Key-Insulated Public-Key Cryptosystems

Yevgeniy Dodis* Jonathan Katz† Shouhuai Xu‡ Moti Yung§

June 17, 2002

Abstract

Cryptographic computations (decryption, signature generation, etc.) are often performed on a relatively insecure device (e.g., a mobile device or an Internet-connected host) which cannot be trusted to maintain secrecy of the private key. We propose and investigate the notion of *key-insulated security* whose goal is to minimize the damage caused by secret-key exposures. In our model, the secret key(s) stored on the insecure device are refreshed at discrete time periods via interaction with a physically-secure — but computationally-limited — device which stores a "master key". All cryptographic computations are still done on the insecure device, and the public key remains unchanged. In a (t, N)-key-insulated scheme, an adversary who compromises the insecure device and obtains secret keys for up to t periods of his choice is unable to violate the security of the cryptosystem for *any* of the remaining $N - t$ periods. Furthermore, the scheme remains secure (for *all* time periods) against an adversary who compromises *only* the physically-secure device.

We notice that key-insulated schemes significantly improve the security guarantee of forward-secure schemes [3, 5], in which exposure of the secret key at even a single time period (necessarily) compromises the security of the system for all future time periods. This improvement is achieved with minimal cost: infrequent key updates with a (possibly untrusted) secure device.

We focus primarily on key-insulated public-key encryption. We construct a (t, N)-key-insulated encryption scheme based on any (standard) public-key encryption scheme, and give a more efficient construction based on the DDH assumption. The latter construction is then extended to achieve chosen-ciphertext security.

*`dodis@cs.nyu.edu`. Department of Computer Science, New York University.

†`jkatz@cs.columbia.edu`. Department of Computer Science, Columbia University. Portions of this work were done while the author was at Telcordia Technologies.

‡`sxu1@gmu.edu`. Department of Information and Software Engineering, George Mason University.

§`moti@cs.columbia.edu`. CertCo, Inc.

1 Introduction

Motivation. Exposure of secret keys is perhaps the most devastating attack on a cryptosystem since it typically means that security is entirely lost. This problem is probably the greatest threat to cryptography in the real world: in practice, it is typically easier for an adversary to obtain a secret key from a naive user than to break the computational assumption on which the system is based. The threat is increasing nowadays with users carrying mobile devices which allow remote access from public or foreign domains.

Two classes of methods exist to deal with this problem. The first tries to prevent key exposure altogether. While this is an important goal, it is not always practical. For example, when using portable devices to perform cryptographic operations (e.g., decrypting transmissions using a mobile phone) one must expect that the device itself may be physically compromised in some way (e.g., lost or stolen) and thus key exposure is inevitable. Furthermore, complete prevention of key exposure — even for non-mobile devices — will usually require some degree of physical security which can be expensive and inconvenient. The second approach assumes that key exposure will inevitably occur and seeks instead to minimize the damage which results when keys are obtained by an adversary. Secret sharing [37], threshold cryptography [14, 13], proactive cryptography [33], exposure-resilient cryptography [10] and forward-secure signatures [3, 5] may all be viewed as different means of taking this approach.

The most successful solution will involve a combination of the above approaches. Physical security may be ensured for a *single* device and thus we may assume that data stored on this device will remain secret. On the other hand, this device may be computationally limited or else not suitable for a particular application and thus we are again faced with the problem that some keys will need to be stored on insecure devices which are likely to be compromised during the lifetime of the system. Therefore, techniques to minimize the damage caused by such compromises must also be implemented.

Our Model. We focus here on a notion we term *key-insulated security*. Our model is the following (the discussion here focuses on public-key encryption, yet the term applies equally-well to the case of digital signatures). The user begins by registering a single public key PK. A "master" secret key SK^* is stored on a device which is physically secure and hence resistant to compromise. All decryption, however, is done on an insecure device for which key exposure is expected to be a problem. The lifetime of the protocol is divided into distinct periods $1, \ldots, N$ (for simplicity, one may think of these time periods as being of equal length; e.g., one day). At the beginning of each period, the user interacts with the secure device to derive a temporary secret key which will be used to decrypt messages sent during that period; we denote by SK_i the temporary key for period i. On the other hand, the public key PK used to encrypt messages does *not* change at each period; instead, ciphertexts are now labeled with the time period during which they were encrypted. Thus, encrypting M in period i results in ciphertext $\langle i, C \rangle$.

The insecure device, which does all actual decryption, is vulnerable to repeated key exposures; specifically, we assume that up to $t < N$ periods can be compromised (where t is a parameter). 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 decrypt messages sent during time period i. Our notion of security (informally) is that this is all an adversary can do. In particular, the adversary will be unable to determine any information about messages sent during *all* time periods other than those in which a compromise occurred. This is the strongest level of security one can expect in such a model. We call a scheme satisfying the above notion (t, N)-key-insulated.

If the physically-secure device is completely trusted, we may have this device generate (PK, SK^*)

itself, keep SK^*, and publish PK. When a user requests a key for period i, the device may compute SK_i and send it to the user. 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), the user may generate (PK, SK) himself, publish PK, and then derive keys SK^*, SK_0. The user then sends SK^* to the device and stores SK_0 himself. When the user requests a key for period i, the device sends "partial" key SK_i' to the user, who may then compute the "actual" key SK_i using SK_{i-1} and SK_i'. In this way, the user's security is guaranteed during *all* time periods with respect to the device itself, provided that the knowledge of SK^* *alone* is not sufficient to derive any of the actual keys SK_i. We note that this strong security guarantee is essential when a single device serves many different users, offering them protection against key exposure. In this scenario, users may trust this device to update their keys, but may not want the device to be able to read their encrypted traffic. Thus, there is no reason this device should have complete (or *any*!) knowledge of their "actual" keys. Finally we note that assuring that the devices are synchronized to the same period (so that only one secret key per period is given by the physically secure device) and that they handle proper authenticated interaction is taken care of by an underlying protocol (which is outside our model).

Other Applications. Besides the obvious application to minimizing the risk of key exposures across multiple time periods, key-insulated security may also be used to protect against key exposures across multiple locations, or users. For example, a company may establish a single public key and distribute (different) secret keys to its various employees; each employee is differentiated by his "non-cryptographic ID" i (e.g., a social security number or last name), and can use his own secret key SK_i to perform the desired cryptographic operation. This approach could dramatically save on the public key size, and has the property that the system remains secure (for example, encrypted messages remain hidden) for all employees whose keys are not exposed.

A key-insulated scheme may also be used for purposes of delegation [23]; here, a user (who has previously established a public key) delegates his rights in some specified, limited way to a second party. In this way, even if up to t of the delegated parties' keys are lost, the remaining keys — and, in particular, the user's secret key —- are secure.

Finally, we mention the application of key escrow by legal authorities. For example, consider the situation in which the FBI wants to read email sent to a particular user on a certain date. If a key-insulated scheme (updated daily) is used, the appropriate key for up to t desired days can be given to the FBI without fear that this will enable the FBI to read email sent on other days. A similar application (with weaker security guarantees) was considered by [2].

Our Contributions. We introduce the notion of key-insulated security and construct efficient schemes secure under this notion. Although our definition may be applied to a variety of cryptographic primitives, we focus here on public-key encryption. In Section 3, we give a generic construction of a (t, N)-key-insulated encryption scheme based on any (standard) public-key encryption scheme. Section 4 gives a more efficient construction which is secure under the DDH assumption. Both of these schemes achieve semantic security; however, we show in Section 5 how the second scheme can be improved to achieve chosen-ciphertext security. The complexity of all our schemes is essentially independent of the total number of users N. However, at least one of the parameters is polynomial in t. This makes our schemes applicable only for moderate values of t, which is, however, sufficient for many applications. In a companion paper [16], we consider key-insulated security of signature schemes.

