AAECC (2006) 16: 379–396
DOI 10.1007/s00200-005-0183-y

Mihir Bellare · **Adriana Palacio**

Protecting against key-exposure: strongly key-insulated encryption with optimal threshold

Received: 5 September 2004 / Revised: 26 April 2005 / Published online: 23 December 2005
© Springer-Verlag 2005

Abstract Key-insulated encryption schemes use a combination of key splitting and key evolution to protect against key exposure. Existing schemes, however scale poorly, having cost proportional to the number t of time periods that may be compromised by the adversary, and thus are practical only for small values of t. Yet in practice t might be large.

This paper presents a strongly key-insulated encryption scheme with *optimal threshold*. In our scheme, t need not be known in advance and can be as large as one less than the total number of periods, yet the cost of the scheme is not impacted. This brings key-insulated encryption closer to practice. Our scheme is based on the Boneh-Franklin identity-based encryption (IBE) scheme [9], and exploits algebraic properties of the latter.

Another contribution of this paper is to show that (not strongly) key-insulated encryption with optimal threshold and allowing random-access key updates (which our scheme and all others known allow) is equivalent to a restricted form of IBE. This means that the connection between key-insulated encryption and IBE is not accidental.

Keywords Key exposure · Key update · Encryption · Identity-based encryption

M. Bellare (✉)
Dept. of Computer Science & Engineering, University of California at San Diego,
9500 Gilman Drive, La Jolla, California 92093, USA
E-mail: mihir@cs.ucsd.edu.
http://www-cse.ucsd.edu/ users/mihir. Supported in part by NSF grants CCR-0098123, ANR-0129617 and CCR-0208842, and by an IBM Faculty Partnership Development Award.

A. Palacio
Dept. of Computer Science & Engineering, University of California at San Diego,
9500 Gilman Drive, La Jolla, California 92093, USA E-mail: apalacio@cs.ucsd.edu.
http://www-cse. ucsd.edu/ users/apalacio. Supported in part by an NSF graduate fellowship.

1 Introduction

In practice the most important threat to the security of a public-key encryption scheme is exposure of the decryption key due to compromise of the underlying system. (In recent years we are seeing an increase in the speed, automation and sophistication of intrusion attacks on computer systems. This is coupled with an increase in the frequency of vulnerability reports, making it more difficult for system administrators to keep up to date with patches [13].) This paper provides means to protect against this type of key exposure via the framework of key-insulated encryption [17]. We first provide some background and then discuss our results.

KEY-UPDATING SCHEMES AND THEIR SECURITY PARAMETERS. One approach to protect against the threat of key exposure is to split a decryption key into shares stored on different devices. However, this entails distributing the decryption computation across multiple devices, for example via threshold decryption [20,11], which is not always practical. Another approach, pioneered by [2,5], is to evolve the secret key with time. This can provide forward-security, meaning compromise of the current key does not render usages of previous keys insecure, but compromise still entails that future uses of the key are insecure and the public key must be revoked.

A *key-updating encryption scheme* [17] combines key splitting with key evolution, with the aim of obtaining some of the security benefits of splitting while leaving decryption a stand-alone user operation. Initialization involves providing an auxiliary *helper* (this could be a smartcard or a remote device) with a *master helper key* hsk and the user with a *stage* 0 *user secret key* usk_0. The user's public encryption key pk is treated like that of an ordinary encryption scheme with regard to certification, but its lifetime is divided into stages $i = 1, 2, \ldots, N$, with encryption in stage i performed as a function of pk, i and the plaintext, and decryption in stage i performed by the user using a *stage i user secret key* usk_i that is obtained by the following key-update process performed at the start of stage i: first, the helper sends to the user, over a secure channel, a *stage i helper key* hsk_i computed as a function of hsk and i; second, the user computes usk_i as a function of usk_{i-1} and hsk_i; and third, the user discards (erases) usk_{i-1}. The security intent is that: (1) if the helper is not compromised, user secret keys for more than t different stages must be exposed to compromise ciphertexts encrypted for any other stage, and (2) even if the helper is compromised, the user secret key of at least one stage must be exposed to compromise a ciphertext. The terminology of [17] is that a scheme satisfying (1) is *key insulated with threshold t* while a scheme satisfying both (1) and (2) is *strongly key insulated with threshold t*. (Both of these notions can be considered under either chosen-plaintext or chosen-ciphertext attacks, but we consider only the latter due to the growing consensus that this is the more appropriate in practice [7,36,31,23,35].)

PREVIOUS SCHEMES AND THEIR SCALABILITY. For any given value of the threshold parameter t, Dodis et al. [17] present a strongly key-insulated encryption scheme with threshold t. (They have numerous schemes but the one to which we refer is the only one secure against chosen-ciphertext attacks, and is based on [15].) However it has costs proportional to t. Namely, the public key consists of $3t$ elements in a group whose discrete logarithm problem must be hard, while encryption in stage i requires $t^2 \lg(i)$ group multiplications (plus a few exponentiations). We suggest that this dependence on t represents a lack of scalability and leads to costs that could be prohibitive in practice. Here are some arguments to support this view.

First, the desired security threshold t depends on the particulars of the application, including the frequency of updates and the total number of stages. These parameters may not be known in advance to the scheme designer. Furthermore, they may change with time as the security demands of the application change, in which case usage of a scheme such as the above would require the application to certify a new public key for each such parameter change. Second, a realistic risk assessment leads one to desire security with a large value of t. The reason is that once the user's system is compromised, it is likely to stay compromised through numerous successive stages, until such time as the compromise is discovered, the hole is patched, the intruder is evicted, and the system is rebooted. As an example, suppose the public key is valid for a year and updates are performed once per hour. If we want to give a system a day to recover from compromise, and we want to tolerate 10 different compromises in the year, then t must be at least $10 \cdot 24 = 240$. The size of the public key in the above-mentioned scheme of [17] is then $3 \cdot 240 = 720$ group elements, which is quite prohibitive.

OUR TARGET. We suggest that in order to have a practical realization of key-updating encryption, we should target a strongly key insulated encryption scheme with *optimal threshold*. By this we mean that regardless of the number of user stages that are compromised, ciphertexts intended for any uncompromised stage remain secure. (This is the case where the helper is uncompromised, meaning it replaces condition (1) discussed above. Condition (2) stays the same as before.) This must be true even if the total number of user stages is not known in advance and may depend on the adversary. Notice that a scheme with this property is automatically scalable. There is no threshold parameter in the picture, and since the total number of stages is not fixed, the key sizes and the costs of encryption and decryption will not depend on the threshold or the total number of stages. With such a design, an application can dynamically change its update frequency and yet be able to tolerate compromise of the maximum possible number of user stages.

The next question is how to design such a scheme. We consider using identity-based encryption (IBE) schemes for this purpose as discussed next.

USING IBE. Recall that in an identity-based encryption (IBE) scheme [34], an entity's public key is its identity i, and a trusted authority, holding a master key s, can issue to this entity a secret decryption key s_i computed as a function of s and i. The security attribute is that encryption under the public key of an entity remains secure even in the face of exposure of the secret keys of any number of other entities. Such IBE schemes have been designed [9, 14].

