Strong Key-Insulated Signature Schemes

YEVGENIY DODIS[*] JONATHAN KATZ[†] SHOUHUAI XU[‡] MOTI YUNG[§]

August 9, 2002

Abstract

Digital signing is at the heart of Internet based transactions and e-commerce. In this global communication environment, signature computation will be frequently performed on a relatively insecure device (e.g., a mobile phone) that cannot be trusted to completely (and at all times) maintain the secrecy of the private key. Loss of private keys has a devastating effect on digital signature schemes and is considered a catastrophic event. If the loss of a key is not noticed early enough, the liability involved in the unlimited number of possible abuses (signature forgeries) may be prohibitive to the wide-scale deployment of digital signatures for high-volume transactions.

In an effort to deal with this, we propose the study of *strong key-insulated* signature schemes whose goal is to minimize the damage caused by secret-key exposures. In the key-insulated model (recently considered for public-key encryption schemes [8]), the secret key(s) stored on an insecure device are refreshed at discrete time periods via interaction with a physically-secure (but computationally-limited) 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 construct strong key-insulated signature schemes based on a variety of assumptions. First, we demonstrate and prove secure a generic construction of a strong $(N-1, N)$-key-insulated signature scheme using any standard signature scheme. We then give an improved construction of a strong (t, N)-signature scheme whose security may be based on the discrete logarithm assumption in the random oracle model. Finally, we construct strong $(N-1, N)$-key-insulated schemes based on any "trapdoor signature scheme" (a notion we introduce here); this leads to very efficient solutions based on, e.g., the RSA assumption in the random oracle model. We also investigate a close connection between our notion of key-insulated signature schemes and that of identity based signature schemes.

1 Introduction

Cryptographic primitives are typically defined based on the assumption of "perfectly secure" storage which prevents any exposure of the private keys. However, given the rapid development and ever-changing nature of the communication and computational infrastructure, waiting for this perfect (and unobtainable) situation to materialize will surely prevent the deployment of cryptography altogether.

In reality, for any signature scheme deployed on an inexpensive, mobile, and (hence) relatively insecure device, the threat of secret key exposure is perhaps the most debilitating. Exposure of the secret key typically

[*] `dodis@cs.nyu.edu`. Department of Computer Science, New York University.
[†] `jkatz@cs.columbia.edu`. Department of Computer Science, Columbia University. Work supported in part by DIMACS.
[‡] `shouhuai@yahoo.com`. Department of Computer Science, George Mason University.
[§] `moti@cs.columbia.edu`. CertCo, Inc.

means that all security guarantees are lost; furthermore, it is often much easier to obtain a secret key from a naive or unsuspecting user than to break the computational assumption on which the security of the system is based. With the increasing prevalence of mobile, wireless devices for which the importance of secure authentication is paramount, concerns about key exposure must be addressed in a satisfactory and timely fashion, and this is the issue we deal with here.

A recently-proposed method of minimizing the damage caused by secret key exposures is that of *key-insulated cryptography* [8] (motivated by earlier work on *forward-secure cryptography* [4, 5]). 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 system, 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 [8]; here, we provide the first definitions and constructions for key-insulated signature schemes. We believe that signature schemes are important case and deserve special efficient constructions and theoretical investigation of their own. Precise modeling and tailored constructions for the case of digital signatures results, in fact, in a wide variety of efficient protocols for this primitive. Theoretical connections between this primitive and the basic primitive of ID-based signatures is investigated as well.

1.1 Overview of the Model

We review the informal description of the model as given in [8], adapted here for the case of digital signatures rather than public-key encryption. As in a standard signature scheme, the user begins by registering a single public key PK. 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 $1, \ldots, N$; one may think of these each as one day. 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 to sign messages during that period. The public key PK used to verify signatures remains fixed. Signatures are labeled with the time period during which they were generated. Thus, signing message M during period i results in signature $\langle i, s \rangle$.

The user's insecure device, on which the temporary keys are stored, 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, this device 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 [8], 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.