Related Work. Arriving at the right definitions and models for the notion we put forth here has been somewhat elusive. For example, Girault [22] considers a notion similar to key-insulated security of signature schemes. However, [22] does not present any formal definitions, nor does it present schemes which are provably secure. Recently and concurrently with our work, other attempts at formalizing key-insulated public-key encryption have been made [39, 31]. However, these works consider only a *non-adaptive* adversary who chooses which time periods to expose at the outset of the protocol, whereas we consider the more natural and realistic case of an *adaptive* adversary who may choose which time periods to expose at any point during protocol execution. Furthermore, the solution of [39] for achieving chosen-ciphertext security is proven secure in the random oracle model; our construction of Section 5 is proven secure against chosen-ciphertext attacks in the standard model ([31] does not address chosen-ciphertext security at all). Finally, our definition of security is stronger than that considered in [39, 31]. Neither work considers the case of an untrusted, physically-secure device. Additionally, [31] require only that an adversary cannot fully determine an un-exposed key SK_i; we make the much stronger requirement that an adversary cannot break the underlying cryptographic scheme for any (set of) un-exposed periods.

Our notion of security complements the notion of forward security for digital signatures.[1] In this model [3, 5], an adversary who compromises the system during a particular time period obtains *all* the secret information which exists at that point in time. Clearly, in such a setting one cannot hope to prevent the adversary from signing messages associated with future time periods (since the adversary has all relevant information), even though no explicit key exposures happen during those periods. Forward-secure signatures, however, prevent the adversary from signing messages associated with *prior* time periods. Many improved constructions of forward-secure signatures have subsequently appeared [1, 29, 26, 32].

Our model uses a stronger assumption in that we allow for (a limited amount of) physically-secure storage which is used exclusively for key updates and is not used for the actual cryptographic computations. As a consequence, we are able to obtain a much stronger level of security in that the adversary is unable to sign/decrypt messages at *any* non-compromised time period, both in the future and in the past.

Relation to Identity-Based Cryptography. The idea of ID-based cryptography [38] (for concreteness, we concentrate on the case of ID-based encryption) is to have a trusted center publish a single public key so that users who know only each other's "non-cryptographic" identities (e.g., e-mail addresses) can securely communicate. In particular, a PKI (in which every user is additionally associated with a public key) is not needed beyond knowledge of a single global public key. Of course, the trusted center now must provide each user with a secret key which is a function of his identity. Roughly speaking, an ID-based scheme is secure if no coalition of users can compromise the privacy of any other user. Note, however, that the trusted server can compromise the security of *any* user (since this center knows all secrets of the system).

It is easy to see that an ID-based encryption scheme may be converted an $(N-1, N)$-key-insulated encryption scheme by viewing the period number as an "identity" and having the physically-secure device implement the trusted center. The converse is true as well; in other words, a (t, N)-key-insulated encryption scheme *with a fully trusted device* may be viewed as a relaxation of ID-based encryption, where we do not insist on $t = N-1$. We notice that the first practical ID-based encryption scheme was proposed only recently by Boneh and Franklin [8] in the random oracle model. Moreover, even though the model of ID-based encryption assumes a fully trusted center, it was observed by [6] that the particular scheme of [8] — when viewed as an $(N-1, N)$-key-insulated encryption scheme — can be very easily modified so that the secure device no longer needs to be trusted. This almost immediately gives a fully secure key-insulated encryption scheme. It should

[1] Although forward-security also applies to public-key encryption, forward-secure encryption schemes are not yet known. The related notion of "perfect forward secrecy" [15], where the parties exchange ephemeral keys on a per-session basis, is incomparable to our notion here.

be noted, however, that the security of this scheme is proven in the random oracle model under a very specific, number-theoretic assumption. By focusing on key-insulated security for $t \ll N$, as we do here, schemes based on weaker assumptions (in particular, not utilizing the random oracle which is the standard model we consider in this paper) and/or with improved efficiency and functionality may be designed. In particular, our results yield several ID-based encryption schemes which are provably secure *in the standard model*, when at most t out of N users collude. It is still a big open problem to design a fully secure ID-based (or key-insulated) encryption scheme without the random oracle assumption.

2 Definitions

2.1 The Model

We now provide a formal model for key-insulated security, focusing on the case of public-key encryption (other key-insulated primitives can be defined similarly; e.g., signature schemes are treated in [16]). Our definition of a key-updating encryption scheme parallels the definition of a key-evolving signature scheme which appears in [5], with one key difference: in a key-updating scheme there is some data (in particular, SK^*) which is never erased since it is stored on a physically-secure device. However, since the physically-secure device may not be fully trusted, new security concerns arise.

Definition 1 *A* key-updating (public-key) encryption scheme *is a 5-tuple of poly-time algorithms* $(\mathcal{G}, \mathcal{U}^*, \mathcal{U}, \mathcal{E}, \mathcal{D})$ *such that:*

- $\mathcal{G}$*, the* key generation algorithm*, is a probabilistic algorithm which takes 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*.*

- $\mathcal{U}^*$*, the* device key-update algorithm*, is a deterministic algorithm which takes as input an index* i *for a time period (throughout, we assume* $1 \leq i \leq N$*) and the master key* SK^**. It returns the partial secret key* SK'_i *for time period* i*.*

- $\mathcal{U}$*, the* user key-update algorithm*, is a deterministic algorithm which takes as input an index* i*, secret key* SK_{i-1}*, and a partial secret key* SK'_i*. It returns secret key* SK_i *for time period* i *(and erases* SK_{i-1}, SK'_i*).*

- $\mathcal{E}$*, the* encryption algorithm*, is a probabilistic algorithm which takes as input a public-key* PK*, a time period* i*, and a message* M*. It returns a ciphertext* $\langle i, C \rangle$*.*

- $\mathcal{D}$*, the* decryption algorithm*, is a deterministic algorithm which takes as input a secret key* SK_i *and a ciphertext* $\langle i, C \rangle$*. It returns a message* M *or the special symbol* $\perp$*.*

We require that for all messages M*,* $\mathcal{D}_{SK_i}(\mathcal{E}_{PK}(i, M)) = M$*.*

A key-updating encryption scheme is used as one might expect. A user begins by generating $(PK, SK^*, SK_0) \leftarrow \mathcal{G}(1^k, N)$, 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. At the beginning of time period i, the user requests $SK'_i = \mathcal{U}^*(i, SK^*)$ from the secure device. Using SK'_i and SK_{i-1}, the user may compute $SK_i = \mathcal{U}(i, SK_{i-1}, SK'_i)$. This key may be used to decrypt messages sent during time period i without further access to the device. After computation of SK_i, the user must erase SK'_i and SK_{i-1}. Note that encryption is always performed with respect to a fixed public key PK which need not be changed. Also note that the case when the device is fully trusted corresponds to $SK_0 = \perp$ and $SK_i = SK'_i$.

RANDOM-ACCESS KEY UPDATES. All the schemes we construct will have a useful property we call *random-access key updates.* For any current period j and any desired period i, it is possible to update the secret key from SK_j to SK_i in "one shot". Namely, we can generalize the key updating algorithms $\mathcal{U}^*$ and $\mathcal{U}$ to take a pair of periods i and j such that $\mathcal{U}^*((i,j), SK^*)$ outputs the partial key SK'_{ij} and $\mathcal{U}((i,j), SK_j, SK'_{ij})$ outputs SK_i. Our definition above implicitly fixes $j = i-1$. We remark that random-access key updates are impossible to achieve in the forward-security model.

2.2 Security

The are three types of exposures we protect against: (1) ordinary *key exposure*, which models (repeated) compromise of the insecure storage (i.e., leakage of SK_i); (2) *key-update* exposure, which models (repeated) compromise of the insecure device during the key-updating step (i.e., leakage of SK_{i-1} and SK'_i); and (3) *master key exposure*, which models compromise of the physically-secure device (i.e., leakage of SK^*; this includes the case when the device itself is untrusted).

