Strong Key-Insulated Signature Schemes

Yevgeniy Dodis[1], Jonathan Katz[2], Shouhuai Xu[3], and Moti Yung[4]

[1] Department of Computer Science, New York University, USA. dodis@cs.nyu.edu
[2] Department of Computer Science, University of Maryland (College Park), USA. Work done in part while at DIMACS. jkatz@cs.umd.edu
[3] Department of Information and Computer Science, University of California at Irvine, USA. shxu@ics.uci.edu
[4] Department of Computer Science, Columbia University, USA. moti@cs.columbia.edu

Abstract. Signature computation is frequently performed on insecure devices — e.g., mobile phones — operating in an environment where the private (signing) key is likely to be exposed. Strong key-insulated signature schemes are one way to mitigate the damage done when this occurs. In the key-insulated model [6], the secret key stored on an insecure device is refreshed at discrete time periods via interaction with a physically-secure device which stores a "master key". All signing is still done by the insecure device, and the public key remains fixed throughout the lifetime of the protocol. In a strong (t, N)-key-insulated scheme, an adversary who compromises the insecure device and obtains secret keys for up to t periods is unable to forge signatures for any of the remaining $N - t$ periods. Furthermore, the physically-secure device (or an adversary who compromises only this device) is unable to forge signatures for *any* time period.

We present here constructions of strong key-insulated signature schemes based on a variety of assumptions. First, we demonstrate a generic construction of a strong $(N-1, N)$-key-insulated signature scheme using any standard signature scheme. We then give a construction of a strong (t, N)-signature scheme whose security may be based on the discrete logarithm assumption in the random oracle model. This construction offers faster signing and verification than the generic construction, at the expense of $O(t)$ key update time and key length. Finally, we construct strong $(N-1, N)$-key-insulated schemes based on any "trapdoor signature scheme" (a notion we introduce here); our resulting construction in fact serves as an identity-based signature scheme as well. This leads to very efficient solutions based on, e.g., the RSA assumption in the random oracle model.

1 Introduction

Security of cryptographic primitives typically relies on the assumption that secret keys are kept "perfectly secure"; standard cryptosystems provide no security guarantees in case secret keys are ever exposed. In practice, however, the assumption that keys are never exposed is often unwarranted. In many cases, it is easier to obtain a secret key from a stolen device (or by otherwise tricking an unsuspecting user) than to break the computational assumption on which the security of the system is based. Clearly, methods of effectively dealing with key exposure are needed.

A recently-proposed method of minimizing the damage caused by secret key exposures is that of *key-insulated cryptography* [6]. In this model, physical security (and hence secrecy of stored data) is guaranteed for a single device that holds a "master" secret key SK^* corresponding to a fixed public key PK. Day-to-day cryptographic operations, however, are performed by an insecure device which "refreshes" its key periodically by interacting with the secure device. In a (t, N)-key-insulated cryptosystem (informally) an adversary who compromises the insecure device and obtains keys for up to t time periods is unable to violate the security of the cryptosystem for any of the remaining $N - t$ periods; we elaborate for the specific case of digital signatures below. In a *strong* key-insulated scheme, security is additionally guaranteed with respect to the secure device itself or compromises thereof; this is vital when the secure device may be untrusted. Strong key-insulated public-key encryption schemes have been defined and constructed recently [6]; here, we provide definitions and constructions for strong key-insulated signature schemes.

OVERVIEW OF THE MODEL. We review the informal description of the key-insulated model as given in [6], adapted here for the case of digital signatures. As in a standard signature scheme, the user begins by registering a single public key PK; this key will remain fixed for the lifetime of the protocol. A master secret key SK^*, generated along with PK, is stored on a device which is physically secure and hence resistant to compromise. All signing, however, is done by the user on an insecure device for which key exposures may occur. The lifetime of the protocol is divided into distinct periods $1, \ldots, N$; at the beginning of period i, the user interacts with the secure device to derive a temporary secret key SK_i which will be used by the insecure device to sign messages during that period. Signatures are labeled with the time period during which they were generated; thus, signing message M during period i results in signature $\langle i, s \rangle$.

As mentioned above, the user's insecure device is assumed to be vulnerable to repeated key exposures; specifically, we assume that up to $t < N$ periods can be compromised. Our goal is to minimize the effect such compromises will have. Of course, when a key SK_i is exposed an adversary will be able to sign messages of his choice for period i. Our notion of security is that this is the best an adversary can do. In particular, the adversary will be unable to forge a signature on a new message for any of the remaining $N - t$ periods. We call a scheme satisfying this notion (t, N)-*key-insulated*.

If the physically-secure device is completely trusted, it may generate (PK, SK^*) itself, keep SK^*, and publish PK on behalf of the user. When the user requests a key for period i, the device may compute SK_i and send it. More involved methods are needed when the physically-secure device is *not* trusted by the user. In this, more difficult case (which we consider here), a solution is to have the user generate (PK, SK), publish PK, and then derive keys SK^*, SK_0. The user then sends SK^* to the device and stores SK_0 himself on the insecure device. When the user wants to update his key to that of period j (and the user currently holds the key for period i) the physically-secure device computes and sends "partial" key $SK'_{i,j}$ to the user, who may then compute the "actual" key SK_j using SK_i and $SK'_{i,j}$. If designed appropriately, the user's security may be guaranteed during *all* time periods with respect to the device itself. Schemes meeting this level of security are termed *strong*. As noted previously [6], strong key-insulation is essential when a single device serves many different users. Here, users may trust the device to update their keys but may not want the device to be able to sign on their behalf.