The assumption of a basic level of synchronization, so that every party in the system knows the index of the current period, is necessary for the model to be well-defined. Also necessary is some form of authentication between the user and the physically-secure device during the key update phase. Note that if a key K is used for this authentication and 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 instead might 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).

1.2 Our Contributions

The initial work on key-insulated cryptosystems [8] focused exclusively on the case of public-key encryption; here, we focus on specialized and improved solutions for the complementary case of digital signatures. Adapting a "folklore" result (which has been put forth in [4]), 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 strong (t, N)-key-insulated signature scheme whose security may be reduced to the discrete logarithm assumption in the random oracle model. Finally, noting a connection — which had been overlooked in some previous work — between key-insulated and identity-based cryptosystems, 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 last approach also generalizes several recent (and independent from this work) proposals [6, 15, 27, 28] for identity-based signature schemes based on the so called "Gap Diffie-Hellman Groups" (see [25]).

We believe that this demonstated variety of schemes for the specialized protection of digital signatures is an important step toward full deployment of a Public Key Infrastructure in realistic environments.

1.3 Related Work

Girault [11] investigates a notion similar to that of key insulation of digital signatures in the context of smart card research. However, this preliminary work has no formal model and no proofs of security. Attempts at key-insulated public-key encryption were considered by Tzeng and Tzeng [33] and also by Lu and Shieh [20] (but only against a weak non-adaptive adversary). Key-insulated public-key encryption was first formalized, and schemes with rigorous proofs of security given, in the recent work of Dodis, et al. [8].

The notion of key insulation is related to, yet distinct from, the notion of forward security [4, 5, 3, 19, 16, 21]. In the forward-secure model (introduced by [4, 5]), the secret key is updated without any interaction with an outside device; thus, an adversary compromising the system obtains *all* the secret information existing at that point in time. In this setting the adversary cannot be prevented from signing messages associated with future time periods which is a major consideration herein. Forward-secure signature schemes, however, prevent the adversary from signing messages associated with *prior* time periods. A consequence is that even the legal user is unable to generate signatures for prior time periods.

Compared to the forward-secure model, the key-insulated model makes the stronger assumption of (a limited amount of) physically-secure storage.[1] For this reason, much stronger security guarantees — namely, that

[1]For many applications, this secure storage can be realized, for example, by a personal smartcard or a non-networked server.

the adversary cannot sign messages associated with *any* non-compromised time period — are possible. Furthermore, the assumption of secure storage enables the honest user to request "old" keys thereby allowing the legal user (who has been correctly authenticated) to sign documents for prior time periods. As we mentioned above, this is impossible in the forward-secure setting. This random access to keys also allows a two-level signature scheme in which one, "basic" level is used for normal day-to-day operation (and the key-insulated security guarantee still holds) and a second, "privileged" level is reserved for emergency or highly sensitive transactions.

Somewhat related to key insulation is the problem of signature delegation [12]. 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 [12] 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.

Independent of the present work, we have become aware of related work in the context of re-keyed digital signatures [2]. 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 *strongly* secure. Our discrete logarithm scheme has no counterpart in [2]. Our scheme based on trapdoor signatures and specialized for RSA may be viewed as the "Guillou-Quisquater" [14] analogue to their "Ong-Schnorr" [26] 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 a close connection with identity-based signature schemes [31] (a connection which has been overlooked in some previous work [11]). An identity-based signature scheme may be viewed as an $(N-1, N)$-key-insulated scheme, and vice versa. We note, however, that ensuring *strong* security requires additional work, so it might be harder to build strong key-insulated signatures than regular ID-based signatures. We discuss this connection in more detail in Section 5.

Our work has already infuenced further modeling and further schemes. Itkis and Reyzin [17] add to our notion the idea of proactive refresh capability which allows even a stronger adversary. Their adversary may corrupt both the signer and the other device as well, in an alternating fashion. We hope that various strengthenings and similar developments can be further motivated by this work, since we believe that protection of cryptographic keys is very central to the notion of security.