To formally model key exposure attacks, we give the adversary access to two (possibly three) types of oracles. The first is a *key exposure oracle* $\mathsf{Exp}_{SK^*,SK_0}(\cdot)$ which, on input i, returns the temporary secret key SK_i (note that SK_i is uniquely defined by SK^* and SK_0). The second is a *left-or-right encryption oracle* [4], $\mathsf{LR}_{PK,\vec{b}}(\cdot,\cdot,\cdot)$, where $\vec{b} = b_1 \dots b_N \in \{0,1\}^N$, defined as:

$$\mathsf{LR}_{PK,\vec{b}}(i, M_0, M_1) \stackrel{\text{def}}{=} \mathcal{E}_{PK}(i, M_{b_i})$$

This models encryption requests by the adversary for time periods and message pairs of his choice. We allow the adversary to interleave encryption requests and key exposure requests, and in particular the key exposure requests of the adversary may be made adaptively and in any order. Finally, we may also allow the adversary access to a *decryption oracle* $\mathcal{D}^*_{SK^*,SK_0}(\cdot)$ that, on input $\langle i, C\rangle$, computes $\mathcal{D}_{SK_i}(\langle i, C\rangle)$. This models a chosen-ciphertext attack by the adversary.

The vector $\vec{b}$ for the left-or-right oracle will be chosen randomly, and the adversary succeeds by guessing the value of b_i for any un-exposed time period i. Informally, a scheme is secure if any probabilistic polynomial time (PPT) adversary has success negligibly close to 1/2. More formally:

Definition 2 *Let $\Pi = (\mathcal{G}, \mathcal{U}^*, \mathcal{U}, \mathcal{E}, \mathcal{D})$ be a key-updating encryption scheme. For adversary A, define the following:*

$$\begin{aligned}\mathsf{Succ}_{A,\Pi}(k) \quad &\stackrel{\text{def}}{=} \quad \Pr\Big[(PK, SK^*, SK_0) \leftarrow \mathcal{G}(1^k, N); \vec{b} \leftarrow \{0,1\}^N; \\ &\qquad (i, b') \leftarrow A^{\mathsf{LR}_{PK,\vec{b}}(\cdot,\cdot,\cdot), \mathsf{Exp}_{SK^*,SK_0}(\cdot), \mathcal{O}(\cdot)}(PK) : b' = b_i\Big],\end{aligned}$$

where i was never submitted to $\mathsf{Exp}_{SK^*,SK_0}(\cdot)$, and $\mathcal{O}(\cdot) = \perp$ for a plaintext-only attack and $\mathcal{O}(\cdot) = \mathcal{D}^*_{SK^*,SK_0}(\cdot)$ for a chosen-ciphertext attack (in the latter case the adversary is not allowed to query $\mathcal{D}^*(\langle i, C\rangle)$ if $\langle i, C\rangle$ was returned by $\mathsf{LR}(i,\cdot,\cdot)$). Π is (t, N)-key-insulated *if, for any* PPT *A who submits at most t requests to the key-exposure oracle, $|\mathsf{Succ}_{A,\Pi}(k) - 1/2|$ is negligible.*

As mentioned above, we may also consider attacks in which an adversary breaks in to the user's storage while a key update is taking place (i.e., the exposure occurs between two periods $i-1$ and i); we call this a *key-update exposure* at period i. In this case, the adversary receives SK_{i-1},

SK'_i, and (can compute) SK_i. Informally, we say a scheme has *secure key updates* if a key-update exposure at period i is equivalent to key exposures at periods $i-1$ and i and no more. More formally:

Definition 3 *Key-updating encryption scheme* Π *has* secure key updates *if the view of any adversary* A *making a key-update exposure request at period* i *can be perfectly simulated by an adversary* A' *who makes key exposure requests at periods* $i-1$ *and* i.

This property is desirable in real-world implementations of a key-updating encryption scheme since an adversary who gains access to the user's storage is likely to have access for several consecutive time periods (i.e., until the user detects or re-boots), including the key updating steps.

We also consider attacks which compromise the physically-secure device (this includes attacks in which this device is untrusted). Here, our definition requires that the encryption scheme be secure against an adversary which is given SK^* as input. Note that we do *not* require security against an adversary who compromises both the user's storage and the secure device — in our model this is impossible since, given SK^* and SK_i, an adversary can compute SK_j (at least for $j > i$) by himself.

Definition 4 *Let* Π *be a key-updating scheme which is* (t, N)*-key-insulated. For any adversary* B*, define the following:*

$$\begin{aligned}\mathsf{Succ}_{B,\Pi}(k) \stackrel{\text{def}}{=} \Pr\Big[&(PK, SK^*, SK_0) \leftarrow \mathcal{G}(1^k, N); \vec{b} \leftarrow \{0,1\}^N;\\ &(i, b') \leftarrow B^{\mathsf{LR}_{PK,\vec{b}}(\cdot,\cdot,\cdot), \mathcal{O}(\cdot)}(PK, SK^*) : b' = b_i\Big],\end{aligned}$$

where $\mathcal{O}(\cdot) = \perp$ *for a plaintext-only attack and* $\mathcal{O}(\cdot) = \mathcal{D}^*_{SK^*, SK_0}(\cdot)$ *for a chosen-ciphertext attack (in the latter case the adversary is not allowed to query* $\mathcal{D}^*(\langle i, C\rangle)$ *if* $\langle i, C\rangle$ *was returned by* $\mathsf{LR}(i, \cdot, \cdot)$*).* Π *is* strongly (t, N)-key-insulated *if, for any* PPT B*,* $|\mathsf{Succ}_{B,\Pi}(k) - 1/2|$ *is negligible.*

3 Generic Semantically-Secure Construction

Let (G, E, D) be any semantically secure encryption scheme. Rather than giving a separate (by now, standard) definition, we may view it simply as a $(0, 1)$-key-insulated scheme. Namely, only one secret key SK is present, and any PPT adversary, given PK and the left-or-right-oracle $\mathsf{LR}_{PK,b}$, cannot predict b with success non-negligibly different from $1/2$. Hence, our construction below can be viewed as an amplification of a $(0, 1)$-key-insulated scheme into a general (t, N)-key-insulated scheme.

We will assume below that $t, \log N = O(\mathsf{poly}(k))$, where k is our security parameter. Thus, we allow exponentially-many periods, and can tolerate exposure of any polynomial number of keys. We assume that E operates on messages of length $\ell = \ell(k)$, and construct a (t, N)-key-insulated scheme operating on messages of length $L = L(k)$.

Auxiliary Definitions. We need two auxiliary definitions: that of an *all-or-nothing transform* [35, 9] (AONT) and a *cover-free family* [19, 17]. Informally, an AONT splits the message M into n secret shares $x_1, \ldots, x_n$ (and possibly one public share z), and has the property that (1) the message M can be efficiently recovered from *all* the shares $x_1, \ldots, x_n, z$, but (2) missing even a single share x_j gives "no information" about M. As such, it is a generalization of $(n-1, n)$-secret sharing. We formalize this, modifying the conventional definitions [9, 10] to a form more compatible with our prior notation.

Definition 5 *An efficient randomized transformation $\mathcal{T}$ is called an (L,ℓ,n)-*AONT *if: (1) on input $M \in \{0,1\}^L$, $\mathcal{T}$ outputs $(X,z) \stackrel{\text{def}}{=} (x_1,\ldots,x_n,z)$, where $x_j \in \{0,1\}^\ell$; (2) there exists an efficient inverse function $\mathcal{I}$ such that $\mathcal{I}(X,z)=M$; (3) $\mathcal{T}$ satisfies the indistinguishability property described below.*
Let $X_{-j}=(x_1,\ldots,x_{j-1},x_{j+1},\ldots,x_n)$ and $\mathcal{T}_{-j}(M)=(X_{-j},z)$, where $(X,z)\leftarrow\mathcal{T}(M)$. Define the left-or-right oracle $\mathsf{LR}_b(j,M_0,M_1)\stackrel{\text{def}}{=}\mathcal{T}_{-j}(M_b)$, where $b\in\{0,1\}$. For any PPT A, we let

$$\mathsf{Succ}_{A,\mathcal{T}}(k) \stackrel{\text{def}}{=} \Pr[b\leftarrow\{0,1\}; b'\leftarrow A^{\mathsf{LR}_b(\cdot,\cdot,\cdot)}(1^k) \ : \ b'=b]$$

We require that $|\mathsf{Succ}_{A,\mathcal{T}}(k)-1/2|$ is negligible.