As noted in [17], any IBE scheme can be converted into a key-insulated encryption scheme in the following trivial way: let the master helper key be master key s of the IBE scheme, and let the user's stage i secret key be s_i, which is computed by the helper, using s, and sent to the user, at the start of stage i. This key-insulated scheme has optimal threshold, but as [17] go on to point out, it is not *strongly* key insulated. Indeed, if the helper is compromised the master key s is revealed, and then the adversary can compute the user secret key for any stage. This means there is a single point of failure for the system, exactly what key splitting was supposed to avoid in the first place.

Even though an IBE scheme does not directly yield a strongly key-insulated scheme, we show how to construct strongly key-insulated encryption schemes out of IBE schemes that have certain special properties. In particular we show how to construct one from the Boneh and Franklin [9] IBE scheme.

THE SKIE-OT SCHEME. In this paper we present the first strongly key-insulated scheme with optimal threshold. Our scheme, called SKIE-OT, is based on the secure against chosen-ciphertext attacks version of the Weil-pairing-based Boneh-Franklin [9] identity-based encryption scheme (BF-IBE). It exploits the algebraic structure of the latter to split keys and perform suitable key updates.

The SKIE-OT is as efficient as the underlying BF-IBE scheme. In particular key sizes are quite small, and encryption and decryption cost roughly three exponentiations plus some hashing. Since the scheme has optimal threshold, this is true regardless of the number of stages and the number of stages whose compromise can be tolerated. Accordingly, SKIE-OT is significantly more practical than the previous schemes of [17].

We validate the security of SKIE-OT via proofs showing that SKIE-OT is secure (meaning strongly key insulated with optimal threshold) as long as the underlying BF-IBE scheme is secure (meaning a secure identity-based encryption scheme under chosen-ciphertext attacks as per the definition of [9]). In particular, since Boneh and Franklin have shown that the BF-IBE scheme is secure in the random-oracle model of [6] under the bilinear Diffie-Hellman (bilinear DH) assumption, the same assumptions suffice to guarantee security of SKIE-OT.

SKIE-OT, like all the schemes in [17], allows "random-access key updates." Namely, for any $i \geq 1$ and $j \geq 0$, the user, given usk_j and hsk_i, can compute usk_i in polynomial time. (In particular, it does not need hsk_l for $l \neq i$.)

We remark that our design is simple, based on appropriately combining different known techniques rather than introducing any fundamentally novel technique. (We suggest, however, that the problem itself is nontrivial, and that our ability to provide a simple effective solution at this stage is in large part due to the availability of the powerful tools recently introduced by Boneh and Franklin [9].) However, for practical purposes it is important to note the solution and provide the supporting security analyses.

A SCHEME BASED ON THE COCKS-IBE. Subsequent to [9], Cocks presented an alternative IBE scheme [14]. (The basic version can be proven secure against chosen-plaintext attacks in the random-oracle model assuming hardness of the quadratic residuosity problem, and one can also strengthen the scheme to be secure against chosen-ciphertext attack [19].) The technique underlying SKIE-OT can be applied to build a strongly key-insulated encryption scheme with optimal threshold based on the Cocks-IBE as well. This is made possible by the fact that the Cocks-IBE permits appropriate key splitting. (The key splitting method is mentioned in [14], as pointed out to us by Dan Boneh.)

AN EQUIVALENCE RESULT AND ITS IMPLICATIONS. A second contribution of this paper is a result that helps shed light on the above. We have already seen that any IBE scheme trivially yields a (not strongly) key-insulated encryption scheme with optimal threshold. But perhaps key-insulated encryption is easier than IBE. It turns out that it is not, at least if the key-insulated scheme has the random-access key-update property mentioned above in the context of SKIE-OT. (This property is possessed by all known schemes including ours.) Namely, random-access key update allowing, (not strongly) key-insulated encryption with optimal threshold is equivalent to *restricted-ID* IBE. (This is an IBE scheme in which the identities that an adversary can attack are restricted to some polynomial range specified by the adversary as opposed to being allowed to be any strings.) Not only does one exist if and only if the other exists, but, more pragmatically, we show that either of these objects can be easily transformed into the other. This means that the role played by IBE in our constructions is crucial and not coincidental.

RANDOM ORACLES. The proofs supporting the BF-IBE scheme [9], and thus ultimately supporting SKIE-OT, are in the random-oracle model [6]. The proofs supporting the scheme of [17], not being in the random-oracle model, are providing better security guarantees (cf. [10]). But proofs in the random-oracle model do have significant value in practice (cf. [6]), and one must weigh what one gives up on provable guarantees against the practical benefits of the new schemes, which are considerable.

Recently, IBE schemes with proofs of security in the standard (i.e. not random-oracle) model have appeared [8, 37]. However, they are secure only against chosen-plaintext attack, not against chosen-ciphertext attack. There are standard transforms that might make them secure against chosen-ciphertext attack [32], but these are very expensive. Thus at the moment it does not appear that one can do implement (strongly) key-insulated encryption with optimal threshold without random oracles.

TOWARDS PRACTICE. The broad question of whether key-updating encryption could be practical can be viewed as having two parts. One is to find effective cryptographic realizations. The other is to investigate the practicality of the model and concept, independently of the cryptographic realization. Our work has addressed only the first part. It would be naive to think that this alone makes key-updating encryption practical, but it is a step towards this end. Given the recognized importance of the key-exposure problem, we feel that the research community should endeavor to assess the potential of new ideas to address it.

As to whether the concept as a whole is practical, it seems too early to tell. Many of the important system level questions related to the model have yet to be seriously addressed. As a final contribution of this paper, we point to some of the important issues in Appendix B.

RELATED WORK. Intrusion-resilient encryption is an extension of key-insulated encryption in which the helper key also evolves with time. This makes it possible to achieve stronger security properties. However, existing intrusion-resilient encryption schemes [16] are more complex and less efficient than our strongly key-insulated encryption scheme. Furthermore they suffer from a lack of scalability analogous to that discussed above in that the number of stages must be known in advance, and the sizes of keys depend (although logarithmically) on this number. Our construction thus remains of practical interest as being a simple and efficient way to protect keys against compromise.

This paper has concentrated on public-key encryption. Designing key-insulated signature schemes is simple in comparison [18]. This is not surprising and reflects a general phenomenon, namely that signatures have been easier to achieve than encryption in the types of models we are considering. For example, numerous forward-secure signature schemes are known [2,5,1,24,27,28,26] but forward-secure encryption remained open until recently, when solutions exploiting the pairing-based techniques of [9] were provided by [12]. Similarly, intrusion-resilient signature schemes were designed in [25], but intrusion-resilient encryption had to await, and build on, the same pairing-based techniques [16].

Subsequent to the first exposure of our work [4], Hanaoka, Hanaoka, Shikata and Imai [22] suggested a simple way to create a strongly key-insulated encryption scheme with optimal threshold based on an IBE scheme and a standard public-key encryption scheme. However, their scheme is only secure against chosen-plaintext attack, not against chosen-ciphertext attack like our scheme.