2 Definitions

2.1 The Model

In this section, we provide a formal model and definition for key-insulated signature schemes and their security. We begin with the definition of a key-updating signature scheme, which generalizes the notion of a key-evolving signature scheme [5]. In a key-updating signature scheme there is some data (namely, SK^*) that is never erased; this data need not be erased since it will be stored on a physically-secure device and hence never exposed.

Definition 1 *A key-updating signature scheme* Π *is a 5-tuple of polynomial 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.

A key-updating encryption scheme is used as one might expect. 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.[2] 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 does not need to be changed.

Remark 1 *The above definition corresponds to schemes supporting* random-access key updates *[8]; 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.*

2.2 Security

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 will model each of these attacks by defining appropriate oracles to which the adversary is given access. To model key exposures, we give the adversary access to a *key exposure oracle* $\mathsf{Exp}_{SK^*, SK_0}(\cdot)$ that does

[2] The purpose of SK_0 is to ensure *strong* security; i.e., protection against (compromises of) the physically-secure device. If such protection is not needed, we may simply set $SK_0 = \perp$.

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 (as is the case for some of our schemes), the oracle may simply run* Upd^* *"from scratch" whenever needed to answer an oracle query. To be fully general, we might have allowed 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* $\Pi = (\mathsf{Gen}, \mathsf{Upd}, \mathsf{Upd}^*, \mathsf{Sign}, \mathsf{Vrfy})$ *be a key-updating signature scheme. For any adversary A, define the following:*

$$\mathsf{Succ}_{A,\Pi}(k) \stackrel{\text{def}}{=} \Pr\left[\mathsf{Vrfy}_{PK}(M, \langle i, s\rangle) = 1 \;\middle|\; \begin{array}{c} (PK, SK^*, SK_0) \leftarrow \mathsf{Gen}(1^k, N), \\ (M, \langle i, s\rangle) \leftarrow A^{\mathsf{Sign}_{SK^*,SK_0}(\cdot,\cdot), \mathsf{Exp}_{SK^*,SK_0}(\cdot)}(PK) \end{array}\right],$$

where (i, M) was never submitted to the signing oracle and i was never submitted to the key exposure oracle. We say that Π is (t, N)-key-insulated if, for any PPT *A who submits at most t requests to the key-exposure oracle,* $\mathsf{Succ}_{A,\Pi}(k)$ *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 tha $SK'_{i,j} = SK^*$ 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 breaks in to 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). Here, 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. Note that we do not protect against an adversary who compromises *both* the physically-secure device and the user's storage; in our model, this is impossible to achieve. (But see the recent work of [17] for some partial solutions to this problem.)

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*, define the following:*

$$\mathsf{Succ}_{B,\Pi}(k) \stackrel{\text{def}}{=} \Pr\left[\mathsf{Vrfy}_{PK}(M,\langle i,s\rangle) = 1 \;\middle|\; \begin{array}{c}(PK, SK^*, SK_0) \leftarrow \mathsf{Gen}(1^k, N),\\ (M,\langle i,s\rangle) \leftarrow B^{\mathsf{Sign}_{SK^*,SK_0}(\cdot,\cdot)}(PK, SK^*)\end{array}\right],$$

where (i,M) *was never submitted to* $\mathsf{Sign}_{SK^*,SK_0}(\cdot,\cdot)$*. We say* 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 the amplification of a $(0,1)$-key-insulated scheme to a perfectly key-insulated scheme. 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 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.

If period i is "activated" (cf. Section 2.2), then the value of pk_i is well defined. In this case, let Eq be the event that $pk = pk_i$. If period i is not activated then the value of pk_i is not well defined and we simply define $\Pr[\mathsf{Eq}] = 0$. 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 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 $S_{sk'}(\cdot)$. If A ever makes a key exposure request 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. ■

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 — yet generic and powerful — method to achieve strong security for *any* key-insulated scheme (not just the folklore scheme above) and hence offer protection against the device.