Clearly, some form of authentication between the user and the physically-secure device is necessary. Note that if a key K used for this authentication is stored on the insecure device, an adversary who exposes keys even *once* obtains K and can then impersonate the user during subsequent key updates (thus obtaining signing keys for subsequent time periods). As in previous work, however, we assume that authentication is handled by an underlying protocol (outside the scope of this work) which is immune to such attacks. As one possible example, K might never be stored on the insecure device but will instead be obtained directly from the user each time authentication is needed (e.g., K may be a password or a key derived from biometric information).

OUR CONTRIBUTIONS. The initial work on key-insulated cryptosystems [6] focused primarily on the case of public-key encryption; here, we focus on the complementary case of digital signatures. Adapting a "folklore" result (see [1]), we first show a generic construction of a strong $(N-1, N)$-key-insulated signature scheme from any standard signature scheme. We then give a more efficient strong (t, N)-key-insulated signature scheme whose security may be reduced to the discrete logarithm assumption in the random oracle model. This construction offers faster signing and verification than the generic construction, at the expense of $O(t)$ key update time and key length. Finally, we construct strong $(N-1, N)$-key-insulated signature schemes based on any "trapdoor signature scheme" (a term we introduce here); this results in very efficient solutions based on, e.g., the RSA assumption in the random oracle model. Our construction (which may be viewed as a generalization of recent independent work [4, 14, 24, 25]) may also be used as an identity-based signature scheme; we believe this is of independent interest since no rigorous proofs of security for ID-based signature schemes were previously known.

RELATED WORK. Girault [10] investigates a notion similar to key-insulated digital signatures in the context of smart-card research. However, this preliminary work contained no formal model or proofs of security. Key-insulated public-key encryption was considered by Tzeng and Tzeng [30] and also by Lu and Shieh [19], but these works only consider security against a weak, non-adaptive adversary. Key-insulated public-key encryption was first formally defined in [6], and schemes with rigorous proofs of security are given there.

Somewhat related to key insulation is the problem of signature delegation [11]. In this model, a user wants to delegate use of a signing key in a particular way. For example (to place it in our setting), a user may delegate the right to sign messages for a single day. Here, one seeks to prevent exposure of the "master" signing key when a small number of delegated keys are exposed. On the other hand, to prevent excessive delegation it is required that exposure of many delegated keys completely reveals the master key. The key-insulated model makes no such requirement, and this allows for greater efficiency and flexibility. We also note that the existing practical delegation schemes [11] are not provably-secure against an *adaptive* adversary who chooses which keys to expose at any point during its execution. Finally, our *strong* schemes also protect against forgeries by the physically-secure device itself; this has no counterpart in the context of signature delegation.

Besides the key-insulated model, many alternate approaches have been proposed to address the risks associated with key exposure. The first such example is that of forward security [2, 3]. In this model no external device is present, and the entire secret key is stored on — and updated by — the insecure device itself. Clearly, any exposure now compromises *all future* time periods; forward-secure signature schemes, however, prevent compromise of *prior* time periods. A consequence of this model (which is not present in our model) is that even the valid user is unable to recover keys for prior time periods once the appropriate secret key has been erased.

More recently — and subsequent to the present work — the key-insulated model has been extended and strengthened to yield the notion of intrusion-resilience [16]. This model adds to our notion a proactive refresh capability which may be performed more frequently than key updates; hence, intrusion-resilient schemes can tolerate multiple corruptions of both the user and the physically secure device (called the "home base") while maintaining security of the scheme for all time periods during which the user's device remained uncorrupted. Furthermore, if both user and home base are corrupted simultaneously, the scheme remains secure for all prior time periods (as in forward-secure schemes).

Each of these models may be appropriate for use in different contexts. Forward-secure schemes are advantageous in that the user need not interact with any other device. On the other hand, when interaction with such a device may be assumed, key-insulated and intrusion-resilient schemes provide a stronger level of security. Finally, although intrusion-resilience represents a stronger level of security than key-insulation, the assumption of physically-secure storage confers other benefits in the key-insulated model. For example, the key-insulated model enables the honest user to request "old" keys thereby allowing, e.g., the signing of documents for prior time periods when needed. This is impossible in the forward-secure or intrusion-resilient settings. Also, known intrusion-resilient schemes [16, 15] are (thus far) less efficient than the key-insulated schemes presented here, suggesting that one use the latter when physical security of the "home base" can be guaranteed.

Independent of the present work, we have become aware of related work in the context of re-keyed digital signatures [1]. Recasting this work in our model, one may observe that they construct $(N-1, N)$-key-insulated signature schemes based on either (1) generic signature schemes or (2) the factoring assumption. Their generic construction is essentially identical to ours except that we additionally ensure that our scheme is *strong* key-insulated. Our discrete logarithm scheme has no counterpart in [1]. Our scheme based on trapdoor signatures and specialized for RSA may be viewed as the "Guillou-Quisquater" [13] analogue to their "Ong-Schnorr" [23] factoring-based scheme, where again we additionally ensure strong security of our construction. The notion of random access to keys is unique to our treatment.

Finally, we mention that an identity-based signature scheme [28] immediately gives an $(N-1, N)$-key-insulated signature scheme (ensuring *strong* security requires some additional work). However, we are not aware of any previous formal definitions or proofs of security for identity-based signature schemes. Our proof of security for the construction of Section 5 (which is based on, e.g., [13]) may be easily adapted to show that this construction is in fact an identity-based signature scheme as well.

2 The Model

For completeness, we provide a formal definition of key-insulated signature schemes and their security (based on [6]). We begin with the definition of a key-updating signature scheme, which generalizes the notion of a key-evolving signature scheme [3]. In a key-updating signature scheme there is some data (namely, SK^*) that is never erased; this data will be stored on a physically-secure device and hence is never exposed.

