recalled below. In some sense, it can be thought of as a double application of a key-insulated scheme obtained from the Boneh-Franklin IBE [8] using two distinct Private Key Generators (PKGs) as helpers. To ensure the security in the sense of definition 1, a private key for period i consists of an aggregation of private keys for identities i and $i-1$.

Key generation: given a security paramter $\lambda \in \mathbb{N}$, this algorithm
1. chooses bilinear map groups $(\mathbb{G}, \mathbb{G}_T)$ of order $p > 2^\lambda$, a generator $g \in \mathbb{G}$, hash functions $H : \{0,1\}^* \rightarrow \mathbb{G}$, $G : \mathbb{G}_T \rightarrow \{0,1\}^n$ (modeled as random oracles in the security analysis),
2. picks $\alpha_1, \alpha_2 \xleftarrow{R} \mathbb{Z}_p^*$ and computes $g_1 = g^{\alpha_1}$, $g_2 = g^{\alpha_2}$,
3. computes $u_{-1} = H(-1)$, $u_0 = H(0)$,
4. computes $d_{-1} = u_{-1}^{\alpha_1}$, $d_0 = u_0^{\alpha_2}$,
5. initializes the user's private key to $\mathsf{usk}_0 = d_{-1}d_0 \in \mathbb{G}$,
6. Helpers' private keys are set to $\mathsf{mst}_1 = \alpha_1$ and $\mathsf{mst}_2 = \alpha_2$ while the public key is $\mathsf{pk} := \{p, \mathbb{G}, \mathbb{G}_T, e, g, g_1, g_2, H, G\}$.

Helper-Update: given $\mathsf{mst}_j = \alpha_j$ and a period number $i \in \{1, 2, \ldots, N\}$, helper $j \in \{1, 2\}$
1. returns $\perp$ if $i \neq j \bmod 2$,
2. computes $d_{i-2} = H(i-2)^{\alpha_j}$, $d_i = H(i)^{\alpha_j}$,
3. outputs an update key $\mathsf{hsk}_i = d_i/d_{i-2} \in \mathbb{G}$

User-Update: given usk_{i-1}, hsk_i and i,
1. set $\mathsf{usk}_i = \mathsf{usk}_{i-1} \cdot \mathsf{hsk}_i \in \mathbb{G}$,
3. return usk_i and discard usk_{i-1}, hsk_i.

At time period i, user's private key is always set to

$$\mathsf{usk}_i = d_i d_{i-1} = H(i)^{\alpha_j} H(i-1)^{\alpha_{j-1}} \in \mathbb{G}$$

with $j = 2 - (i \bmod 2)$.

Encrypt: given pk, $i \in \mathbb{N}$, a message $m \in \{0,1\}^n$ is encrypted into

$$\sigma = \big(g^s, m \oplus G(W)\big)$$

for a random $s \overset{R}{\leftarrow} \mathbb{Z}_p^*$ and $W = \big(e(g_j, H(i)) \cdot e(g_{j-1}, H(i-1))\big)^s$ with $j = 2 - (i \bmod 2)$.

Decrypt: given $\sigma = (A, B)$ and usk_i, compute

$$m = B \oplus G(e(A, \mathsf{usk}_i))$$

The above version of the scheme is only secure against chosen-plaintext attacks. The authors of [26] obtain the CCA-security in the random oracle model through the Fujisaki-Okamoto conversion [23].

B Security Proof for the Parallel Key-Insulated KEM

Chosen-ciphertext security is defined as follows for parallel key-insulated KEMs.

Definition 3. *A parallel key-insulated KEM is secure against chosen-ciphertext attacks if no PPT adversary has non-negligible advantage in the following game:*

1. *The challenger $\mathcal{C}$ runs the key generation algorithm, gives* pk *to the adversary $\mathcal{A}$ and keeps* mst_0, mst_1 *and* uks_0 *to itself.*
2. *$\mathcal{A}$ adaptively issues a series of queries which are either:*
   - *Key Exposure queries as in definition 1*
   - *Decapsulation queries $\langle j, \sigma \rangle$: $\mathcal{C}$ responds by generating* usk_j *to run the decapsulation algorithm on σ and pass the result to $\mathcal{A}$.*
3. *When $\mathcal{A}$ is ready to be challenged, she chooses period number $j^\star \in \{1, \ldots, N\}$. The challenger $\mathcal{C}$ runs algorithm* **Encapsulate**$(j^\star, \mathsf{pk})$ *to produce a random key $K^\dagger$ along with its encapsulation $\sigma^\star$. At this point, $\mathcal{C}$ tosses a coin $b^\star \overset{R}{\leftarrow} \{0,1\}$. If $b^\star = 1$, $\mathcal{C}$ defines $K^\star = K^\dagger$. Otherwise, it sets $K^\star \overset{R}{\leftarrow} \mathcal{K}$ as a randomly chosen element from the key space $\mathcal{K}$. The pair $(K^\star, \sigma^\star)$ is sent as a challenge to $\mathcal{A}$.*
4. *$\mathcal{A}$ issues new queries as in stage* 2.
5. *She eventually outputs $b \in \{0,1\}$ and wins if $b = b^\star$ provided similar restrictions to those of definition 1 are respected.*

As shown in [18], a chosen-ciphertext secure KEM immediately gives rise to an IND-CCA2 public key cryptosystem when combined with a suitable symmetric encryption scheme. Although the present setting slightly differs from the traditional public key setting, it is straightforward to extend the proof of theorem 5 in [18] to our context.