2 Definitions

We let $\mathbb{N} = \{1, 2, \dots\}$ be the set of positive integers, and if $N \in \mathbb{N}$ then we let $[N] = \{1, \dots, N\}$. The notation $x \overset{R}{\leftarrow} S$ denotes that x is selected randomly from set S. If A is a possibly randomized algorithm then the notation $x \overset{R}{\leftarrow} A(a_1, a_2, \dots)$ denotes that x is assigned the outcome of the experiment of running A on inputs $a_1, a_2, \dots$.

2.1 Key-updating encryption schemes and their security

This follows [17], which in turn extended [5]. A *key-updating encryption scheme* $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$ is specified by five polynomial-time algorithms whose functionality is as follows:

- The randomized *key-generation algorithm* KG takes input security parameter k and returns (pk, usk_0, hsk) where pk is the user public key, usk_0 is the *stage 0 user secret key*, and hsk is the *master helper key*. The user is initialized with pk, usk_0 while the helper is initialized with pk, hsk.
- At the start of stage $i \geq 1$, the helper applies the *helper key-update algorithm* HKU to i, pk, hsk to obtain a *stage i helper key* hsk_i, which is then assumed to be conveyed to the user via a secure channel.
- At the start of stage $i \geq 1$, the user receives hsk_i from the helper and then applies the *user key-update algorithm* UKU to i, pk, hsk_i, usk_{i-1} to obtain the *stage i user secret key* usk_i. The user then discards (erases) usk_{i-1}.
- Anyone can apply the randomized *encryption algorithm* Enc to a stage number i, the user public key pk and message $M \in \{0, 1\}^*$ to obtain a ciphertext C intended for the user to decrypt in stage i.
- In stage i the user can apply the *decryption algorithm* Dec to i, pk, its stage i secret key usk_i, and a ciphertext C to obtain either a message M or the special symbol $\perp$ indicating failure. We require that if C was produced by applying the encryption algorithm to i, pk, M then $\mathsf{Dec}(i, pk, usk_i, C) = M$.

Next we formalize the notion of a key-updating scheme being (strongly) key insulated with optimal threshold. This is based on the ideas of [17] but we introduce some simplifications. For readers familiar with [17], Appendix A shows that the simplifications do not weaken the security requirements. Security considers two types of attacks, namely attacks on the user and attacks on the helper. In both cases we consider chosen-ciphertext attacks.

ATTACKS ON THE USER. The formalization of security for the user requires a strong form of privacy, namely indistinguishability as per [21,30], in the face of key-exposure and chosen-ciphertext attacks. To define it we consider the following experiment related to key-updating encryption scheme $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$, adversary A and security parameter k. The key-generation algorithm KG is run on input k to produce (pk, usk_0, hsk). Adversary A gets input pk and returns an integer $N \in \mathbb{N}$ specified in unary. A challenge bit b is chosen at random, and the execution of A is continued with A now being provided the following oracles:

- A decryption oracle $\mathsf{Dec}(i, pk, usk_i, \cdot)$ for each user stage $i = 1, \ldots, N$. This models a chosen-ciphertext attack.
- A *key-exposure oracle* $\mathsf{Exp}(\cdot, pk, usk_0, hsk)$ which the adversary can query with any value $i \in [N]$ of its choice to get back the stage i user secret key usk_i and the stage i helper key hsk_i. This models the ability of the adversary to compromise any user stage of its choice. (We make the conservative assumption that when an adversary has compromised the user in stage i it not only obtains usk_i but has compromised the channel between user and helper and thus also gets hsk_i.)
- A *left-or-right oracle* $\mathsf{Enc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$ [3] which given $j \in [N]$ and equal length messages M_0, M_1 returns a *challenge ciphertext* $C \overset{R}{\leftarrow} \mathsf{Enc}(j, pk, M_b)$.

The adversary may query these oracles adaptively, in any order it wants, subject only to the restriction that it make exactly one query to the left-or-right oracle. Let j denote the stage number of this query and let C denote the ciphertext returned by the left-or-right oracle in response to this query. Eventually, A outputs a guess bit d and halts. It is said to win if $d = b$, ciphertext C was not queried to $\mathsf{Dec}(j, pk, usk_j, \cdot)$ after it was returned by the left-or-right oracle, and j was not queried to the key-exposure oracle. The adversary's advantage is the probability that it wins minus $1/2$, and the key-updating scheme KUS is said to be *key insulated with optimal threshold* if the advantage of any polynomial-time adversary is negligible.

We stress that the number of stages N is a random variable depending on the adversary, and that there is no upper bound on the number of user stages that the adversary is allowed to corrupt. This is in contrast to [17] where the total number of stages N, and the maximum number t of corrupted stages, are parameters of the scheme fixed in advance. One implication of our strengthened requirement is scalability. (This is directly implied by our definition and does not have to be a separate requirement.)

ATTACKS ON THE HELPER. Adversary A, given pk, is assumed to have compromised the helper and thus be in possession of the master helper key hsk. The security requirement is that, as long as none of the user stages is compromised, ciphertexts intended for any user stage remain secure. The formalization follows the one above.

We consider the following experiment related to key-updating encryption scheme $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$, adversary A and security parameter k. The key-generation algorithm KG is run on input k to produce (pk, usk_0, hsk). Adversary A gets input pk, hsk, and returns an integer $N \in \mathbb{N}$ specified in unary. A challenge bit b is chosen at random, and the execution of A is continued with A now being provided the decryption oracles and a left-or-right oracle as above. (But it is *not* provided a key-exposure oracle.) The adversary may query these oracles adaptively, in any order it wants, subject only to the restriction that it make exactly one query to the left-or-right oracle. Let j denote the stage number of this query and let C denote the ciphertext returned by the left-or-right oracle in response to this query. Eventually, A outputs a guess bit d and halts. It is said to win if $d = b$ and ciphertext C was not queried to $\mathsf{Dec}(j, pk, usk_j, \cdot)$ after it was returned by the left-or-right oracle. The adversary's advantage is the probability that it wins minus $1/2$, and the key-updating scheme KUS is said to be *secure against attacks on the helper* if the advantage of any polynomial-time adversary is negligible. The scheme is *strongly key insulated with optimal threshold* if it is key insulated with optimal threshold and also secure against attacks on the helper.

For both types of attacks, we adopt the convention that the *time complexity* of an adversary A is the execution time of the experiment used to define the advantage of A, including the time taken for key generation and initializations, and the time taken by the oracles to compute replies to the adversary's queries. This convention simplifies concrete security considerations.

2.2 Identity-based encryption schemes

IBE SCHEMES. This follows [33,9]. An IBE scheme $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBKI}, \mathsf{IBEnc}, \mathsf{IBDec})$ is specified by four polynomial-time algorithms whose functionality is as follows:

- The *key-generation algorithm* IBKG takes input security parameter k and returns a pair (pk, s) consisting of a *parameter list* pk and a *master key* s.
- Given a user identity $i \in \mathbb{N}$, the trusted center can apply the *decryption-key issuance algorithm* IBKI to pk, s, i to obtain a decryption key $ibsk_i$ that, along with pk, is then sent to user i over a secure channel.
- The *encryption algorithm* IBEnc takes input an *identity* $i \in \mathbb{N}$, the parameter list pk, and a message $M \in \{0, 1\}^*$ and returns a ciphertext c.
- A user holding the secret key $ibsk_i$ can apply the (deterministic) *decryption algorithm* IBDec to its identity i, the parameter list pk, the secret key $ibsk_i$ and ciphertext c to recover the message M.