A family of subsets $S_1,\ldots,S_N$ over some universe U is said to be *t-cover-free* if no t subsets $S_{i_1},\ldots,S_{i_t}$ contain a (different) subset S_{i_0}; in other words, for all $\{i_0,\ldots,i_t\}$ with $i_0\notin\{i_1,\ldots,i_t\}$, we have $S_{i_0}\not\subseteq\cup_{j=1}^t S_{i_j}$. A family is said to be *(t,α)-cover-free*, where $0<\alpha<1$, if, for all $\{i_0,\ldots,i_t\}$ with $i_0\notin\{i_1,\ldots,i_t\}$, we have $|S_{i_0}\backslash\cup_{j=1}^t S_{i_j}|\ge\alpha|S_{i_0}|$. Such families are well known and have been used several times in cryptographic applications [11, 30, 21]. In what follows, we fix $\alpha=1/2$ for simplicity, and will use the following (essentially optimal) result, non-constructively proven by [19] and subsequently made efficient by [30, 25].

Theorem 1 ([19, 30, 25]) *For any N and t, one can efficiently construct a $(t,\frac{1}{2})$-cover-free collection of N subsets $S_1,\ldots,S_N$ of $U=\{1,\ldots,u\}$ with $|S_i|=n$ for all i, satisfying $u=\Theta(t^2\log N)$ and $n=\Theta(t\log N)$.*

Since we assumed that $t,\log N=O(\mathsf{poly}(k))$, we have $u,n=O(\mathsf{poly}(k))$ as well.

Construction. For simplicity, we first describe the scheme which is not strongly secure (see Definition 4), and then show a modification making it strongly secure. Let $S_1,\ldots,S_N\subset[u]\stackrel{\text{def}}{=}\{1,\ldots,u\}$ be the $(t,\frac{1}{2})$-cover-free family of n-element sets, as given by Theorem 1. Also, let $\mathcal{T}$ be a secure (L,ℓ,n)-AONT. Our (t,N)-key-insulated scheme will have a set of u independent encryption/decryption keys (sk_r,pk_r) for our basic encryption E, of which only the subset S_i will be used at time period i. Specifically, the public key of the scheme will be $PK=\{pk_1,\ldots,pk_u\}$, the secret key at time i will be $SK_i=\{sk_r \ : \ r\in S_i\}$, and the master key (for now) will be $SK^*=\{sk_1,\ldots,sk_u\}$. We define the encryption of $M\in\{0,1\}^L$ at time period i as:

$$\mathcal{E}_{PK}(i,M)=\langle\ i,\ (E_{pk_{r_1}}(x_1),\ldots,E_{pk_{r_n}}(x_n),\ z)\ \rangle,$$

where $(x_1,\ldots,x_n,z)\leftarrow\mathcal{T}(M)$ and $S_i=\{r_1,\ldots,r_n\}$. To decrypt $\langle i,(y_1,\ldots,y_n,z)\rangle$ using $SK_i=\{sk_r \ : \ r\in S_i\}$, the user first recovers the x_j's from the y_j's using D, and then recovers the message $M=\mathcal{I}(x_1,\ldots,x_n,z)$. Key updates are trivial: the device sends the new key SK_i and the user erases the old key SK_{i-1}. Obviously, the scheme supports secure key updates as well as random-access key updates.

Security. We sketch the intuition for (t,N)-key-insulated security of this scheme. The definition of the AONT implies that the system is secure at time period i provided the adversary misses *at least one* key sk_r, where $r\in S_i$. Indeed, semantic security of E implies that the adversary completely misses the shares encrypted with sk_r in this case, and hence has no information about the message M. On the other hand, if the adversary learn any t keys $SK_{i_1},\ldots,SK_{i_t}$, he learns the auxiliary keys $\{sk_r \ : \ r\in S_{i_1}\cup S_{i_2}\ldots\cup S_{i_t}\}$. Hence, the necessary and sufficient condition for (t,N)-key-insulated security is exactly the t-cover freeness of the S_i's! The parameter $\alpha=\frac{1}{2}$ is used to improve the exact security of our reduction.

Theorem 2 *The generic scheme Π described above is (t, N)-key-insulated with secure key updates, provided (G, E, D) is semantically-secure, $\mathcal{T}$ is a secure (L, ℓ, n)-AONT, and the family $S_1, \ldots, S_N$ is $(t, \frac{1}{2})$-cover-free. Specifically, breaking the security of Π with advantage ε implies the same for either (G, E, D) or $\mathcal{T}$ with advantage at least $\Omega(\varepsilon/t)$.*

Proof: Let A be the adversary for Π with $\mathsf{Succ}_{A,\Pi}(k) = \frac{1}{2} + \varepsilon$. First, we create the following adversary A' such that $\mathsf{Succ}_{A',\Pi} \geq \frac{1}{2} + \varepsilon \cdot \frac{\alpha n}{u} = \frac{1}{2} + \Omega(\frac{\varepsilon}{t})$. A' first picks a random index $r \in [u]$. Then it runs A up to the point when A outputs (i, b'). At this stage, A' looks at indices $i_1, \ldots, i_t$ of the t exposed time periods, and checks if $r \in S_i \setminus \cup_{j=1}^t S_{i_j}$. If this test succeeds, A' also outputs (i, b'). Else, it outputs (i, c), where c is a random bit. In other words, A' uses the output of A provided the guess r is such that sk_r is used at period i but A did not learn sk_r. Since A cannot output $i \in \{i_1 \ldots i_t\}$ and since our family is $(t, \frac{1}{2})$-cover-free, there are at least $\alpha|S_i| = n/2$ indices $r' \in S_i \setminus \cup_{j=1}^t S_{i_j}$. Also, since A' chose $r \in [u]$ at random and independently of the run of A, with probability at least $q = \frac{n}{2u} = \Omega(\frac{1}{t})$ we get that A' will use the output of A, so that $\mathsf{Succ}_{A',\Pi} \geq (1-q)\frac{1}{2} + q(\frac{1}{2} + \varepsilon) \geq \frac{1}{2} + \Omega(\frac{\varepsilon}{t})$, as claimed.

Next, we create a more favorable environment for A' to simplify the proof. Right after A' picks its random r, we give A' the secret keys sk_p for all $p \neq r$. At this point, there is no need to encrypt with any keys other than pk_r (A' can decrypt anyway). Moreover, there is no need for our environment to pick a full-fledged N-bit vector $\vec{b}$; rather, only b_i's such that $r \in S_i$ should be chosen. In fact, rather than choosing the b_i's (where $r \in S_i$) independently, we choose only one random bit b and set $b_i = b$ for all i s.t. $r \in S_i$. Clearly, this only helps A'.[2] Since A' is committed to output a non-random bit b' only for period i such that $r \in S_i$ and the original adversary A did not learn sk_r, we get that $\Pr(b' = b) \geq \frac{1}{2} + \Omega(\frac{\varepsilon}{t})$ in the modified environment.

To summarize, we can assume A' runs in the following environment Env_0. A' picks a random $r \in [u]$. We pick a random key pair (sk_r, pk_r) for E and a random bit $b \in \{0, 1\}$. We give A' the public key pk_r, and access to the "reduced" left-or-right oracle $\mathsf{LR}'_{pk_r,b}(i, M_0, M_1)$ which can be called only for i satisfying $r \in S_i$. The oracle runs $(X, z) \leftarrow \mathcal{T}(M_b)$, and returns the following: $(T_{-j}(M_b), E_{pk_r}(x_j))$, where $j \in [n]$ is the position of r inside S_i. The goal of A' is to predict b, and we assumed that it does so correctly with probability $q_0 = \Pr(b = b' \mid Env_0) \geq \frac{1}{2} + \Omega(\frac{\varepsilon}{t})$.

Next, we run A' in a different environment Env_1. It is identical to Env_0 except that on left-or-right query (i, M_0, M_1) (where $r \in S_i$), rather than returning $(T_{-j}(M_b), E_{pk_r}(x_j))$, Env_1 instead returns $(T_{-j}(M_b), E_{pk_r}(0))$. Namely, it encrypts the all-zero string 0 instead of the share x_j. We let $q_1 = \Pr(b = b' \mid Env_1)$.