Definition 1. *A key-updating signature scheme is a 5-tuple of poly-time algorithms* $(\mathsf{Gen}, \mathsf{Upd}^*, \mathsf{Upd}, \mathsf{Sign}, \mathsf{Vrfy})$ *such that:*

- Gen*, the* key generation algorithm*, is a probabilistic algorithm taking as input a security parameter* 1^k *and the total number of time periods* N*. It returns a public key* PK*, a master key* SK^**, and an initial key* SK_0*.*
- Upd^**, the* device key-update algorithm*, is a probabilistic algorithm taking as input indices* i, j *for time periods (throughout, we assume* $1 \leq i, j \leq N$*) and the master key* SK^**. It returns a partial secret key* $SK'_{i,j}$*.*
- Upd*, the* user key-update algorithm*, is a deterministic algorithm taking as input indices* i, j*, a secret key* SK_i*, and a partial secret key* $SK'_{i,j}$*. It returns the secret key* SK_j *for time period* j*.*
- Sign*, the* signing algorithm*, is a probabilistic algorithm taking as input an index* i *of a time period, a message* M*, and a secret key* SK_i*.* $\mathsf{Sign}_{SK_i}(i, M)$ *returns a signature* $\langle i, s\rangle$ *consisting of the time period* i *and a signature* s*.*
- Vrfy*, the* verification algorithm*, is a deterministic algorithm taking as input the public key* PK*, a message* M*, and a pair* $\langle i, s\rangle$*.* $\mathsf{Vrfy}_{PK}(M, \langle i, s\rangle)$ *returns a bit* b*, where* $b = 1$ *means the signature is accepted.*

If $\mathsf{Vrfy}_{PK}(M, \langle i, s\rangle) = 1$*, we say that* $\langle i, s\rangle$ *is a* valid *signature of* M *for period* i*. We require that all signatures output by* $\mathsf{Sign}_{SK_i}(i, M)$ *are accepted as valid by* Vrfy*.*

In a key-updating signature scheme, a user begins by generating $(PK, SK^*, SK_0) \leftarrow \mathsf{Gen}(1^k, N)$, registering PK in a central location (just as he would for a standard public-key scheme), storing SK^* on a physically-secure device, and storing SK_0 himself. When the user — who currently holds SK_i — wants to obtain SK_j, the user requests $SK'_{i,j} \leftarrow \mathsf{Upd}^*(i, j, SK^*)$ from the secure device. Using SK_i and $SK'_{i,j}$, the user computes $SK_j = \mathsf{Upd}(i, j, SK_i, SK'_{i,j})$; this key may be then used to sign messages during time period j without further access to the device. After computation of SK_j, the user erases SK_i and $SK'_{i,j}$. Note that verification is always performed with respect to a fixed public key PK which is never changed.

Remark 1. The above definition corresponds to schemes supporting random-access key updates [6]; that is, schemes in which one can update SK_i to SK_j in one "step" for any i, j. A weaker definition allows $j = i + 1$ only. All schemes presented in this paper support random-access key updates.

BASIC KEY INSULATION. The adversary we consider is extremely powerful: (1) it may request signatures on messages of its choice during time periods of its choice, adaptively and in any order (i.e., we do not restrict the adversary to making its queries in chronological order); (2) it may expose the secrets contained on the insecure device for up to t adaptively-chosen time periods (alternately, it may choose to expose the secrets stored on the physically-secure device); and (3) it can compromise the insecure device during a key-update phase, thus obtaining partial keys in addition to full-fledged secret keys. The adversary is considered successful if it can forge a valid signature $\langle i, s\rangle$ on message M such that the adversary never requested a signature on M for period i and furthermore the adversary never exposed the insecure device at time period i.

We model each of these attacks by defining appropriate oracles to which the adversary is given access. To model key exposures, we define a *key exposure oracle* $\mathsf{Exp}_{SK^*,SK_0}(\cdot)$ that does the following on input i: (1) The oracle first checks whether period i has been "activated"; if so, the oracle returns the value already stored for SK_i. Otherwise, (2) the oracle runs $SK'_i \leftarrow \mathsf{Upd}^*(0, i, SK^*)$ followed by $SK_i = \mathsf{Upd}(0, i, SK_0, SK'_i)$, returns and stores the value SK_i, and labels period i as "activated". We also give the adversary access to a *signing oracle* $\mathsf{Sign}_{SK^*,SK_0}(\cdot,\cdot)$ that does the following on input i, M: (1) The oracle first checks whether period i has been "activated"; if so, the oracle returns $\mathsf{Sign}_{SK_i}(i, M)$ (where a value for SK_i is already stored). Otherwise, (2) the oracle runs $SK'_i \leftarrow \mathsf{Upd}^*(0, i, SK^*)$ followed by $SK_i = \mathsf{Upd}(0, i, SK_0, SK'_i)$, stores SK_i, returns $\mathsf{Sign}_{SK_i}(i, M)$, and labels period i as "activated".

Remark 2. Storing the values of the secret keys for "activated" periods is only necessary when Upd^* is probabilistic; when it is deterministic, the oracle may simply run Upd^* "from scratch" whenever needed to answer an oracle query. To be fully general, one could allow the adversary to access a "re-issuing oracle" which on input i re-computes the secret key SK_i via $SK_i \leftarrow \mathsf{Upd}(0, i, SK_0, \mathsf{Upd}^*(0, i, SK^*))$. The schemes presented here all remain secure under a more complex definition of this form.