SECURITY OF AN IBE SCHEME. We consider the following experiment related to IBE scheme $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBKI}, \mathsf{IBEnc}, \mathsf{IBDec})$, adversary A and security parameter k. The key-generation algorithm IBKG is run on input k to produce (pk, s). Adversary A gets input pk and returns an integer $N \in \mathbb{N}$ specified in unary. A challenge bit b is chosen at random, and the execution of A is continued with A now being provided the following oracles:

- Decryption oracles $\mathsf{IBDec}(i, pk, ibsk_i, \cdot)$ for all $i = 1, \dots, N$
- A *key-exposure oracle* $\mathsf{Exp}(\cdot, pk, s)$ that when queried with $i \in [N]$ returns the decryption key $ibsk_i = \mathsf{IBKI}(pk, s, i)$ of user i. This models the ability of the adversary to compromise any user of its choice.

- A *left-or-right oracle* $\mathsf{IBEnc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$ which given $j \in [N]$ and equal length messages M_0, M_1 returns a *challenge ciphertext* $c \overset{R}{\leftarrow} \mathsf{IBEnc}(j, pk, M_b)$.

The adversary may query these oracles adaptively, in any order it wants, subject only to the restriction that it make exactly one query to the left-or-right oracle. Let j denote the user identity of this query and let c denote the ciphertext returned by the left-or-right oracle in response to this query. Eventually, A outputs a guess bit d and halts. It is said to win if $d = b$, ciphertext c was not queried to $\mathsf{IBDec}(j, pk, ibsk_j, \cdot)$ after it was returned by the left-or-right oracle, and j was not queried to the key-exposure oracle. The adversary's advantage is the probability that it wins minus $1/2$. The IBE scheme IBES is said to be *secure against chosen-ciphertext attacks* if the advantage of any polynomial-time adversary is negligible.

REMARK. The formalization of security given above differs from that of [9]. Above, the identities that an adversary can query (to its key-exposure or left-or-right oracle) are restricted to integers in the range $1, \ldots, N$, meaning to a polynomial range specified by the adversary. (Since N must be specified in unary, it cannot exceed the running time of the adversary, which is polynomial in the security parameter.) In [9], the adversary can query any identity in $\mathbb{N}$. (Subject of course to being able to write it down, which effectively means identities are restricted to integers in a range $1, \ldots, 2^{\mathrm{poly}(k)}$.) This restriction is important to one direction of Theorem 4.1. (Namely we do not know whether a key-insulated encryption scheme with optimal threshold implies an IBE scheme meeting the stronger notion of [9].) However our weaker notion of security (which is of course met by the BF-IBE scheme) suffices for Corollary 3.3 and the other direction of Theorem 4.1.

3 The SKIE-OT scheme

Our strongly key-insulated scheme with optimal threshold is based on the Boneh-Franklin (BF) identity-based encryption (IBE) scheme and exploits some algebraic properties of the latter. In order to avoid taking the reader through the full BF-IBE scheme, we begin by presenting a simplified abstraction of it in which we detail only a few items that are necessary for our transformation and treat the rest as "black boxes." We then show how to build on this to construct SKIE-OT. This section concludes with our result stating that SKIE-OT is strongly key insulated with optimal threshold, assuming the BF scheme is a secure IBE scheme under chosen-ciphertext attacks.

WHAT BF SUPPLIES. The BF-IBE scheme is specified by a tuple of algorithms $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBKI}, \mathsf{IBEnc}, \mathsf{IBDec})$, where

- The *key-generation* algorithm IBKG takes input security parameter k and returns a pair (pk, s) consisting of a *parameter list* $pk = (q, \mathbb{G}, H, \ldots)$ and a *master key* $s \in \mathbb{Z}_q^*$, where q is a prime number, $\mathbb{G}$ is (the description of) an additive (cyclic) group of order q, and $H\colon \mathbb{N} \to \mathbb{G}^*$ is a hash function whose range is the nonzero elements of the group. The ellipsis marks indicate that the parameter list pk contains a few other parameters, but for our purpose it does not matter what they are, so we do not detail them.[1]

- The deterministic *decryption-key issuance algorithm* IBKI takes input an *identity* i which could be an arbitrary integer, the parameter list pk and the master key s, and returns a decryption key $ibsk_i = s \cdot H(i) \in \mathbb{G}$ (this denotes the group element $H(i)$ added to itself s times via the group operation) that, along with pk, is then sent to user i over a secure channel.
- The randomized *encryption algorithm* IBEnc takes input an identity i, the parameter list pk, and a message $M \in \{0, 1\}^*$ and returns a ciphertext c.[2]
- A user holding the secret key $ibsk_i$ can apply the *decryption algorithm* IBDec to its identity i, the parameter list pk, the secret key $ibsk_i$ and ciphertext c to recover the message M.

DISCUSSION OF THE BF-IBE SCHEME. The identity i functions as the public key of the entity having this identity. In the BF-IBE scheme, the secret key $ibsk_i = s \cdot H(i)$ is computed by a trusted party who holds the master key s, and then given by this party to entity i. The details of how encryption and decryption are performed in the IBE scheme are not important for us. What we will exploit is the fact that the secret key $ibsk_i$ is computed as a linear function of the master key s, and that the scheme meets the notion of privacy against chosen-ciphertext attacks recalled in Section 2.2. Under this notion, an adversary gets to compromise some number of entities of its choice and obtain their secret keys, and yet it remains computationally infeasible to obtain the secret key of any uncompromised entity, or even to obtain partial information about messages encrypted under that key, all this being under a chosen-ciphertext attack. It is shown in [9] that this security is achieved in the random-oracle model under the bilinear DH assumption.

OUR SKIE-OT SCHEME. The component algorithms of our key-updating scheme, KUS = (KG, HKU, UKU, Enc, Dec), are depicted in Figure 1. Here we briefly explain the ideas.

We recall that a key-updating encryption scheme that is key insulated with optimal threshold, but not strongly key insulated, can be trivially obtained from any IBE scheme, as indicated in [17]. The public key of a user is a parameter list $pk = (q, \mathbb{G}, H, \dots)$ for the IBE scheme. The master helper key is the master key s of the IBE scheme. View the stage number i as an identity for the IBE scheme. The user secret key in stage i is $ibsk_i = s \cdot H(i)$, the secret key corresponding to entity i in the IBE scheme. Encryption is then performed as a function of i, pk as per the IBE scheme except that we additionally include the value of i in the ciphertext. Decryption in stage i uses $s \cdot H(i)$ as the secret key to run the decryption algorithm of the IBE scheme.

The weakness of the above scheme is that if the helper is compromised, then the attacker obtains s and the security of all user stages is compromised. We address this as follows. In our scheme, s is not held by the helper, but rather split into

[1] For a reader familiar with [9], we remark that the quantities include a prime number p such that $p = 6q - 1$, a generator of $\mathbb{G}$, and some more hash functions. $\mathbb{G}$ is the group of points on an elliptic curve over a field of order p.