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 $\mathsf{Upd}^{*\prime}$ and Upd' do not change. 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.*

Remark 3 *We can also support the following variant of forward security (in addition to the usual key-insulation property) if we use any* forward-secure *signature scheme (e.g., that of [16, 21]) in place of an ordinary signature* *scheme Θ above. Now, even if* both *the user and the secure device are compromised at period i, at least all the previous periods $1 \ldots (i-1)$ are "secure". (Of course, we must necessarily must give up the random-access key update property, and only allow to change the keys from periods i to $i+1$).*

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 — and less generic — 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, "basic" 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 [24, 29] 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: $(w, r_1 - tx, r_2 - ty)$ (where computation is done $\bmod q$). 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 [24, 23] 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

Figure 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. This is vital for our intended applications. For completeness, we include here a theorem describing the security of this construction; the proof of security appears in Appendix A.

Theorem 1 *Under the discrete logarithm assumption, and modeling $H(\cdot)$ as a random oracle, the scheme of Figure 1 is strong (t, N)-key-insulated and has secure key updates.*

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 (secure) 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. ■

We note that the conversion to a perfectly key-insulated scheme is quite efficient. The length of public key 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 trapdoor signature scheme Θ, we note that the scheme of Guillou and Quisquater [14] provides an example of such a scheme whose security is equivalent to the RSA assumption (in the random oracle model). However, a number of additional schemes satisfy this requirement as well (i.e., [9, 22, 26, 32, 30]). Thus our technique is quite flexible and allows for adaptation of a number of standard (and previously analyzed) schemes. As an example, the second scheme of [2] may be viewed as an instance of our construction instantiated with the Ong-Schnorr trapdoor signature scheme [26], implying an immediate proof of security.

Relation to Identity-based Signatures. Recall that an *ID-based signature scheme* [31] allows the trusted venter to publish a single public key PK for the system (keeping the "master" key SK^*), and use SK^* to extract a valid signing key 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. Obviously, by identifying our concept of time periods with the concept of identities, any ID-based signature scheme is equivalent to a perfectly (but not necessarily strong) key-insulated signature scheme. Indeed, when our "trapdoor" construction above is instantiated with the Guillou-Quisquater scheme, the resulting scheme is essentially equivalent to the original proposal of Shamir [31] for an ID-based signature. We mention, however, that prior to our work no truly formal definitions or 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 huge practical relevance of both ID-based and key-insulated signatures. The above connection that we found is also very interesting.

We also remark that very recently (and independently from this work) several proposals [28, 27, 6, 15] for ID-based signatures were given. (Among those, only [6] provided formal definitions and analysis; indeed, one of the schemes of [15] was recently broken [7].) Interestingly, they all can be viewed as applying our "trapdoor" methodology of Theorem 2 to various regular trapdoor signatures, since all these signatures use the same function f^{-1}. Roughly, the corresponding function (considered in some special "gap Diffie-Hellman" groups; see [25]) had the form $f^{-1}_{g,g^a}(g^b) = g^{ab}$. This (inverse) function can be indeed computed given the trapdoor a. And 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 general approach seems to encompass a variety of currently proposed schemes.

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 [14] 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[3] threshold techniques (i.e., [10]). 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 of [2] can be made strong as well. Finally, the recent proposals for ID-based signature schemes [28, 27, 6, 15] utilizing $f^{-1}_{g,g^a}(x) = x^a$ and having master key a, can also be trivially 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

[3]For our application we may assume a trusted dealer since the user *himself* acts as the dealer during the key generation phase. Furthermore, we may assume that the physically-secure device is (at worst) honest-but-curious since if this is not the case then this device can simply refuse to cooperate with the user altogether.

multiply it by $(H(i))^{a_1}$.

References

[1] M. Abdalla, J.H. An, M. Bellare, and C. Namprempre. From Identification to Signatures via the Fiat-Shamir Transform: Minimizing Assumptions for Security and Forward Security. Eurocrypt '02.

[2] M. Abdalla and M. Bellare. Rekeyed Digital Signature Schemes: Damage-Containment in the Face of Key Exposure. Manuscript. July, 2001.