Definition 2. *Let Π be a key-updating signature scheme and fix t. For any adversary A, we may perform the following experiment:*

$$(PK, SK^*, SK_0) \leftarrow \mathsf{Gen}(1^k, N);\ (M, \langle i, s\rangle) \leftarrow A^{\mathsf{Sign}_{SK^*,SK_0}(\cdot,\cdot),\mathsf{Exp}_{SK^*,SK_0}(\cdot)}(PK).$$

We say that A succeeds if $\mathsf{Vrfy}_{PK}(M, \langle i, s\rangle) = 1$, (i, M) was never submitted to the signing oracle, i was never submitted to the key exposure oracle, and A made at most t calls to the key-exposure oracle. Denote the probability of A's success by $\mathsf{Succ}_{A,\Pi}(k)$. We say that Π is (t, N)-key-insulated *if for any* PPT *A, $\mathsf{Succ}_{A,\Pi}(k)$ is negligible. We say Π is* perfectly *key-insulated if Π is $(N-1, N)$-key-insulated.*

We remark that we allow the adversary to interleave signing requests and key exposure requests, and in particular the key exposure requests of the adversary may be made adaptively (based on the entire transcript of the adversary's execution) and in any order.

SECURE KEY UPDATES. For the purposes of meeting Definition 2, we could let $SK'_{i,j} = SK^*$ for all i, j; the user could then run Upd^* and Upd by himself to derive SK_i (and then erase SK^*). Of course, one reason for not doing so is the realistic concern that an adversary who gains access to the insecure device is likely to have access for several consecutive time periods (i.e., until the user detects or re-boots) including the *key update steps*. In this case, an adversary attacking the scheme above would obtain SK^* and we would not be able to achieve even $(1, N)$-key-insulated security.

To address this problem, we consider attacks in which an adversary compromises the user's storage while a key is being updated from SK_i to SK_j; we call this a *key-update exposure at* (i, j). When this occurs, the adversary receives $SK_i, SK'_{i,j}$, and SK_j (actually, the latter can be computed from the former). We say a scheme has *secure key updates* if a key-update exposure at (i, j) is of no more help to the adversary than key exposures at both periods i and j. More formally:

Definition 3. *A key-updating signature scheme Π has* secure key updates *if the view of any adversary A making a key-update exposure at (i, j) can be perfectly simulated by an adversary A' making key exposure requests at periods i and j.*

STRONG KEY INSULATION. Finally, we address attacks that compromise the physically-secure device (this includes attacks by the device itself, in case it is untrusted). Our definition is similar to Definition 2 except that instead of having access to the key exposure oracle, the adversary is simply given the master key SK^*. Schemes which are secure in this sense — and also (t, N)-key-insulated — are termed *strong* (t, N)-key-insulated. We do not protect against adversaries who compromise *both* the physically-secure device and the user's storage; in our model, this is impossible to achieve. (Intrusion-resilient schemes [16] partially protect against such attacks by allowing the insecure device to interact with the secure device even when not updating its key.)

Definition 4. *Let $\Pi = (\mathsf{Gen}, \mathsf{Upd}, \mathsf{Upd}^*, \mathsf{Sign}, \mathsf{Vrfy})$ be a signature scheme which is (t, N)-key-insulated. For adversary B, we perform the following experiment:*

$$(PK, SK^*, SK_0) \leftarrow \mathsf{Gen}(1^k, N);\ (M, \langle i, s\rangle) \leftarrow B^{\mathsf{Sign}_{SK^*, SK_0}(\cdot,\cdot)}(PK, SK^*).$$

We say that B succeeds if $\mathsf{Vrfy}_{PK}(M, \langle i, s\rangle) = 1$ and (i, M) was never submitted to the signing oracle. Denote the probability of B's success by $\mathsf{Succ}_{B,\Pi}(k)$. We say that Π is strong (t, N)-key-insulated *if for any* PPT *B, $\mathsf{Succ}_{B,\Pi}(k)$ is negligible.*

3 Generic, Perfectly Key-Insulated Signature Scheme

We demonstrate a perfectly key-insulated signature scheme that can be constructed from any existentially unforgable (standard) signature scheme $\Theta = (G, S, V)$. Rather than repeating the standard definition of security, we may view Θ as as $(0, 1)$-key-insulated scheme in the natural way. Thus, our construction can be viewed as amplifying a $(0, 1)$-key-insulated scheme to a perfectly key-insulated scheme for larger N. We later show how to achieve strong key insulation with minimal additional cost.

The basic construction achieving perfect $(N - 1, N)$-key-insulation is folklore. Gen generates a pair of keys $(PK, SK^*) \leftarrow G(1^k)$, sets the public key to PK, sets $SK_0 = \perp$, and stores SK^* on the physically-secure device. At the beginning of time period i, the device generates a fresh pair of keys $(pk_i, sk_i) \leftarrow G(1^k)$ and certifies pk_i for time period i by signing it as follows: $\mathsf{cert}_i = (pk_i,\ S_{SK^*}(pk_i \| i))$. It then sets $SK_i = \langle sk_i, \mathsf{cert}_i\rangle$ and sends SK_i to the user, who erases the previous key. The user signs a message M at time period i by using the "temporary" key sk_i and appending the certificate cert_i; that is, $\mathsf{Sign}_{SK_i}(i, M) = \langle i, \sigma, \mathsf{cert}_i\rangle$, where $\sigma \leftarrow S_{sk_i}(M)$. To verify, one first verifies correctness of the cerificate and then uses the period verification key pk_i to verify the signature σ, accepting only if both are valid. We remark that it is crucial to sign the time period i along with pk_i since this prevents an adversary from re-using the same certificate at a different time.