[2] The basic version of the BF-IBE scheme only allows encryption of plaintext messages of a specific length which is a parameter of the scheme, but via standard hybrid encryption techniques we may extend the message space so that strings of any length may be encrypted. For simplicity we assume this is done here.

```
Algorithm $\mathsf{KG}(k)$
    $(pk, s) \overset{R}{\leftarrow} \mathsf{IBKG}(k)$
    Parse $pk$ as $(q, \mathbb{G}, H, \dots)$
    $usk \overset{R}{\leftarrow} \mathbb{Z}_q$ ; $hsk \leftarrow (s - usk) \bmod q$
    $ibsk_0 \leftarrow \mathsf{IBKI}(0, pk, s)$ ; $usk_0 \leftarrow (usk, ibsk_0)$
    Return $(pk, usk_0, hsk)$
```

```
Algorithm $\mathsf{UKU}(i, pk, hsk_i, usk_{i-1})$
    Parse $pk$ as $(q, \mathbb{G}, H, \dots)$
    Parse $usk_{i-1}$ as $(usk, ibsk_{i-1})$
    $ibsk_i \leftarrow usk \cdot H(i) + hsk_i$ in $\mathbb{G}$
    $usk_i \leftarrow (usk, ibsk_i)$
    Return $usk_i$
```

```
Algorithm $\mathsf{HKU}(i, pk, hsk)$
    Parse $pk$ as $(q, \mathbb{G}, H, \dots)$
    $hsk_i \leftarrow hsk \cdot H(i)$ in $\mathbb{G}$
    Return $hsk_i$
```

```
Algorithm $\mathsf{Enc}(i, pk, M)$
    $c \overset{R}{\leftarrow} \mathsf{IBEnc}(i, pk, M)$
    $C \leftarrow (i, c)$
    Return $C$
```

```
Algorithm $\mathsf{Dec}(i, pk, usk_i, C)$
    Parse $C$ as $(j, c)$
    If $j \neq i$ then return $\perp$
    Parse $usk_i$ as $(usk, ibsk_i)$
    $M \leftarrow \mathsf{IBDec}(i, pk, ibsk_i, c)$
    Return $M$
```

Fig. 1 The component algorithms of our SKIE-OT scheme $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$, based on the algorithms $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBEnc}, \mathsf{IBDec})$ describing the Boneh-Franklin IBE scheme

shares via a one-out-of-two secret-sharing scheme, with one share held by the user and the other by the helper. That is, $s \equiv usk + hsk (\text{mod } q)$, where the stage i user secret key is $usk_i = (usk, ibsk_i)$ with $ibsk_i = (usk + hsk) \cdot H(i)$, and the master helper key is hsk. Update of the user secret key must be performed without reconstructing s, since otherwise an adversary compromising the user at update time could obtain s and thus compromise all stages. We perform update without reconstruction of s by exploiting the fact that for any i, the map $x \mapsto x \cdot H(i)$ is a homomorphism from the additive group $\mathbb{Z}_q$ to the additive group $\mathbb{G}$. At the start of stage i, the helper uses hsk to compute $hsk_i = hsk \cdot H(i)$ and sends it to the user. The latter, holding $usk_{i-1} = (usk, ibsk_{i-1})$, sets $ibsk_i = usk \cdot H(i) + hsk_i$ in $\mathbb{G}$. By the homomorphic property we have

$$\begin{aligned} usk \cdot H(i) + hsk_i &= usk \cdot H(i) + hsk \cdot H(i) \\ &= (usk + hsk) \cdot H(i) \\ &= ibsk_i \ . \end{aligned}$$

The user sets its updated secret key to $usk_i = (usk, ibsk_i)$ and erases usk_{i-1}.

KEY SIZES AND COSTS. The public key in SKIE-OT (which is the parameter list of the BF-IBE scheme) consists of two k-bit primes p, q, where k is the security parameter and $p = 6q - 1$, and two elements of $\mathbb{G}$ where the latter is an elliptic curve group. In addition, the scheme has several associated public hash functions. The sizes of the master helper key, the user secret key for any stage, and the helper key for any stage are all $O(k)$. Encryption in stage i involves performing encryption as per the BF-IBE scheme which requires two exponentiations, four hash function applications and one Weil-paring computation [9]. Decryption requires one exponentiation, three hash function applications and one Weil-paring computation. As observed in [9], the Weil paring can be computed efficiently using an algorithm due to Miller [29] whose running time is cubic.

SECURITY OF SKIE-OT. The following two lemmas show that the advantage of any adversary against the SKIE-OT scheme, performing an attack on the user in the first case, and an attack on the helper in the second, can be upper bounded by the advantage of a related adversary against the BF-IBE scheme.

Lemma 3.1 *Let A be an adversary of time complexity T against SKIE-OT, attacking the user. Assume that the adversary compromises t user stages. Then there exists an adversary B performing a chosen-ciphertext attack against the underlying BF-IBE scheme with at least the same advantage. Furthermore, the time complexity of B is T and the number of entities compromised by B during its attack is t.* □

Proof of Lemma 3.1 Let $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$ be the SKIE-OT scheme and $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBEnc}, \mathsf{IBDec})$ be the BF-IBE scheme. We construct an adversary B that uses A to perform a chosen-ciphertext attack against IBES. Fix $k \in \mathbb{N}$. The experiment that defines the advantage of B begins by running $\mathsf{IBKG}(k)$ to produce (pk, s). On input $pk = (q, \mathbb{G}, H, \dots)$, adversary B randomly selects an element $usk \in \mathbb{Z}_q$. It then runs A on input pk until A outputs $N \in \mathbb{N}$, which B also returns. B is given access to decryption oracles $\mathsf{IBDec}(i, pk, ibsk_i, \cdot)$ for $i = 1, \dots, N$, a key-exposure oracle $\mathsf{Exp}(\cdot, pk, s)$, and a left-or-right oracle $\mathsf{IBEnc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$, where the challenge bit b was chosen at random. The adversary's goal is to guess b.

When the execution of B proceeds, it continues to run A and uses its oracles to respond to A's queries. In response to a query (j, c) to the decryption oracle $\mathsf{Dec}(i, pk, usk_i, \cdot)$, where $j \neq i$, B returns $\perp$. In response to a query (j, c) to the decryption oracle $\mathsf{Dec}(j, pk, usk_j, \cdot)$, B forwards the query to its decryption oracle $\mathsf{IBDec}(j, pk, ibsk_j, \cdot)$ and returns the answer M to A. By the definition of algorithm Dec, in both cases, the answer is exactly what A's decryption oracle would have returned. In response to a query i to the key-exposure oracle $\mathsf{Exp}(\cdot, pk, usk_0, hsk)$, B makes the query i to its key-exposure oracle $\mathsf{Exp}(\cdot, pk, s)$, obtaining the decryption key $ibsk_i = s \cdot H(i)$. B then sets $usk_i \leftarrow (usk, ibsk_i)$ and $hsk_i \leftarrow ibsk_i - usk \cdot H(i)$ in $\mathbb{G}$, and returns usk_i as the stage i user secret key and hsk_i as the stage i helper key to A. Since usk was chosen at random, $hsk_i = (s - usk) \cdot H(i)$, and $ibsk_i = usk \cdot H(i) + hsk_i$, A's view is identical to its view in the attack against KUS. In response to A's query j, M_0, M_1 to the left-or-right oracle $\mathsf{Enc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$, B forwards the query to its left-or-right oracle $\mathsf{IBEnc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$, obtaining a ciphertext c. It then sets $C \leftarrow (j, c)$ and returns this to A. By the definition of algorithm Enc, the answer is exactly what A's left-or-right oracle would have returned. When A outputs its guess bit d and halts, B returns d and halts.