[3] M. Abdalla and L. Reyzin. A New Forward-Secure Digital Signature Scheme. Asiacrypt '00.

[4] R. Anderson. Invited lecture, CCCS '97.

[5] M. Bellare and S.K. Miner. A Forward-Secure Digital Signature Scheme. Crypto '99.

[6] J. Cha and J. Cheon. An Identity-based Signature Scheme from Gap Diffie-Hellman Groups. Available from IACR E-print archive, report 2002/18, `http://eprint.iacr.org/2002/018/`.

[7] J. Cheon. A Universal Forgery of Hess's Second ID-based Signature against the Known-message Attack. Available from IACR E-print archive, report 2002/28, `http://eprint.iacr.org/2002/028/`.

[8] Y. Dodis, J. Katz, S. Xu and M. Yung. Key-Insulated Public Key Cryptosystems. Eurocrypt '02.

[9] A. Fiat and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. Crypto '86.

[10] R. Gennaro, T. Rabin, S. Jarecki, and H. Krawczyk. Robust and Efficient Sharing of RSA Functions. J. Crypto 13(2): 273–300 (2000).

[11] M. Girault. Relaxing Tamper-Resistance Requirements for Smart Cards Using (Auto)-Proxy Signatures. CARDIS '98.

[12] O. Goldreich, B. Pfitzmann, and R.L. Rivest. Self-Delegation with Controlled Propagation — or — What if You Lose Your Laptop? Crypto '98.

[13] 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).

[14] L.C. Guillou and J.-J. Quisquater. A Practical Zero-Knowledge Protocol Fitted to Security Microprocessors Minimizing Both Transmission and Memory. Eurocrypt '88.

[15] F. Hess. Exponent Group Signature Schemes and Efficient Identity Based Signature Schemes Based on Pairings. Available from IACR E-print archive, report 2002/12, `http://eprint.iacr.org/2002/012/`.

[16] G. Itkis and L. Reyzin. Forward Secure Signatures with Optimal Signing and Verifying. Crypto '01.

[17] G. Itkis and L. Reyzin. SiBIR: Signer-Base Intrusion-Resilient Signatures, To appear, Crypto 2002.

[18] J. Katz and M. Yung. Threshold Crytptosystems Based on Factoring. Available at `http://eprint.iacr.org`.

[19] H. Krawczyk. Simple Forward-Secure Signatures From any Signature Scheme. ACM CCCS '00.

[20] C.-F. Lu and S.W. Shieh. Secure Key-Evolving Protocols for Discrete Logarithm Schemes. RSA 2002.

[21] T. Malkin, D. Micciancio, and S. Miner. Composition and Efficiency Tradeoffs for Forward-Secure Digital Signatures. Eurocrypt 2002.

[22] S. Micali. A Secure and Efficient Digital Signature Algorithm. Technical Report MIT/LCS/TM-501, MIT, 1994.

[23] K. Ohta and T. Okamoto. On Concrete Security Treatment of Signatures Derived from Identification. Crypto '98.

[24] T. Okamoto. Provably Secure and Practical Identification Schemes and Corresponding Signature Schemes. Crypto '92.

[25] T. Okamoto and D. Pointcheval. The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes. In *Public Key Cryptography*, pp. 104–118, 2001.

[26] H. Ong and C. Schnorr. Fast Signature Generation with a Fiat-Shamir-Like Scheme. Eurocrypt '90.

[27] K. Paterson. ID-based Signatures from Pairings on Elliptic Curves. Available from IACR E-print archive, report 2002/4, `http://eprint.iacr.org/2002/004/`.

[28] R. Sakai, K. Ohgishi, M. Kasahara. Cryptosystems based on pairing, In *Proc. of SCIC*, 2001.

[29] C.P. Schnorr. Efficient Signature Generation by Smart Cards. J. Crypto 4(3): 161–174 (1991).

[30] C.P. Schnorr. Security of 2^t-root Identification and Signatures. Crypto '96.

[31] A. Shamir. Identity-Based Cryptosystems and Signature Schemes. Crypto '84.