Signing requires computation equivalent to the original (basic) signature scheme, while the cost of signature verification is increased by a factor of two. In practice, verifying the validity of cert_i need only be done once per period when multiple signatures are verified. Security of the scheme is given by the following lemma.

Lemma 1. *If Θ is existentially unforgeable under a chosen message attack, then Π as described is $(N-1, N)$-key-insulated. Furthermore, Π has secure key updates.*

Proof. That Π has secure key updates is trivial. We therefore focus on the proof of perfect key insulation. Let A attack Π. A forgery occurs when the adversary forges a valid signature $\langle i, \sigma, (pk, \tau)\rangle$ of some message M at time period i such that: (1) τ is a vaild signature of $(pk\|i)$ w.r.t. PK; (2) σ is a valid signature of M w.r.t. pk; (3) period i was not exposed; and (4) (i, M) was not submitted to the signing oracle.

Denote the event of a forgery by F. Wlog, we assume that the period i is "activated" (cf. Section 2), so that the value of pk_i is well defined. We let Eq be the event that $pk = pk_i$. Clearly, $\Pr(\mathsf{F}) = \Pr(\mathsf{F} \wedge \mathsf{Eq}) + \Pr(\mathsf{F} \wedge \overline{\mathsf{Eq}})$.

Case 1: In case events F and Eq both occur then $pk = pk_i$. Assume that A makes at most $q(k) = \mathsf{poly}(k)$ queries to the signing oracle overall. We construct A' attacking Θ as follows: A' has as input a verification key pk' for which it does not know the corresponding secret key sk', and also has oracle access to the signing oracle $S_{sk'}(\cdot)$. A' chooses a random index $r \in \{1, \ldots, q(k)\}$, generates a random key pair $(PK, SK^*) \leftarrow G(1^k)$, and runs A on input PK. Let i^* be the period for which the r^{th} signing query of A was made. If a previous signing query was made for that same period i^*, the experiment is aborted. Otherwise, adversary A' implicitly uses (pk', sk') to respond to the query by making use of its signing oracle $S_{sk'}(\cdot)$. For signature queries $r+1, \ldots, q(k)$, if A requests a signature for period i^* the signature is computed using $S_{sk'}(\cdot)$. Additionally, if A ever makes a key exposure query for period i^*, the experiment is aborted. All other oracle queries are answered by A' in the expected manner; namely, by generating fresh temporary keys and using the corresponding secret keys to answer signing and key exposure requests. If the final output of A is $(M, \langle i, \sigma, (pk, \tau)\rangle)$ and the experiment was never aborted, then A' simply outputs (M, σ).

With probability at least $1/q(k)$, the experiment is not aborted and $i^* = i$ (recall, i is the period for which a forgery is made). The success probability of A' in forging a signature for Θ is thus at least $\Pr[\mathsf{F} \wedge \mathsf{Eq}]/q(k)$. By the assumed security of Θ, this quantity must be negligible. Since $q(k)$ is polynomial in k, it must be that $\Pr[\mathsf{F} \wedge \mathsf{Eq}]$ is negligible as well.

Case 2: In case events F and $\overline{\mathsf{Eq}}$ both occur, then either period i is not "activated" or else $pk \neq pk_i$. We construct A' attacking Θ as follows: A' has as input a verification key pk' for which it does not know the corresponding secret key sk', and has access to a signing oracle $S_{sk'}(\cdot)$. A' sets $PK = pk'$ and implicitly sets the master key $SK^* = sk'$. A' then simulates the entire run of A by generating (on its own) all the temporary keys as needed, and using its signing oracle $S_{sk'}(\cdot)$ to produce the needed certificates. If the final output of A is $(M, \langle i, \sigma, (pk, \tau)\rangle)$ then A' simply outputs $(pk\|i, \tau)$. The success probability of A' in forging a signature for Θ is then exactly $\Pr[\mathsf{F} \wedge \overline{\mathsf{Eq}}]$. By the assumed security of Θ, this quantity is negligible. $\square$

ACHIEVING STRONG KEY INSULATION. The above construction is extensively used in practice. However, the scheme assumes a fully-trusted device on which to store SK^* since, as described, the device can sign messages without the user's consent. We now present a simple method to achieve strong security for *any* key-insulated scheme (i.e., not just the folklore scheme above).

Let $\Pi = (\mathsf{Gen}, \mathsf{Upd}^*, \mathsf{Upd}, \mathsf{Sign}, \mathsf{Vrfy})$ be a (t, N)-key-insulated signature scheme and let $\Theta = (G, S, V)$ be a standard signature scheme. We construct a scheme Π' as follows. $\mathsf{Gen}'(1^k)$ runs $(PK, SK^*, SK_0) \leftarrow \mathsf{Gen}(1^k, N)$ followed by $(pk, sk) \leftarrow G(1^k)$. It sets $PK' = (PK, pk)$, $SK^{*\prime} = SK^*$ and $SK'_0 = (SK_0, sk)$. In other words, the user get "his own" signing key sk. The key updating algorithms are modified in the expected way, so that at all times the user stores a key of the form (SK_i, sk). When signing, the user computes both the signature of M w.r.t. Π and the signature of $(M\|i)$ w.r.t. S. Formally, $\mathsf{Sign}'_{(SK_i,sk)}(i, M) = \langle \mathsf{Sign}_{SK_i}(i, M), S_{sk}(M\|i)\rangle$. To verify, simply check the validity of both signatures.