The proof is now almost complete. The fact that $q_0 \geq \frac{1}{2} + \Omega(\frac{\varepsilon}{t})$ implies that either: (a) $q_0 - q_1 \geq \Omega(\frac{\varepsilon}{t})$; or (b) $q_1 \geq \frac{1}{2} + \Omega(\frac{\varepsilon}{t})$. We show that either case is a contradiction: case (a) to the indistinguishability of encryption E, while case (b) to the indistinguishability of AONT $\mathcal{T}$.

Case (a): If $q_0 - q_1 \geq \Omega(\frac{\varepsilon}{t})$, we break the indistiguishability of E by means of the following adversary A_1 which in turn runs A' as follows. When A' chooses $r \in [u]$, A_1 views the public key of E as pk_r and picks a random $b \in \{0, 1\}$. From now on, A_1 runs A' and answers the left-or-right queries (i, M_0, M_1) of A' as follows. If $r \notin S_i$, it ignores it. Else, it sets $(X, z) \leftarrow \mathcal{T}(M_b)$, and gives its *own left-or-right oracle* the query $(x_j, 0)$, where j is the position of r inside S_i. When it gets y (encryption of either x_j or 0) back from its oracle, it returns to A' the answer (X_{-j}, z, y). When A' finally outputs its guess b', A_1 checks if $b = b'$. If so, it guesses its own bit d was 0 (i.e., x_j was always encrypted), else that it was 1 (0 was always encrypted). It is easy to see that if $d = 0$, we

[2] One way to see this is to imagine that we picked all the b_i's independently, then picked a random b and told A' the set of i such that $b_i = b$ (and thus, the set of i where $b_i = 1 - b$), but did not disclose b.

exactly run A' in Env_0, else — exactly in Env_1. Hence, A_1 predicts d correctly with probability $\frac{1}{2}(1-q_1)+\frac{1}{2}q_0 \geq \frac{1}{2}+\Omega(\frac{\varepsilon}{t})$, contradicting the security of E.

CASE (B): If $q_1 \geq \frac{1}{2}+\Omega(\frac{\varepsilon}{t})$, we break the indistingushability of $\mathcal{T}$ by means of the following adversary A_2 which in turn runs A' as follows. A_2 picks a random key (pk_r, sk_r) and runs A' up to completion, outputting the same b' as A' does. To answer the left-or-right-query (i, M_0, M_1), where $r \in S_i$, A_2 calls its own oracle of (j, M_0, M_1), where j is the position of r inside S_i. It gets back $T_{-j}(M_b)$, and returns A' the pair $(T_{-j}(M_b), E_{pk_r}(0))$. Clearly, A_2 exactly recreates Env_1, and hence predicts its own b with probability $q_1 \geq \frac{1}{2}+\Omega(\frac{\varepsilon}{t})$, contradicting security of $\mathcal{T}$. ■

STRONG KEY-INSULATED SECURITY. The above scheme is not *strongly* (t, N)-key-insulated since the device stores all the secret keys $(sk_1, \ldots, sk_u)$. However, we can easily fix this problem. The user generates one extra key pair (sk_0, pk_0). It publishes pk_0 together with the other public keys, but keeps sk_0 for itself (never erasing it). Assuming now that $\mathcal{T}$ produces $n+1$ secret shares $x_0, \ldots, x_n$ rather than n, we just encrypt the first share x_0 with pk_0 (and the others, as before, with the corresponding keys in S_i). Formally, let $S_i' = S_i \cup \{0\}$, the master key is still $SK^* = \{sk_1, \ldots, sk_u\}$, but now $PK = \{pk_0, pk_1, \ldots, pk_u\}$ and the i-th secret key is $SK_i = \{sk_r \;:\; r \in S_i'\}$. Strong key-insulated security of this scheme follows a similar argument as in Theorem 2.

EFFICIENCY. The main parameters of the scheme are: (1) the size of PK and SK^* are both $u = O(t^2 \log N)$; and (2) the user's storage and the number of local encryptions per global encryption are both $n = O(t \log N)$. In particular, the surprising aspect of our construction is that it supports an *exponential* number of periods N and the main parameters depend mainly on t, the number of exposures we allow. Since t is usually quite small (say, $t = O(1)$ and certainly $t \ll N$), we obtain good parameters considering the generality of the scheme. (In Section 4 we use a specific encryption scheme and achieve $|PK|, |SK^*| = O(t)$ and $|SK_i| = O(1)$.)

Additionally, the choice of a secure (L, ℓ, n)-AONT defines the tradeoff between the number of encrypted bits L compared to the total encryption size, which is $(\beta n\ell + |z|)$, where β is the expansion of E, and $|z|$ is the size of the public share. In particular, if $L = \ell$, we can use any traditional $(n-1, n)$-secret sharing scheme (e.g., Shamir's scheme [37], or even XOR-sharing: pick random x_j's subject to $M = \bigoplus x_j$). This way we have no public part, but the ciphertext increases by a factor of βn as compared to the plaintext. Computationally-secure AONT's allow for better tradeoffs. For example, using either the computational secret sharing scheme of [28], or the AONT constructions of [10], we can achieve $|z| = L$, while ℓ can be as small as the security parameter k (in particular, $\ell \ll L$). Thus, we get *additive* increase $\beta n\ell$, which is essentially independent of L. Finally, in the random oracle model, we could use the construction of [9] achieving $|z| = 0$, $L = \ell(n-1)$, so the ciphertext size is $\beta\ell n \approx \beta L$. Finally, in practice one would use the above scheme to encrypt a random key K (which is much shorter than M) for a symmetric-key encryption scheme, and concatenate to this the symmetric-key encryption of M using K.

ADAPTIVE VS. NON-ADAPTIVE ADVERSARIES. Theorem 2 holds for an *adaptive* adversary who makes key exposure requests based on all information collected so far. We notice, however, that both the security and the efficiency of our construction could be somewhat improved for non-adaptive adversaries, who choose the key-exposure periods $i_1, \ldots, i_t$ at the outset of the protocol (which is the model of [39, 31, 2]). For example, it is easy to see that we no longer lose the factor t in the security of our reduction in Theorem 2. As for the efficiency, instead of using an AONT (which is essentially an $(n-1, n)$-secret sharing scheme), we can now use any $(n/2, n)$-"ramp" secret sharing scheme [7]. This means that n shares reconstruct the secret, but any $n/2$ shares yield no information about the secret. Indeed, since our family is $(t, \frac{1}{2})$-cover-free, any non-exposed period will have the adversary miss more than half of the relevant secret keys. For non-adaptive adversaries, we know at the outset which secret keys are non-exposed, and can use a simple hybrid argument over these keys to prove the security of the modified scheme. For example, we can use the "ramp" generalization of Shamir's secret sharing scheme[3] proposed by Franklin and Yung [20], and achieve $L = \ell n/2$ instead of $L = \ell$ resulting from regular Shamir's $(n-1, n)$-scheme.

4 Semantic Security Based on DDH

In this section, we present an efficient strongly (t, N)-key-insulated scheme, whose semantic security can be proved under the DDH assumption.

We first describe the basic encryption scheme we build upon. The key generation algorithm $\mathsf{Gen}(1^k)$ selects a random prime q with $|q| = k$ such that $p = 2q + 1$ is prime. This defines a unique subgroup $\mathbb{G} \subset \mathbb{Z}_p^*$ of size q in which the DDH assumption is assumed to hold; namely, it is hard to disinguish a random tuple (g, h, u, v) of four independent elements in $\mathbb{G}$ from a random tuple satisfying $\log_g u = \log_h v$. Given group $\mathbb{G}$, key generation proceeds by selecting random elements $g, h \in \mathbb{G}$ and random $x, y \in \mathbb{Z}_q$. The public key consists of g, h, and the Pedersen commitment [34] to x and y: $z = g^x h^y$. The secret key contains both x and y. To encrypt $M \in \mathbb{G}$, choose random $r \in \mathbb{Z}_q$ and compute $(g^r, h^r, z^r M)$. To decrypt (u, v, w), compute $M = w/u^x v^y$. This scheme is very similar to El Gamal encryption [18], except it uses two generators. It has been recently used by [27] in a different context.