[32] V. Shoup. On the Security of a Practical Identification Scheme. J. Crypto 12(4): 247–160 (1999).

[33] W.-G. Tzeng and Z.-J. Tzeng. Robust Key-Evolving Public Key Encryption Schemes. Available at `http://eprint.iacr.org`.

A Proof of Theorem 1

We prove the theorem in a number of steps. We first consider the notion of a key-updating identification protocol and state the natural definition of key-insulated security in this context. Then, we show and prove secure a particular identification scheme based on the discrete logarithm assumption. Applying the Fiat-Shamir tranformation [9] to this protocol yields the protocol of Figure 1; security of the transformed protocol (considered as a signature scheme) in the random oracle model follows in a straightforward way from the results of [5] (see also [1]).[4]

We work in the standard framework for identification protocols in which public keys are associated with users. The prover has a secret key SK associated with a public key PK, and the prover wants to identify himself to the verifier. We assume here, for simplicity, a three-round protocol in which the prover sends an initial message, the verifier sends a random "challenge", and the prover responds with some "answer". Informally, an identification protocol is secure if an adversary, even after participating as a verifier in many interactions,

[4]Fiat-Shamir transform in the context of standard security and forward security; however, it is clear that their results can be extended to the case of key-insulated security.

cannot impersonate the prover to another verifier. We note that our definition of security includes the case of an adversary who may act as a *dishonest* verifier and choose his challenge in an arbitrary manner.

Completely analogous to the key-updating signature scheme defined in Section 2, we may also define a key-updating identification scheme. Here we have a master key SK^* which is stored on a physically-secure device. At the beginning of time period i, the prover interacts with the secure device in order to obtain a key SK_i valid for the current time period. The prover (who may be operating on an insecure device) proves his identity to a verifier in period i using key SK_i.

As with key-updating signatures, we may consider an adversary who interacts with the prover in an execution of the identification protocol during various time periods, and may additionally compromise the insecure device and obtain the temporary keys for a limited number of time periods. We say that an identification scheme is (t, N)-key-insulated if, for any adversary who compromises the system at most t times, the adversary will not be able to successfully impersonate the prover during any time period other than those in which a compromise occurred. A formal definition along the lines of Definition 2 is easily obtained from the above discussion. We may define *strong* security in an analogous fashion to Definition 4.

A proof of the following may be immediately dervived from [1] (cf. footnote 4).

Theorem 3 *For any strong (t, N)-key-insulated identification scheme, the corresponding key-updating signature scheme derived by applying the Fiat-Shamir transform [9] (and assuming a random oracle) is strong (t, N)-key-insulated. Furthermore, if the identification scheme has secure key updates then so does the signature scheme.*

In fact, we note that the identification scheme need only be secure against a passive adversary (see [1]) for Theorem 3 to hold; however, our construction achieves the stronger level of security (security against an adaptive adversary) anyway.

The identification scheme will be defined in the obvious way based on Figure 1. Primes p, q with $p = 2q + 1$ are fixed, as are elements $g, h \in \mathbb{Z}_p^*$ of order q. A user's public key PK is chosen by first picking $x_0^*, y_0^*, \ldots, x_\ell^*, y_\ell^* \in \mathbb{Z}_q$ and setting $v_i^* = g^{x_i^*} h^{y_i^*}$. The public key is $(v_0^*, \ldots, v_\ell^*)$, SK^* is $(x_1^*, y_1^*, \ldots, x_\ell^*, y_\ell^*)$, and SK_0 is simply (x_0^*, y_0^*). Key updates are done as in Figure 1.

The identification protocol for time period i proceeds as follows. The prover has secret key $SK_i = (x_i, y_1) = (\sum_{k=0}^t x_k^* i^k, \sum_{k=0}^t y_k^* i^k)$ and the verifier computes period public key v_i as $v_i = \Pi_{j=0}^{\ell} (v_i^*)^{i^j}$. To begin, the prover chooses $r_1, r_2 \in \mathbb{Z}_q$, computes $w = g^{r_1} h^{r_2}$, and sends w as the first message. The verifier responds with random $\tau \in \mathbb{Z}_q$ as the challenge. The prover calculates $a = r_1 - \tau x_i \bmod q$ and $b = r_2 - \tau y_i \bmod q$ and sends response (a, b). The verifier then checks whether $w \stackrel{?}{=} g^a h^b v_i^\tau$.