Since B simulates A's environment in its attack against KUS perfectly, A behaves as it does there and B wins as long as A does. By our conventions for measuring time complexity, the time complexity of B is T. Furthermore, B makes the same number of queries to its key-exposure oracle, compromising that number of entities, as user stages A compromises by querying its key-exposure oracle. The conclusion follows. □

Lemma 3.2 *Let A be an adversary of time complexity T against SKIE-OT, attacking the helper. Then there exists an adversary B performing a chosen-ciphertext attack against the underlying BF-IBE scheme with at least the same advantage.*

Furthermore, the time complexity of B is T and this adversary does not compromise any entities during its attack. □

Proof of Lemma 3.2 Let $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$ be the SKIE-OT scheme and $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBEnc}, \mathsf{IBDec})$ be the BF-IBE scheme. We show how to construct an adversary B that runs A as a subroutine and performs a chosen-ciphertext attack against IBES. Fix $k \in \mathbb{N}$. The experiment that defines the advantage of B begins by running $\mathsf{IBKG}(k)$ to produce (pk, s). Adversary B is given input $pk = (q, \mathbb{G}, H, \dots)$. In order to simulate A's environment in its attack against KUS, B must provide A with a master helper key corresponding to the public key pk. To do so, it selects an element $hsk \in \mathbb{Z}_q$ at random. It runs A on input pk, hsk until A outputs $N \in \mathbb{N}$, which B also returns. B is then given access to decryption oracles $\mathsf{IBDec}(i, pk, ibsk_i, \cdot)$ for $i = 1, \dots, N$, a key-exposure oracle $\mathsf{Exp}(\cdot, pk, s)$, and a left-or-right oracle $\mathsf{IBEnc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$, where the challenge bit b was chosen at random. The adversary's goal is to guess b.

When the execution of B proceeds, it continues to run A and uses its oracles to respond to A's queries. In response to a query (j, c) to the decryption oracle $\mathsf{Dec}(i, pk, usk_i, \cdot)$, where $j \neq i$, B returns $\perp$. In response to a query (j, c) to the decryption oracle $\mathsf{Dec}(j, pk, usk_j, \cdot)$, B forwards the query to its decryption oracle $\mathsf{IBDec}(j, pk, ibsk_j, \cdot)$ and returns the answer M to A. By the definition of algorithm Dec, in both cases, the answer is exactly what A's decryption oracle would have returned. In response to A's query j, M_0, M_1 to the left-or-right oracle $\mathsf{Enc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$, B forwards the query to its left-or-right oracle $\mathsf{IBEnc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$, obtaining a ciphertext c. It then sets $C \leftarrow (j, c)$ and returns this to A. By the definition of algorithm Enc, the answer is exactly what A's left-or-right oracle would have returned. When A outputs its guess bit d and halts, B returns d and halts.

It is easy to see that by the way hsk is chosen and the way B responds to A's oracle queries, A's view is identical to its view in the attack against KUS. Since the simulation is perfect, A behaves as it does there and B wins as long as A does. Our conventions for measuring time complexity imply that the time complexity of B is T. Furthermore, B does not make any queries to its key-exposure oracle, i.e., it does not compromise any entities during its attack. The conclusion follows. □

From these lemmas, the following security result for our SKIE-OT scheme follows directly.

Theorem 3.3 *If the BF-IBE scheme is secure against chosen-ciphertext attacks then the key-updating scheme SKIE-OT is strongly key insulated with optimal threshold.* □

As a result, SKIE-OT is secure (in the random oracle model) under the same assumptions used in [9] to prove the BF-IBE scheme secure.

4 Equivalence result

Let $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$ be a key-updating scheme. Having obtained pk, usk_0, hsk by running KG on input k, we know that the user secret keys for stages $l = 1, \ldots, j$ can be computed based on the associated stage helper keys as follows:

$$\text{For } l = 1, \ldots, j \text{ do:} \quad hsk_l \leftarrow \mathsf{HKU}(l, pk, hsk);$$
$$usk_l \leftarrow \mathsf{UKU}(l, pk, hsk_l, usk_{l-1}) \ .$$

We say that key-updating scheme KUS allows *random-access key updates* if there is a polynomial-time *random-access user-key-update algorithm* RUKU which takes input i, j, pk, hsk_i, usk_j and outputs usk_i for any $i \geq 1$ and $j \geq 0$.[3] This is useful for error recovery. Also, it allows the user to maintain its decryption capability for ciphertexts from the past, despite having to erase the secret key for one stage at the start of the next. It is easy to see that SKIE-OT allows random-access key updates, as do all the schemes in [17].

Our result is that a (not strongly) key-insulated encryption scheme with optimal threshold that allows random-access key updates is essentially the same thing as an identity-based encryption scheme, in that either of these objects can be easily turned into the other. The following states it more formally. The theorem is true both for chosen-plaintext attacks and chosen-ciphertext attacks, although our formalization only refers to the latter.

Theorem 4.1 *There exists a secure identity-based encryption scheme if and only if there exists a key-insulated encryption scheme with optimal threshold that allows random-access key updates.* □

Proof of Theorem 4.1 The proof is constructive, showing how either object is easily transformed into the other.

First assume $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBKI}, \mathsf{IBEnc}, \mathsf{IBDec})$ is an IBE scheme, specified according to the format of Section 2.2, and meeting the notion of security specified there.

We construct from it the trivial key-updating scheme that we have discussed often before. It is easy to see that this is a key-insulated scheme with optimal threshold that allows random-access key updates. The novel direction is the converse.

For the converse, assume $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$ is a key-insulated encryption scheme with optimal threshold that allows random-access key updates, and let RUKU denote the random-access user key-update algorithm. We now design an IBE scheme $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBKI}, \mathsf{IBEnc}, \mathsf{IBDec})$. The constituent algorithms are depicted in Figure 2. The idea is that the master secret key of the trusted party in the IBE scheme contains both the stage 0 user secret key usk_0 and the helper master key hsk. The entity with identity i is identified with stage i of the user. The trusted authority wants to issue usk_i to user i as its secret decryption key. In the absence of extra properties, the trusted authority could compute usk_i by starting from usk_0, hsk and computing $usk_1, \ldots, usk_i$ in turn via the user key update and helper key update algorithms. This, however, takes time polynomial in i, which is not polynomial time. (The trusted authority of the