The modified scheme is obviously (t, N)-key-insulated as before (a formal proof is immediate). Strong security also follows as long as Θ is secure, since an adversary who has only the master key SK^* can never forge a signature on a "new" message $(M\|i)$ with respect to Θ. We remark that it is crucial that the period i be signed along with M using sk. To summarize:

Lemma 2. *If Π is (t, N)-key-insulated and Θ is existentially unforgeable, then Π' as described is strong (t, N)-key-insulated.*

4 (t, N)-Key Insulation under the DLA

While the scheme of the previous section is asymptotically optimal in all parameters, in practice one might hope for more efficient solutions, especially for strong security. In particular, one might hope to avoid the doubling (tripling) of signature/verification time and also to reduce the length of a signature. In the following sections, we provide schemes based on specific assumptions in which signing and verifying require only a single application of the signing/verification algorithm of the underlying scheme. The signature length will also be essentially the same as that of the underlying scheme.

In this section, we present a (t, N)-key-insulated scheme which may be proven secure under the discrete logarithm assumption. Unfortunately, the lengths of the public key and the master key grow linearly with t (yet they are independent of N). Thus, while practical for small values of t, it does not completely solve the problem for $t \approx N$. We defer such a solution to the following section.

Our scheme builds on the Okamoto-Schnorr signature scheme [22, 26] which we review here. Let p, q be primes such that $p = 2q + 1$ and let $\mathcal{G}$ be the subgroup of $\mathbb{Z}_p^*$ of order q. Fix generators $g, h \in \mathcal{G}$. A public key is generated by choosing $x, y \in_R \mathbb{Z}_q$ and setting $v = g^x h^y$. To sign message M, a user chooses random $r_1, r_2 \in \mathbb{Z}_q$ and computes $w = g^{r_1} h^{r_2}$. Using a hash function H (modeled as a random oracle), the user then computes $t = H(M, w)$, where t is interpreted as an element of $\mathbb{Z}_q$. The signature is: $(w, r_1 - tx, r_2 - ty)$. A signature (w, a, b) on message M is verified by computing $t = H(M, w)$ and then checking that $w \stackrel{?}{=} g^a h^b v^t$. It can be shown [22, 21] that signature forgery is equivalent to computing $\log_g h$.

$\mathsf{Gen}(1^k, N)$: $x_0^*, y_0^*, \ldots, x_t^*, y_t^* \leftarrow \mathbb{Z}_q$ $v_i^* = g^{x_i^*} h^{y_i^*}$, for $i = 0, \ldots, t$ $SK^* = (x_1^*, y_1^*, \ldots, x_\ell^*, y_\ell^*); SK_0 = (x_0^*, y_0^*)$ $PK = (g, h, v_0^*, \ldots, v_\ell^*)$ return (PK, SK^*, SK_0)	
$\mathsf{Upd}^*(i, j, (x_1^*, y_1^*, \ldots, x_\ell^*, y_\ell^*))$: $x'_{i,j} = \sum_{k=1}^t x_k^*(j^k - i^k)$ $y'_{i,j} = \sum_{k=1}^t y_k^*(j^k - i^k)$ return $SK'_{i,j} = (x'_{i,j}, y'_{i,j})$	$\mathsf{Upd}(i, j, (x_i, y_i), (x'_{i,j}, y'_{i,j}))$: $x_j = x_i + x'_{i,j}$ $y_j = y_i + y'_{i,j}$ return $SK_j = (x_j, y_j)$
$\mathsf{Sign}_{(x_i, y_i)}(i, M)$: $r_1, r_2 \leftarrow \mathbb{Z}_q$ $w = g^{r_1} h^{r_2}$ $\tau = H(i, M, w)$ $a = r_1 - \tau x_i; b = r_2 - \tau y_i$ return $\langle i, (w, a, b) \rangle$	$\mathsf{Vrfy}_{(v_0^*, \ldots, v_\ell^*)}(M, \langle i, (w, a, b) \rangle)$: $v_i = \Pi_{k=0}^t (v_i^*)^{i^k}$ $\tau = H(i, M, w)$ if $w = g^a h^b v_i^\tau$ return 1 else return 0

Fig. 1. A strong (t, N)-key-insulated signature scheme.

Our construction achieving strong (t, N)-key-insulated security appears in Figure 1. We stress that the scheme achieves *strong* security without additional modifications, yet the time required for signing and verifying is essentially the same as in the basic Okamoto-Schnorr scheme. Furthermore, using two generators enables a proof of security for an *adaptive* adversary who can choose which time periods to expose at any point during its execution. A proof of the following theorem appears in the full version of this paper.

Theorem 1. *Under the discrete logarithm assumption, the scheme of Figure 1 is strong* (t, N)*-key-insulated and has secure key updates in the random oracle model.*

5 Perfectly Key-Insulated Signature Schemes

We now construct a strong, *perfectly* key-insulated scheme whose security (in the random oracle model) is based on what we call *trapdoor signatures*. This scheme is more efficient than the generic signature scheme presented in Section 3, and results in a variety of specific perfectly key-insulated signatures; e.g., an efficient perfectly key-insulated scheme based on ordinary RSA (in the random oracle model).

Informally, we say that signature scheme $\Theta = (G, S, V)$ is a *trapdoor signature scheme* if the following hold: (1) Key generation consists of selecting a permutation (f, f^{-1}) from some family of trapdoor permutations, choosing random y, and computing $x = f^{-1}(y)$; and (2) the public key is $\langle f, y \rangle$ and the private key is x. It is essential that it is *not* necessary to include f^{-1} as part of the private key.