The next theorem now states the security of our parallel key-insulated KEM in the sense of definition 3 and under the DBDH assumption.

Theorem 2. *If no algorithm (t, ε)-breaks the DBDH assumption, our parallel key-insulated KEM is $(t', 4N(1-q_d/p)\varepsilon)$-secure against chosen-ciphertext attacks for $t' < t - O(N\tau_{exp} + q_d\tau_p)$ where N is the number of time periods, q_d denotes the number of decapsulation queries and τ_{exp}, τ_p respectively stand for the time complexity of an exponentiation in $\mathbb{G}$ and a pairing evaluation.*

Proof. We outline an algorithm $\mathcal{B}$ receiving as input a tuple (g^a, g^b, g^c, T) randomly sampled from either $D_{bdh} = \{(g^a, g^b, g^c, e(g,g)^{abc}) | a, b, c \xleftarrow{R} \mathbb{Z}_p^*\}$ or $D_{rand} = \{(g^a, g^b, g^c, e(g,g)^z) | a, b, c, z \xleftarrow{R} \mathbb{Z}_p^*\}$ and uses the adversary $\mathcal{A}$ to tell which distribution it was taken from.

The simulator $\mathcal{B}$ generates public key components h, g_1, g_2, g_1', g_2' as in the proof of theorem 1. Namely, it sets $h = g^b$, $g_1 = g^a$, $g_2 = g_1^\gamma$ for some randomly chosen $\gamma \xleftarrow{R} \mathbb{Z}_p^*$ while g_1' and g_2' are chosen to properly handle key exposure queries as in the proof of theorem 1. In addition, $\mathcal{B}$ publishes the description of a collision-resistant hash function $H : \mathbb{N} \to \mathbb{Z}_p^*$ and some injective encoding function $H' : \mathbb{G} \to \mathbb{Z}_p$ (we refer to [12] for details on how to obtain such an encoding). Next, $\mathcal{B}$ computes $\omega^\star = H'(g^c) \in \mathbb{Z}_p$ and defines the group element $g' = (g_1 g_2)^{-\omega^\star} g^{t_3}$ for a random $t_3 \xleftarrow{R} \mathbb{Z}_p^*$. The function F_3 is implicitly defined as $F_3(x) = (g_1 g_2)^{x - \omega^\star} g^{t_3}$. The adversary is started with $\{p, \mathbb{G}, \mathbb{G}_T, g, g_1, g_2, h, g_1', g_2', g', H, H'\}$ as input. She then issues a series of key exposure queries which are handled exactly as in the proof of theorem 1. Other queries are treated as follows.

Decapsulation queries: when $\mathcal{A}$ issues a pair $\langle i, \sigma \rangle$ containing a ciphertext $\sigma = (A, B, C, D)$ and a period i, $\mathcal{B}$ computes $\omega = H'(A) \in \mathbb{Z}_p$. If $\omega = \omega^\star$, it aborts. Assuming that H' is injective, this implies that $A = g^c$. Since the DBDH instance was randomly distributed, such a situation only happens with probability q_d/p throughout all queries. If $\omega \neq \omega^\star$, $\mathcal{B}$ determines whether σ is valid by checking if

$$\frac{e(A, F_j(i))}{e(B, g)} = \frac{e(A, F_{j-1}(i-1))}{e(C, g)} = \frac{e(A, F_3(\omega))}{e(D, g)} = 1_{\mathbb{G}_T}.$$

If the above checking fails, $\mathcal{B}$ returns a random element $K \xleftarrow{R} \mathbb{G}_T$ from the key space. Otherwise, it knows that

$$\sigma = (A, B, C, D) = (g^s, F_j(i)^s, F_{j-1}(i-1)^s, F_3(\omega)^s),$$

where $D = (g_1 g_2)^{s(\omega - \omega^*)} g^{s t_3}$, for some unknown $s \in \mathbb{Z}_p^*$. Algorithm $\mathcal{B}$ then computes $g_1^s g_2^s = \left(D / A^{t_3}\right)^{1/\omega - \omega^*}$ which yields the key $K = e(h, g_1 g_2)^s$.

Challenge: when $\mathcal{A}$ produces her challenge request, the returned ciphertext is

$$\sigma^\star = \left(g^c, (g^c)^{t_1}, (g^c)^{t_2}, (g^c)^{t_3}\right)$$

while the challenge key is $K^\star = T^{1+\gamma}$. As $F_1(\ell) = g^{t_1}$, $F_2(\ell - 1) = g^{t_2}$ (we still assume that ℓ is odd) and $F_3(\omega^\star) = g^{t_3}$, $\sigma^\star$ is a valid encapsulation of $K^\star$ if $T = e(g,g)^{abc}$. If T is random in $\mathbb{G}_T$, so is $K^\star$ and the result follows. $\square$