[3] This a somewhat stronger requirement than the one made in [17], who replace hsk_i as input to RUKU with a value $hsk_{i,j}$ computed by the helper based on another algorithm that takes inputs i, j, pk, hsk. We have preferred to simplify the definition to require just one algorithm, but the change makes no difference to any results. All known schemes, both ours and theirs, meet both definitions, and Theorem 4.1 is true for both definitions.

```
Algorithm $\mathsf{IBKG}(k)$
    $(pk, usk_0, hsk) \overset{R}{\leftarrow} \mathsf{KG}(k)$
    $s \leftarrow (usk_0, hsk)$
    Return $(pk, s)$
```

```
Algorithm $\mathsf{IBKI}(pk, s, i)$
    Parse $s$ as $(usk_0, hsk)$
    $hsk_i \leftarrow \mathsf{HKU}(i, pk, hsk)$
    $ibsk_i \leftarrow \mathsf{RUKU}(i, 0, pk, hsk_i, usk_0)$
    Return $ibsk_i$
```

```
Algorithm $\mathsf{IBEnc}(i, pk, M)$
    $c \leftarrow \mathsf{Enc}(i, pk, M)$
    Return $c$
```

```
Algorithm $\mathsf{IBDec}(i, pk, ibsk_i, c)$
    $M \leftarrow \mathsf{Dec}(i, pk, ibsk_i, c)$
    Return $M$
```

Fig. 2 The component algorithms of IBE scheme $\mathsf{IBES} = (\mathsf{IBKG}, \mathsf{IBKI}, \mathsf{IBEnc}, \mathsf{IBDec})$ constructed from the given key-insulated encryption scheme $\mathsf{KUS} = (\mathsf{KG}, \mathsf{HKU}, \mathsf{UKU}, \mathsf{Enc}, \mathsf{Dec})$ and its random-access user key-update algorithm RUKU

IBE scheme must issue $ibsk_i$ to i in time polynomial in $\lg(i)$ and k where k is the security parameter.) This problem is solved via the assumption that the key-updating scheme allows random-access key updates. The trusted authority can issue a decryption key to i by using the random-access key-update algorithms to directly compute $ibsk_i = usk_i$ given usk_0, hsk as shown in Figure 2. The encryption and decryption algorithms are unchanged.

Finally, we have to argue that our constructed IBE scheme is secure under the assumption that the key-updating scheme is key insulated with optimal threshold. This is easy, however, and details are omitted. □

Acknowledgements We thank Dan Boneh for his communication regarding a strongly key-insulated encryption scheme with optimal threshold based on Cock's IBE scheme.

A On the notions of security for key-updating schemes

TYPES OF ATTACKS ON THE USER. In our formulation of attacks on the user presented in Section 2, an adversary compromising stage i obtains not only the stage i user secret key usk_i but also the stage i helper key hsk_i. We consider this to be appropriate because in practice if user stage i is compromised then not only is usk_i exposed, but one should assume the channel from helper to user is compromised for the duration of that stage as well, and thus any communication over it, including hsk_i, should be assumed to be available to the adversary. This issue is recognized, but handled a little differently, in [17], who separate what we call attacks on the user into "key-exposure attacks," in which an adversary compromising stage i obtains usk_i, and "key-update attacks," in which the same adversary obtains hsk_i. We have lumped the two together both for simplicity and because of our contention that consideration of security against key exposure without security against key update is impractical.

Note it is assumed that as part of the process of discovering and ejecting intruders that leads us to consider the possibility of secure stages at some point after compromise, the secure channel, over which the helper key for each stage is communicated, is re-established as well.

Dodis et. al. [17] formalize security against key-update attacks by requiring that the information sent by the helper to the user in stage i be simulatable from the point of view of an adversary that has compromised stage i. Instead, we have simply packaged it into the same framework as key-exposure attacks, asking that an adversary obtaining the information in question still be unable to compromise encryption in un-compromised stages. The requirement of [17] is stronger, but it is hard to see why one should require it rather than just require the appropriate and natural end-goal of user security as we have done. In any cases all known schemes, both ours and theirs, meet their stronger requirement. For these reasons, coupled with a desire for simplicity, we did not require simulatability in the face of key-update attacks as part of our definition.

ONE CHALLENGE BIT VERSUS MANY. The formalization of security against attacks on the user given in [17] provides the adversary with a left-or-right oracle [3]

$$\mathsf{Enc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, \mathbf{b})) \text{ where } \mathbf{b} = (\mathbf{b}[1], \ldots, \mathbf{b}[N]) \in \{0, 1\}^N$$

and N is the total number of stages. A query has the form j, M_0, M_1 where $j \in [N]$ and M_0 are equal-length messages, and in response the oracle returns $C \overset{R}{\leftarrow} \mathsf{Enc}(j, pk, M_{\mathbf{b}[j]})$. On the other hand, our formalization provides the adversary with a left-or-right oracle $\mathsf{Enc}(\cdot, pk, \mathrm{LR}(\cdot, \cdot, b))$ where $b \in \{0, 1\}$. In response to query j, M_0, M_1 as above, it returns $C \overset{R}{\leftarrow} \mathsf{Enc}(j, pk, M_b)$, but only a single query is allowed to the oracle. While our formulation is simpler, one might think the resulting security requirement is weaker. In fact, the two notions of security are equivalent in the sense that a key-updating scheme is secure against attacks on the user under the definition of [17] if and only if it is secure against attacks on the user under our definition. This can be proved via a standard hybrid argument.

B Implementation and system issues

There are numerous issues that would need to be considered with regard to implementing a key-updating system. These issues are in some sense orthogonal to our paper since they are about the model and concept of [17]. We do not have answers to these questions, but we feel it is important for the future to at least raise them.

Obvious issues are the practicality of a two-device setup, and the practicality of dividing the lifetime of a key into stages, which implies that the person encrypting will have to be aware of the current stage number.

An issue that we believe is tricky is the security of the channel from the helper to the user. The keys sent by the helper to the user *cannot* be sent in the clear. The very definition of key-updating encryption implies that this is insecure, because then if the adversary has corrupted just one user stage and not the helper, it can use the helper stage keys to compute user secret keys for all subsequent stages by applying the key-update algorithms. The formal models reflect this by not giving the adversary the helper keys for uncompromised stages, which indicates they are assumed to be sent over a secure channel. The question that we feel needs to be pursued is how this assumption can be implemented. There might be settings where a secure channel from helper to user is possible. (This might be the case when the helper is a smartcard. Another interesting setting is that the user is a cellphone and the helper is its charging device [22].) But if the helper is simply some remote device, the channel may have to be implemented cryptographically. In that case, when a user compromise is discovered, the channel should be assumed to be compromised as well, and a secure channel must be re-established. How this may be done is not clear. One possibility is to distribute new keys to the parties, but that does not seem very practical.