Our Scheme. Our (t, N)-key-insulated scheme builds on the above basic encryption scheme and is presented in Figure 1. The key difference is that, after choosing $\mathbb{G}, g, h$, as above, we select two random polynomials $f_x(\tau) \stackrel{\text{def}}{=} \sum_{j=0}^{t} x_j^* \tau^j$ and $f_y(\tau) \stackrel{\text{def}}{=} \sum_{j=0}^{t} y_j^* \tau^j$ over $\mathbb{Z}_q$ of degree t. The public key consists of g, h and Pedersen commitments $\{z_0^*, \ldots, z_t^*\}$ to the coefficients of the two polynomials (see Figure 1). The user stores the constant terms of the two polynomials (i.e., x_0^* and y_0^*) and the remaining coefficients are stored by the physically-secure device. To encrypt during period i, first z_i is computed from the public key as $z_i \stackrel{\text{def}}{=} \Pi_{j=0}^{t} (z_j^*)^{i^j}$. Then (similar to the basic scheme), encryption of message M is done by choosing $r \in \mathbb{Z}_q$ at random and computing $\langle i, (g^r, h^r, z_i^r M)\rangle$. Using our notation from above, it is clear that $z_i = g^{f_x(i)} h^{f_y(i)}$. Thus, as long as the user has secret key $SK_i = (f_x(i), f_y(i))$ during period i, decryption during that period may be done just as in the basic scheme. As for key evolution, the user begins with $SK_0 = (x_0^*, y_0^*) = (f_x(0), f_y(0))$. At the start of any period i, the device transmits partial key $SK_i' = (x_i', y_i')$ to the user. Note that (cf. Figure 1) $x_i' = f_x(i) - f_x(i-1)$ and $y_i' = f_y(i) - f_y(i-1)$. Thus, since the user already has SK_{i-1}, the user may easily compute SK_i from these values. At this point, the user erases SK_{i-1}, and uses SK_i to decrypt for the remainder of the time period.

Theorem 3 *Under the DDH assumption, the encryption scheme of Figure 1 is strongly (t, N)-key-insulated under plaintext-only attacks. Furthermore, it has secure key updates and supports random-access key updates.*

Proof: Showing secure key updates is trivial, since an adversary who exposes keys SK_{i-1} and SK_i can compute the value SK_i' by itself (and thereby perfectly simulate a key-update exposure at period i). Similarly, random-access key updates can be done using partial keys $SK_{ij}' = (x_{ij}', y_{ij}')$,

[3] Here the message length $L = \ell n/2$, and the ℓ-bit parts $m_1, \ldots, m_{n/2}$ of M are viewed as the $n/2$ lower order coefficients of an otherwise random polynomial of degree $(n-1)$ over $GF[2^\ell]$. This polynomial is then evaluated at n points of $GF[2^\ell]$ to give the final n shares.

$\mathcal{G}(1^k)$: $(g,h,q) \leftarrow \mathsf{Gen}(1^k)$; $x_0^*, y_0^*, \ldots, x_t^*, y_t^* \leftarrow \mathbb{Z}_q$ $z_0^* := g^{x_0^*} h^{y_0^*}, \ldots, z_t^* := g^{x_t^*} h^{y_t^*}$ $PK := (g,h,q,z_0^*,\ldots,z_t^*)$ $SK^* := (x_1^*, y_1^*, \ldots, x_t^*, y_t^*)$; $SK_0 := (x_0^*, y_0^*)$ return PK, SK^*, SK_0	
$\mathcal{U}^*(i, SK^* = (x_1^*, y_1^*, \ldots, x_t^*, y_t^*))$: $x_i' := \sum_{j=1}^t x_j^* \left(i^j - (i-1)^j\right)$ $y_i' := \sum_{j=1}^t y_j^* \left(i^j - (i-1)^j\right)$ return $SK_i' = (x_i', y_i')$	$\mathcal{U}(i, SK_{i-1} = (x_{i-1}, y_{i-1}), SK_i' = (x_i', y_i'))$: $x_i := x_{i-1} + x_i'$ $y_i := y_{i-1} + y_i'$ return $SK_i = (x_i, y_i)$
$\mathcal{E}_{(g,h,q,z_0^*,\ldots,z_y^*)}(i, M)$: $z_i := \Pi_{j=0}^t (z_j^*)^{i^j}$ $r \leftarrow \mathbb{Z}_q$ $C := (g^r, h^r, z_i^r M)$ return $\langle i, C \rangle$	$\mathcal{D}_{(x_i,y_i)}(\langle i, C = (u,v,w) \rangle)$: $M := w / u^{x_i} v^{y_i}$ return M

Figure 1: Semantically-secure key-updating encryption scheme based on DDH.

where $x_{ij}' = f_x(i) - f_x(j)$, $y_{ij}' = f_y(i) - f_y(j)$. The user can then compute $x_i = x_j + x_{ij}'$ and $y_i = y_j + y_{ij}'$.

We now show that the scheme satisfies Definition 2. By a standard hybrid argument [4], it is sufficient to consider an adversary A who asks a single query to its left-or-right oracle (for some time period i of A's choice) and must guess the value b_i. So we assume A makes only a single query to the LR oracle during period i for which it did not make a key exposure request. In the original experiment (cf. Figure 1), the output of $\mathsf{LR}_{PK,\vec{b}}(i, M_0, M_1)$ is defined as follows: choose $r \in \mathbb{Z}_q$ at random and output $\langle i, (g^r, h^r, z_i^r M_{b_i}) \rangle$. Given a tuple (g,h,u,v) which is either a DDH tuple or a random tuple, modify the original experiment as follows: the output of $\mathsf{LR}_{PK,\vec{b}}(i, M_0, M_1)$ will be $\langle i, (u, v, u^{x_i} v^{y_i} M_b) \rangle$. Note that if (g,h,u,v) is a DDH tuple, then this is a perfect simulation of the original experiment. On the other hand, if (g,h,u,v) is a random tuple then, under the DDH assumption, the success of any PPT adversary in this modified experiment cannot differ by more than a negligible amount from its success in the original experiment. It is important to note that, in running the experiment, we can answer all of A's key exposure requests correctly since all secret keys are known. Thus, in contrast to [39, 31], we may handle an adaptive adversary who chooses when to make key exposure requests based on all information seen during the experiment.

Assume now that (g,h,u,v) is a random tuple and $\log_g h \neq \log_u v$ (this will occur with all but negligible probability). We claim that the adversary's view in the modified experiment is independent of $\vec{b}$. Indeed, the adversary knows only t values of $f_x(\cdot)$ and $f_y(\cdot)$ (at points other than i), and since both $f_x(\cdot)$ and $f_y(\cdot)$ are random polynomials of degree t, the values x_i, y_i $(= f_x(i), f_y(i))$ are *information-theoretically* uniformly distributed, subject only to:

$$\log_g z_i = x_i + y_i \log_g h. \tag{1}$$

Consider the output $\langle i, (u, v, u^{x_i} v^{y_i} M_b) \rangle$ of the encryption oracle. Since:

$$\log_u (u^{x_i} v^{y_i}) = x_i + y_i \log_u v, \tag{2}$$

and (1) and (2) are linearly independent, the conditional distribution of $u^{x_i} v^{y_i}$ (conditioned on b_i and the adversary's view) is uniform. Thus, the adversary's view is independent of b_i (and hence

$\vec{b}$). This implies that the success probability of A in this modified experiment is $1/2$, and hence the success probability of A in the original experiment is at most negligibly different from $1/2$.

We now consider security against (compromises of) the physically-secure device; in this case, there are no key exposure requests but the adversary learns SK^*. Again, it is sufficient to consider an adversary who asks a single query to its left-or-right oracle (for time period i of its choice) and must guess the value b_i. Since SK^* only contains the t highest-order coefficients of t-degree polynomials, the pair (x_i, y_i) is information-theoretically uniformly distributed (for all i) subject to $x_i + y_i \log_g h = \log_g z_i$. An argument similar to that given previously shows that the success probability of the adversary is at most negligibly better than $1/2$, and hence the scheme satisfies Definition 4. ■

5 Chosen-Ciphertext Security Based on DDH

We may modify the scheme given in the previous section so as to be resistant to chosen-ciphertext attacks. In doing so, we build upon the chosen-ciphertext-secure (standard) public-key encryption scheme of Cramer and Shoup [12].