Given any trapdoor signature scheme, we construct a perfectly key-insulated signature scheme Π as follows (methods for achieving strong security are discussed below): Gen chooses trapdoor permutation (f, f^{-1}) and publishes $PK = \langle f, H \rangle$ for some hash function H (which will be treated as a random oracle in our analysis). The long-term secret key is $SK^* = f^{-1}$. The key SK_i for time period i is computed as $SK_i = f^{-1}(H(i))$, and a signature on message M during period i is computed (using the basic scheme) via $\sigma \leftarrow S_{SK_i}(M)$. Verification of signature $\langle i, M \rangle$ is done using the basic verification algorithm and "period public key" $PK_i \stackrel{\text{def}}{=} \langle f, H(i) \rangle$. The security of this scheme is given by the following:

Theorem 2. *If Θ is a secure trapdoor signature scheme, then Π (as constructed above) is perfectly key-insulated and has secure key updates.*

Proof. That Π has secure key updates is obvious. Given an adversary A attacking the security of Π, we construct an adversary B attacking the security of Θ. Adversary B is given public key $\langle f, y \rangle$ for an instance of Θ as well as access to a signing oracle $S_x(\cdot)$. Assume that A makes $q(k) = \mathsf{poly}(k)$ queries to hash function $H(\cdot)$. Adversary B chooses a random index $i \in \{1, \ldots, q(k)\}$ and runs A on input $PK = f$. We assume without loss of generality that for any index I, A queries $H(I)$ before querying $\mathsf{Exp}(I)$ or $\mathsf{Sign}(I, *)$ and also before outputting a forgery of the form $(M, \langle I, \sigma \rangle)$; if not, we can have B perform these queries on its own. To answer the j^{th} query of A to $H(\cdot)$ for $j \neq i$, B chooses a random x_j, computes $y_j = f(x_j)$, and returns y_j. To answer the i^{th} query of A to $H(\cdot)$, B simply returns y. Let $I_1, \ldots, I_{q(k)}$ represent the queries of A to $H(\cdot)$. Note that B can answer honestly all oracle queries of the form $\mathsf{Sign}(I_j, *)$ for $1 \leq j \leq q(k)$: when $j \neq i$ then B has the necessary secret key and when $j = i$ then B can make use of its own signing oracle to answer the query. Furthermore, B can answer honestly all oracle queries of the form $\mathsf{Exp}(I_j)$ as long as $j \neq i$; on the other hand, B aborts the simulation if the query $\mathsf{Exp}(I_i)$ is ever asked. When A outputs a forgery $(M, \langle I_j, \sigma \rangle)$, if $j \neq i$ then B aborts; otherwise, B outputs forgery (M, σ). Note that the probability that B does not abort is exactly $1/q(k)$ and therefore $\Pr[\mathsf{Succ}_{B,\Theta}] = 1/q(k) \cdot \Pr[\mathsf{Succ}_{A,\Pi}]$. Since this quantity must be negligible, the success probability of A must be negligible as well. □

This conversion of Θ to a perfectly key-insulated scheme is quite efficient. The length of PK is roughly equal to the length of the public key in Θ, and temporary keys SK_i require as much storage as secret keys in the original scheme. Signing and verifying times in Π are essentially identical to those in Θ. As for concrete instantiations of Θ, the Guillou-Quisquater scheme [13] provides a trapdoor signature scheme based on the RSA assumption (in the random oracle model). However, a number of additional schemes satisfy this requirement as well (e.g., [8, 20, 23, 29, 27]). Thus our technique is quite flexible and allows for adaptation of a number of standard (and previously analyzed) schemes.

We also note that the loss of a factor $q(k) = q_{\text{hash}}$ (where this represents the number of hash queries) in the concrete security reduction above can be improved for schemes based on specific trapdoor permutations. In particular, when the trapdoor permutation is induced by a claw-free permutation (see [7] for a definition) and Θ is constructed via the Fiat-Shamir transform [8] (i.e., the signature corresponds to a proof of knowledge of $f^{-1}(y)$), we can obtain a security bound losing only a factor $O(q_{\text{exp}})$, where q_{exp} denotes the number of key exposures. In particular, we can achieve this tighter security reduction for the RSA-based Guillou-Quisquater scheme mentioned above.

ACHIEVING STRONG SECURITY. Strong security for any scheme following the above construction can be achieved immediately using the "generic" conversion outlined in Section 3 and proven secure in Lemma 2. This increases the cost of signature computation and verification. For specific schemes, however, we can often do better: in particular, when computation of f^{-1} can be done in a 2-out-of-2 threshold manner by the user and the device. As an example, for the RSA-based scheme in which $f_{N,e}(x) \stackrel{\text{def}}{=} x^e \bmod N$ and $f^{-1}_{N,d}(y) \stackrel{\text{def}}{=} y^d \bmod N$ (for $ed = 1 \bmod \varphi(N)$), the user and the device can share d *additively* using standard threshold techniques (e.g., [9]). Here, the user stores (at all times) d_1 and the physically-secure device stores d_2 such that $d_1 + d_2 = d \bmod \varphi(N)$. To compute the key SK_i for period i, the device sends $x_{i,2} = H(i)^{d_2}$ to the user who then computes $SK_i = x_{i,2} \cdot H(i)^{d_1} = H(i)^d$. We note that similar threshold techniques are available for computing f^{-1} in 2^t-root signature schemes [18], showing that the scheme based on Ong-Schnorr signatures can be efficiently made strong as well.

6 Relation to Identity-Based Signature Schemes

An *ID-based signature scheme* [28] allows a trusted center to publish a system-wide public key PK while keeping secret a "master" key SK^*, and to then use SK^* to extract signing keys SK_I corresponding to *any identity* I. The security of ID-based signatures roughly states that no coalition of users can sign on behalf of any other user. By identifying time periods with identities, we see that any ID-based signature scheme yields a perfectly (but not necessarily strong) key-insulated signature scheme. Although the converse does not necessarily hold, we note that our construction of the previous section *does* yield an identity-based signature scheme as well. For example, the RSA-based construction discussed above (for "large" public exponent e) yields essentially the Guillou-Quisquater ID-based signature scheme. We mention, however, that prior to our work no formal proofs of security for any identity-based signature scheme have appeared. We believe that it is extremely important to provide such formal treatment due to the practical relevance of both ID-based and key-insulated signatures.