References

1. Abdalla, M., Reyzin L.: A new forward-secure digital signature scheme. *Advances in Cryptology – ASIACRYPT '00*, Lecture Notes in Computer Science Vol. 1976, T. Okamoto ed., Springer-Verlag, 2000
2. Anderson, R.: Two Remarks on Public-Key Cryptology. Manuscript, 2000, and Invited Lecture at the Fourth Annual Conference on Computer and Communications Security, Zurich, Switzerland, April 1997
3. Bellare, M., Desai, A., Jokipii, E., Rogaway E.: A concrete security treatment of symmetric encryption: Analysis of the DES modes of operation. *Proceedings of the 38 th Symposium on Foundations of Computer Science*, IEEE, 1997
4. Bellare, M., Palacio, A.: Protecting against key exposure: Strongly key-insulated encryption with optimal threshold. Cryptology ePrint Archive: Report 2002/064. `http://eprint.iacr.org/2002/064`
5. Bellare, M., Miner, S.: A forward-secure digital signature scheme. *Advances in Cryptology – CRYPTO '99*, Lecture Notes in Computer Science Vol. 1666 , M. Wiener ed., Springer-Verlag, 1999
6. Bellare, M., Rogaway, P.: Random oracles are practical: A paradigm for designing efficient protocols. In: Proceedings of the 1st Annual Conference on Computer and Communications Security, ACM, 1993
7. Bleichenbacher, D.: A chosen ciphertext attack against protocols based on the RSA encryption standard PKCS #1. *Advances in Cryptology – CRYPTO '98*, Lecture Notes in Computer Science Vol. 1462 , H. Krawczyk ed., Springer-Verlag, 1998
8. Boneh, D., Boyen, X.: Secure identity-based encryption without random oracles. *Advances in Cryptology – CRYPTO '04*, Lecture Notes in Computer Science Vol. 3152 , M. Franklin ed., Springer-Verlag, 2004
9. Boneh, D., Franklin M.: Identity-based encryption from the Weil pairing. SIAM J. Comput. **32**(3), 586–615, (2003). Preliminary version in *Advances in Cryptology – CRYPTO '01*, Lecture Notes in Computer Science Vol. 2139 , J. Kilian ed., Springer-Verlag, 2001
10. Canetti, R., Goldreich, O., Halevi, S.: The random oracle methodology revisited. *Proceedings of the 30 th Annual Symposium on the Theory of Computing*, ACM, 1998
11. Canetti, R., Goldwasser S.: An efficient threshold public-key cryptosystem secure against adaptive chosen-ciphertext attack. *Advances in Cryptology – EUROCRYPT '99*, Lecture Notes in Computer Science Vol. 1592 , J. Stern ed., Springer-Verlag, 1999
12. Canetti, R., Halevi, S., Katz, J.: A Forward-Secure Public-Key Encryption Scheme. *Advances in Cryptology – EUROCRYPT '03*, Lecture Notes in Computer Science Vol. 2656 , E. Biham ed., Springer-Verlag, 2003
13. CERT Coordination Center: Overview of attack trends. April 8, 2002. `http://www.cert.org/`
14. Cocks, C.: An identity based encryption based on quadratic residues. *Cryptography and Coding*, Lecture Notes in Computer Science Vol. 2260, Springer-Verlag, 2001
15. Cramer, R., Shoup V.: A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. *Advances in Cryptology – CRYPTO '98*, Lecture Notes in Computer Science Vol. 1462 , H. Krawczyk ed., Springer-Verlag, 1998
16. Dodis, Y., Franklin, M., Katz, J., Miyaji, A., Yung, M.: Intrusion-Resilient Public-Key Encryption. *Topics in Cryptology – CT-RSA '03*, Lecture Notes in Computer Science Vol. 2612 , M. Joye ed., Springer-Verlag, 2003
17. Dodis, Y., Katz, J., Xu, S. Yung, M.: Key-Insulated Public Key Cryptosystems. *Advances in Cryptology – EUROCRYPT '02*, Lecture Notes in Computer Science Vol. 2332 , L. Knudsen ed., Springer-Verlag, 2002

18. Dodis, Y., Katz, J., Xu, S., Yung, M.: Strong Key-Insulated Signature Schemes. *Public-Key Cryptography '03*, Lecture Notes in Computer Science Vol. 2567 , Y. Desmdedt ed., Springer-Verlag, 2003
19. Fujisaki, E., Okamoto, T.: Secure integration of asymmetric and symmetric encryption schemes. *Advances in Cryptology – CRYPTO '99*, Lecture Notes in Computer Science Vol. 1666 , M. Wiener ed., Springer-Verlag, 1999
20. Gennaro, R., Shoup, V.: Securing threshold cryptosystems against chosen-ciphertext attack. *Advances in Cryptology – EUROCRYPT '98*, Lecture Notes in Computer Science Vol. 1403 , K. Nyberg ed., Springer-Verlag, 1998
21. Goldwasser, S., Micali S.: Probabilistic Encryption. J. Comput. Syst. Sci. **28**, 270–299 (1984)
22. Hanaoka, Y., Hanaoka, G., Shikata, J., Imai, H.: Identity-based encryption with non-interactive key update. Cryptology ePrint Archive: Report 2004/338. `http://eprint.iacr.org/2004/338`
23. IEEE.: IEEE P1363: Standard Specifications For Public Key Cryptography. `http://grouper.ieee.org/groups/1363/P1363/`
24. Itkis, G., Reyzin, L.: Forward-secure signatures with optimal signing and verifying. *Advances in Cryptology – CRYPTO '01*, Lecture Notes in Computer Science Vol. 2139 , J. Kilian ed., Springer-Verlag, 2001
25. Itkis, G., Reyzin L.: SiBIR: Signer-Base Intrusion-Resilient Signatures. *Advances in Cryptology – CRYPTO '02*, Lecture Notes in Computer Science Vol. 2442 , M. Yung ed., Springer-Verlag, 2002
26. Kozlov, A., Reyzin, L.: Forward-Secure Signatures with Fast Key Update. In: Cimato, S., Galdi, C., Persiano, G., (eds) Third International Conference on Security in Communication Networks (SCN '02), Lecture Notes in Computer Science Vol. 2576, Springer-Verlag, 2003
27. Krawczyk, H.: Simple forward-secure signatures from any signature scheme. In: Proceedings of the 7th Annual Conference on Computer and Communications Security, ACM, 2000
28. Malkin, T., Micciancio, D., Miner, S.: Efficient generic forward-secure signatures with an unbounded number of time periods. *Advances in Cryptology – EUROCRYPT '02*, Lecture Notes in Computer Science Vol. 2332 , L. Knudsen ed., Springer-Verlag, 2002
29. Miller, V.: Short programs for functions on curves. Unpublished manuscript, 1986
30. Rackoff, C., Simon, D.: Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. *Advances in Cryptology – CRYPTO '91*, Lecture Notes in Computer Science Vol. 576 , J. Feigenbaum ed., Springer-Verlag, 1991
31. RSA Laboratories. PKCS #1 – RSA Cryptography Standard. `http://www.rsasecu\-rity.com/rsalabs/pkcs/pkcs-1/index.html`
32. Sahai, A.: Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. *Proceedings of the 40*
th Symposium on Foundations of Computer Science, IEEE, 1999
33. Shamir, A.: How to share a secret. Communications of the ACM, **22**, 612–613 (1979)
34. Shamir, A.: Identity-based cryptosystems and signature schemes. *Advances in Cryptology – CRYPTO '84*, Lecture Notes in Computer Science Vol. 196, R. Blakely ed., Springer-Verlag, 1984
35. Shoup, V.: A Proposal for an ISO Standard for Public Key Encryption. Cryptology eprint archive Report 2001/112, Dec 2001. `http://eprint.iacr.org/2001/112/`
36. Shoup,V.: Why chosen ciphertext security matters. IBM Research Report RZ 3076, November, 1998. `http://www.shoup.net`
37. Waters, B.: Efficient identity-based encryption without random oracles. In: Cramer, R. (ed) Advances in Cryptology – EUROCRYPT '05, Lecture Notes in Computer Science, Springer-Verlag, 2005