Our proof that the identification scheme sketched above is indeed key-insulated uses the techniques from [24]. We state the following lemma without proof, and refer the reader to [24] for details:

Lemma 3 *Assume there exists an adversary A with non-negligible probability of impersonating the prover in time period i. Then there exists an algorithm which runs in expected polynomial time and outputs two accepted executions (w, τ, a, b) and (w, τ', a', b') of the identification protocol for period i (i.e., with respect to period public key v_i), with $\tau \neq \tau'$.*

Using this lemma, we now show that an adversary who can break the key-insulated identification scheme can be used to compute $\log_g h$ (which is assumed to be intractable). We denote by $\mathcal{G} \subset \mathbb{Z}_p^*$ the (unique) subgroup of order q in $\mathbb{Z}_p^*$.

Theorem 4 *The identification scheme sketched above is strong (t, N)-key-insulated, assuming the hardness of computing discrete logarithms in $\mathcal{G}$. Furthermore, it has secure key updates.*

Proof That the scheme has secure key updates is obvious. Assume there exists an adversary A who has non-negligible probability of impersonating the prover during a time period for which key exposure did not occur. We show how to use A to compute $\log_g h$. Given g and h, run the identification scheme in the presence of adversary A, taking part in executions of the protocol with A and giving SK_i to the adversary when requested (note that knowledge of $\log_g h$ is not required for any of these steps, and we can therefore handle adversaries A who may act as dishonest verifiers). Let $\mathcal{I}$ be the set of time periods for which the adversary requested a key exposure. By assumption, A has non-negligible probability of impersonating the prover during some time period $i \notin \mathcal{I}$. Let $v_i = \Pi_{j=0}^{\ell}(v_i^*)^{i^j}$. Lemma 1 shows that we can then (with overwhelming probability) generate two accepted executions (w, τ, a, b) and (w, τ', a', b') for period i with $\tau \neq \tau'$. Using this, we may calculate $x_i' = (a' - a)(\tau - \tau')^{-1} \bmod q$ and $y_i' = (b' - b)(\tau - \tau')^{-1} \bmod q$ such that $v_i = g^{x_i'} h^{y_i'}$.

There are exactly q solutions (x, y) such that $v_i = g^x h^y$. We already have one solution (x_i, y_i), and have derived a second solution (x_i', y_i'). Note that even a computationally-unbounded adversary A cannot determine the values (x_i, y_i) we already have. To prove this, first consider any accepted execution (w, t, a, b) of the protocol in period j. The values (x_j, y_j) are constrained as follows:

$$\begin{aligned} \log_g w &= r_1 + r_2 \log_g h \bmod q \\ a &= r_1 - t x_j \bmod q \\ b &= r_2 - t y_j \bmod q, \end{aligned}$$

and hence all q solutions (x_j, y_j) are equally likely. Furthermore, the key exposure requests of the adversary reveal only the ℓ values $\{(x_j, y_j)\}_{j \in \mathcal{I}}$. Since these values are simply the values of the functions $f_1(z) = \sum_{j=0}^{\ell} x_j^* z^j$ and $f_2(z) = \sum_{j=0}^{\ell} y_j^* z^j$ evaluated at ℓ distinct points, the adversary gains no information about the values of these functions at any point not in $\mathcal{I}$.

Therefore, with all but negligible probability $1/q$, the solutions (x_i, y_i) and (x_i', y_i') are distinct. We may then calculate $\log_g h = (x_i - x_i')(y_i' - y_i)^{-1} \bmod q$.

The proof of strong security is exactly similar, and is omitted. ■

Theorem 2 is implied by Theorems 3 and 4.