$\mathcal{G}(1^k)$: $(g, h, q) \leftarrow \mathsf{Gen}(1^k)$; $H \leftarrow \mathsf{CRHF}(1^k)$
for $i = 0$ to t and $n = 0$ to 2:
  $x^*_{i,n}, y^*_{i,n} \leftarrow \mathbb{Z}_q$
for $i = 0$ to t:
  $z^*_i := g^{x^*_{i,0}} h^{y^*_{i,0}}$; $c^*_i := g^{x^*_{i,1}} h^{y^*_{i,1}}$; $d^*_i := g^{x^*_{i,2}} h^{y^*_{i,2}}$
$PK := (g, h, q, H, \{z^*_i, c^*_i, d^*_i\}_{0 \le i \le t})$
$SK^* := (\{x^*_{i,n}, y^*_{i,n}\}_{1 \le i \le t,\, 0 \le n \le 2})$; $SK_0 := (\{x^*_{0,n}, y^*_{0,n}\}_{0 \le n \le 2})$
return PK, SK^*, SK_0

$\mathcal{U}^*(i, SK^*)$:
for $n = 0$ to 2:
  $x'_{i,n} := \sum_{j=1}^{t} x^*_{j,n} \left(i^j - (i-1)^j\right)$
  $y'_{i,n} := \sum_{j=1}^{t} y^*_{j,n} \left(i^j - (i-1)^j\right)$
return $SK'_i = (\{x'_{i,n}, y'_{i,n}\}_{0 \le n \le 2})$

$\mathcal{U}(i, SK_{i-1}, SK'_i)$:
for $n = 0$ to 2:
  $x_{i,n} = x_{i-1,n} + x'_{i,n}$
  $y_{i,n} = y_{i-1,n} + y'_{i,n}$
return $SK_i = (\{x_{i,n}, y_{i,n}\}_{0 \le n \le 2})$

$\mathcal{E}_{PK}(i, M)$:
  $z_i := \Pi_{j=0}^{t} (z^*_j)^{i^j}$; $c_i := \Pi_{j=0}^{t} (c^*_j)^{i^j}$
  $d_i := \Pi_{j=0}^{t} (d^*_j)^{i^j}$
  $r \leftarrow \mathbb{Z}_q$
  $C := (g^r, h^r, z_i^r M, (c_i d_i^\alpha)^r)$,
    where $\alpha \stackrel{\text{def}}{=} H(i, g^r, h^r, z_i^r M)$
  return $\langle i, C \rangle$

$\mathcal{D}_{SK_i}(\langle i, (u, v, w, e) \rangle)$:
  $\alpha := H(i, u, v, w)$
  if $u^{x_{i,1} + x_{i,2}\alpha} v^{y_{i,1} + y_{i,2}\alpha} \neq e$
    return $\perp$
  else $M := w / u^{x_{i,0}} v^{y_{i,0}}$
  return M

Figure 2: Chosen-ciphertext-secure key-updating encryption scheme based on DDH.

We briefly review the "basic" Cramer-Shoup scheme (in part to conform to the notation used in Figure 2). Given generators g, h of group $\mathbb{G}$ (as described in the previous section), secret keys $\{x_n, y_n\}_{0 \le n \le 2}$ are chosen randomly from $\mathbb{Z}_q$. Then, public-key components $z = g^{x_0} h^{y_0}$, $c = g^{x_1} h^{y_1}$, and $d = g^{x_2} h^{y_2}$ are computed. In addition, a function H is randomly chosen from a family of universal one-way hash functions (UOWHF's). The public key is (g, h, q, z, c, d, H).

To encrypt a message $M \in \mathbb{G}$, a random element $r \in \mathbb{Z}_q$ is chosen and the ciphertext is: $(g^r, h^r, z^r M, (cd^\alpha)^r)$, where $\alpha = H(g^r, h^r, z^r M)$. To decrypt a ciphertext (u, v, w, e), we first check whether $u^{x_1+x_2\alpha}v^{y_1+y_2\alpha} = e$. If not, we output $\perp$. Otherwise, we output $M = w/u^{x_0}v^{y_0}$.

In our extended scheme (cf. Figure 2), we choose six random, degree-t polynomials (over $\mathbb{Z}_q$) f_{x_0}, f_{y_0}, f_{x_1}, f_{y_1}, f_{x_2}, and f_{y_2}, where $f_{x_n}(\tau) \stackrel{\text{def}}{=} \sum_{j=0}^t x^*_{j,n}\tau^j$ and $f_{y_n}(\tau) \stackrel{\text{def}}{=} \sum_{j=0}^t y^*_{j,n}\tau^j$ for $0 \leq n \leq 2$. The user stores the constant term of each of these polynomials, and the remaining coefficients are stored by the physically-secure device. The public key consists of g, h, H, and Pedersen commitments to the coefficients of these polynomials. Here, H is chosen from a family of *collision-resistant* hash functions (CRHF's). For such a function H, it is infeasible to find two distinct inputs m_1 and m_2 such that $H(m_1) = H(m_2)$.

To encrypt during period i, a user first computes z_i, c_i, and d_i by evaluating the polynomials "in the exponent" (see Figure 2). Then, similar to the basic scheme, encryption of M is performed by choosing random $r \in \mathbb{Z}_q$ and computing $\langle i, (g^r, h^r, z_i^r M, (c_i d_i^\alpha)^r)\rangle$, where $\alpha \stackrel{\text{def}}{=} H(i, g^r, h^r, z_i^r M)$. Note that we now include the period i in the hash function; this will be important in the analysis. Also notice that $z_i = g^{f_{x_0}(i)}h^{f_{y_0}(i)}$, $c_i = g^{f_{x_1}(i)}h^{f_{y_1}(i)}$, and $d_i = g^{f_{x_2}(i)}h^{f_{y_2}(i)}$. Thus, the user can decrypt (just as in the basic scheme) as long as he has $f_{x_n}(i), f_{y_n}(i)$ for $0 \leq n \leq 2$. The period secret key SK_i contains exactly these values.

Theorem 4 *Under the DDH assumption, the encryption scheme of Figure 2 is strongly (t, N)-key-insulated under chosen-ciphertext attacks. Furthermore, the scheme has secure key updates and supports random-access key updates.*

Proof: That the scheme has secure key updates is trivial, since SK_i' may be computed from SK_{i-1} and SK_i. Random-access key updates are done analogously to the scheme of the previous section. We now show the key-insulated security of the scheme (cf. Definition 2). A standard hybrid argument [4] shows that it is sufficient to consider an adversary A who makes only a single request to its left-or-right oracle (for time period i of the adversary's choice) and must guess the value b_i. We stress that polynomially-many calls to the decryption oracle are allowed.

Assume A makes a single query to the LR oracle during period i for which it did not make a key exposure request. In the original experiment (cf. Figure 2), the output of $\mathsf{LR}_{PK,\vec{b}}(i, M_0, M_1)$ is as follows: choose $r \leftarrow \mathbb{Z}_q$ and output $\langle i, (g^r, h^r, z_i^r M_{b_i}, (c_i d_i^\alpha)^r)\rangle$, where α is as above. As in the proof of Theorem 3, we now modify the experiment. Given a tuple (g, h, u, v) which is either a DDH tuple or a random tuple, we define the output of $\mathsf{LR}_{PK,\vec{b}}(i, M_0, M_1)$ to be $\langle i, (u, v, \tilde{w} = u^{x_{i,0}}v^{y_{i,0}}M_{b_i}, \tilde{e} = u^{x_{i,1}+x_{i,2}\alpha}v^{y_{i,1}+y_{i,2}\alpha})\rangle$, where $\alpha \stackrel{\text{def}}{=} H(i, u, v, \tilde{w})$. Note that if (g, h, u, v) is a DDH tuple, then this results in a perfect simulation of the LR oracle from the original experiment. On the other hand, if (g, h, u, v) is a random tuple, then, under the DDH assumption, the success of any PPT adversary cannot differ by a non-negligible amount from its success in the original experiment. As in the proof of Theorem 3, note that, in running the experiment, we can answer all of A's key exposure queries. Thus, the proof handles an adaptive adversary whose key exposure requests may be made based on all information seen up to that point.