We also remark that very recently (and independently from this work) several proposals [25, 24, 4, 14] for ID-based signatures have been given. (Among these, only [4] provides formal definitions and analysis; indeed, one of the schemes of [14] was recently broken [5].) Interestingly, they all can be viewed as applying our methodology above to various trapdoor signature schemes using the same function f^{-1}. Roughly, the corresponding function (considered in a "gap Diffie-Hellman" group; see [17]) has the form $f^{-1}_{g,g^a}(g^b) = g^{ab}$. This (inverse) function can be efficiently computed given the trapdoor a. Even though f itself is not efficiently computable given only g, g^a, one can easily see that all we need in Theorem 2 is to efficiently sample random pairs of the form (g^b, g^{ab}) (in order to respond to the random oracle queries), which is easy to do for the above f. Thus, our approach encompasses a variety of proposed schemes, and almost immediately yields a simple proof of security in each case.

We note that these proposals for ID-based schemes in gap Diffie-Hellman groups may be efficiently converted to *strong* key-insulated schemes. In particular, they can be made strong by randomly splitting $a = a_1 + a_2$ and noticing that $f^{-1}_{g,g^a}(H(i)) = (H(i))^a = (H(i))^{a_1}(H(i))^{a_2}$, so that the device can compute $(H(i))^{a_2}$ and the user can then multiply it by $(H(i))^{a_1}$ to get the key for the current period.

References

1. M. Abdalla and M. Bellare. Rekeyed Digital Signature Schemes: Damage-Containment in the Face of Key Exposure. Manuscript. July, 2001.
2. R. Anderson. Invited lecture, CCCS '97.
3. M. Bellare and S.K. Miner. A Forward-Secure Digital Signature Scheme. Crypto '99.
4. J. Cha and J. Cheon. An Identity-based Signature Scheme from Gap Diffie-Hellman Groups. Available at `http://eprint.iacr.org/2002/018/`.
5. J. Cheon. A Universal Forgery of Hess's Second ID-based Signature against the Known-message Attack. Available at `http://eprint.iacr.org/2002/028/`.
6. Y. Dodis, J. Katz, S. Xu and M. Yung. Key-Insulated Public-Key Cryptosystems. Eurocrypt 2002.
7. Y. Dodis and L. Reyzin. On the Power of Claw-Free Permutations. SCN 2002.
8. A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Crypto '86.
9. R. Gennaro, T. Rabin, S. Jarecki, and H. Krawczyk. Robust and Efficient Sharing of RSA Functions. J. Crypto 13(2): 273–300 (2000).
10. M. Girault. Relaxing Tamper-Resistance Requirements for Smart Cards Using (Auto)-Proxy Signatures. CARDIS '98.
11. O. Goldreich, B. Pfitzmann, and R.L. Rivest. Self-Delegation with Controlled Propagation — or — What if You Lose Your Laptop? Crypto '98.
12. S. Goldwasser, S. Micali, and R.L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. SIAM J. Computing 17(2): 281–308 (1988).
13. L.C. Guillou and J.-J. Quisquater. A Practical Zero-Knowledge Protocol Fitted to Security Microprocessors Minimizing Both Transmission and Memory. Eurocrypt '88.
14. F. Hess. Exponent Group Signature Schemes and Efficient Identity Based Signature Schemes Based on Pairings. Available at `http://eprint.iacr.org/2002/012/`.
15. G. Itkis. Intrusion-Resilient Signatures: Generic Constructions, or Defeating Strong Adversary with Minimal Assumptions. SCN 2002.
16. G. Itkis and L. Reyzin. SiBIR: Signer-Base Intrusion-Resilient Signatures. Crypto 2002.
17. A. Joux and K. Nguyen. Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups. Available at `http://eprint.iacr.org/2001/003/`.
18. J. Katz and M. Yung. Threshold Crytptosystems Based on Factoring. Asiacrypt 2002.
19. C.-F. Lu and S.W. Shieh. Secure Key-Evolving Protocols for Discrete Logarithm Schemes. RSA 2002.
20. S. Micali. A Secure and Efficient Digital Signature Algorithm. Technical Report MIT/LCS/TM-501, MIT, 1994.
21. K. Ohta and T. Okamoto. On Concrete Security Treatment of Signatures Derived from Identification. Crypto '98.
22. T. Okamoto. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. Crypto '92.

23. H. Ong and C. Schnorr. Fast Signature Generation with a Fiat-Shamir-Like Scheme. Eurocrypt '90.
24. K. Paterson. ID-based Signatures from Pairings on Elliptic Curves. Available at `http://eprint.iacr.org/2002/004/`.
25. R. Sakai, K. Ohgishi, M. Kasahara. Cryptosystems based on pairing. SCIC 2001.
26. C.P. Schnorr. Efficient Signature Generation by Smart Cards. J. Crypto 4(3): 161–174 (1991).
27. C.P. Schnorr. Security of 2^t-root Identification and Signatures. Crypto '96.
28. A. Shamir. Identity-Based Cryptosystems and Signature Schemes. Crypto '84.
29. V. Shoup. On the Security of a Practical Identification Scheme. J. Crypto 12(4): 247–160 (1999).
30. W.-G. Tzeng and Z.-J. Tzeng. Robust Key-Evolving Public Key Encryption Schemes. Available at `http://eprint.iacr.org/2001/009/`.