Assume now that (g, h, u, v) is a random tuple and $\log_g h \neq \log_u v$ (this happens with all but negligible probability). We show that, with all but negligible probability, the adversary's view in the modified experiment is independent of $\vec{b}$. The proof parallels [12, Lemma 2]. Say a ciphertext $\langle i, (u', v', w', e')\rangle$ is *invalid* if $\log_g u' \neq \log_h v'$. Then:

Claim: *If the decryption oracle outputs $\perp$ for all invalid ciphertexts during the adversary's attack, then the value of b_i (and hence $\vec{b}$) is independent of the adversary's view.*

The adversary knows at most t values of $f_{x_0}(\cdot)$ and $f_{y_0}(\cdot)$ (at points other than i). Since $f_{x_0}(\cdot)$ and $f_{y_0}(\cdot)$ are random polynomials of degree t, the values $x_{i,0}, y_{i,0}$ $(= f_{x_0}(i), f_{y_0}(i))$ are uniformly distributed subject only to the constraint given by the public key:

$$\log_g z_i = x_{i,0} + y_{i,0} \log_g h. \tag{3}$$

Furthermore, when the decryption oracle decrypts valid ciphertexts $\langle i, (u', v', w', e')\rangle$, the adversary only obtains linearly-dependent relations $r' \log_g z_i = r' x_{i,0} + r' y_{i,0} \log_g h$ (where $r' \stackrel{\text{def}}{=} \log_g u'$). Similarly, decryptions of valid ciphertexts at other time periods do not further constrain $x_{i,0}, y_{i,0}$. Now consider the third component $u^{x_{i,0}} v^{y_{i,0}} M_{b_i}$ of the encryption oracle (the only one which depends on b_i). Specifically, consider the discrete log of the "one-time pad" $u^{x_{i,0}} v^{y_{i,0}}$:

$$\log_u(u^{x_{i,0}} v^{y_{i,0}}) = x_{i,0} + y_{i,0} \log_u v. \tag{4}$$

Since we assumed that $\log_u v \neq \log_g h$, (3) and (4) are linearly independent and the distribution of $u^{x_{i,0}} v^{y_{i,0}}$ (conditioned on b_i and the adversary's view) is uniform. Thus, $u^{x_{i,0}} v^{y_{i,0}}$ acts as a perfect "one-time pad" and the adversary's view is independent of b_i. The following claim now completes the proof of key-insulated security:

Claim: *With all but negligible probability, the decryption oracle will output $\perp$ for all invalid ciphertexts.*

Consider an invalid ciphertext $\langle j, (u', v', w', e')\rangle$, where j represents a period during which a key exposure request was not made, and let $\alpha' = H(j, u', v', w')$. We show that, with all but negligible probability, this ciphertext is rejected if it is invalid. There are two cases to consider: (1) $j = i$ (recall that i is the period during which the call to the LR oracle is made) and (2) $j \neq i$.

When $j = i$, the proof of the claim follows the proof of [12, Claim 2] exactly. The adversary knows at most t values of $f_{x_1}(\cdot), f_{y_1}(\cdot), f_{x_2}(\cdot)$, and $f_{y_2}(\cdot)$ (at points other than i). Since these are all random polynomials of degree t, the values $(x_{i,1}, y_{i,1}, x_{i,2}, y_{i,2})$ are uniformly distributed subject only to:

$$\begin{aligned}
\log_g c_i &= x_{i,1} + y_{i,1} \log_g h && (5)\\
\log_g d_i &= x_{i,2} + y_{i,2} \log_g h && (6)\\
\log_u \tilde{e} &= x_{i,1} + \alpha x_{i,2} + (\log_u v)\, y_{i,1} + (\log_u v)\, \alpha\, y_{i,2}, && (7)
\end{aligned}$$

where (5) and (6) come from the public key and (7) comes from the output of the encryption oracle. If the submitted ciphertext $\langle i, (u', v', w', e')\rangle$ is invalid and $(u', v', w', e') \neq (u, v, \tilde{w}, \tilde{e})$, there are three possibilities:

Case 1. $(u', v', w') = (u, v, \tilde{w})$. In this case, $e' \neq \tilde{e}$ ensures that the decryption oracle will reject.

Case 2. $(u', v', w') \neq (u, v, \tilde{w})$ but $H(i, u', v', w') = H(i, u, v, \tilde{w})$. This immediately violates the collision-resistance of our hash function and hence cannot occur with non-negligible probability.

Case 3. $H(i, u', v', w') \neq H(i, u, v, \tilde{w})$, i.e. $\alpha \neq \alpha'$. The decryption oracle will reject unless:

$$\log_{u'} e' = x_{i,1} + \alpha' x_{i,2} + (\log_{u'} v')\, y_{i,1} + (\log_{u'} v')\, \alpha'\, y_{i,2}. \tag{8}$$

But (5)–(8) are all linearly independent when $\alpha \neq \alpha'$, $\log_g h \neq \log_u v$ and $\log_g h \neq \log_{u'} v'$ (the ciphertext is invalid), from which it follows that the decryption oracle rejects except with probability $1/q$. (As in [12], each rejection further constrains the values $(x_{i,1}, y_{i,1}, x_{i,2}, y_{i,2})$; however, the k^{th} query will be rejected except with probability at most $1/(q - k + 1)$.)

When $j \neq i$, the proof is a bit more involved. The 8-tuple $(x_{i,1}, y_{i,1}, x_{i,2}, y_{i,2}, x_{j,1}, y_{j,1}, x_{j,2}, y_{j,2})$ is uniformly distributed subject to several constraints. First, we have the three constraints (5)–(7). Next, we have the following two constraints arising from the public key:

$$\log_g c_j = x_{j,1} + y_{j,1} \log_g h \tag{9}$$
$$\log_g d_j = x_{j,2} + y_{j,2} \log_g h. \tag{10}$$

Furthermore, since the adversary could have made up to t key exposure requests (at periods other than i and j), it may now know t values of each of $f_{x_1}, f_{x_2}, f_{y_1}, f_{y_2}$. This means than it knows a linear relation between each pair $(x_{i,1}, x_{j,1})$, $(x_{i,2}, x_{j,2})$, $(y_{i,1}, y_{j,1})$, $(y_{i,2}, y_{j,2})$. Specifically, these relations are of the form:

$$x_{i,1} + \lambda x_{j,1} = s_1 \tag{11}$$
$$x_{i,2} + \lambda x_{j,2} = s_2 \tag{12}$$
$$y_{i,1} + \lambda y_{j,1} = s_3 \tag{13}$$
$$y_{i,2} + \lambda y_{j,2} = s_4, \tag{14}$$

where λ is the corresponding Lagrange coefficient $\lambda = (i - i_1) \cdots (i - i_t)/(j - i_1) \cdots (j - i_t)$. Notice that the same λ appears in all four constraints. On first glance, it appears we have more constraints than unknowns. However, it is easy to see that (13) is linearly dependent on (5), (9), and (11) while (14) is linearly dependent on (6), (10), and (12). Hence, we only have 7 linearly independent constraints and 8 unknowns.

If the ciphertext $\langle j, (u', v', w', e') \rangle$ submitted by the adversary is invalid, the decryption oracle will reject unless:

$$\log_{u'} e' = x_{j,1} + \alpha' x_{j,2} + (\log_{u'} v')\, y_{j,1} + (\log_{u'} v')\, \alpha'\, y_{j,2}. \tag{15}$$

Now, looking at all 8 equations (5)–(7), (9)–(12), (15), we see that they are linearly independent precisely when the following three conditions hold:

1. $\log_g h \neq \log_u v$. This is true with overwhelming probability since (g, h, u, v) is a random tuple.

2. $\log_g h \neq \log_{u'} v'$. This is true since the ciphertext $\langle j, (u', v', w', e') \rangle$ is invalid.

3. $\alpha \neq \alpha'$, i.e. $H(i, u, v, \tilde{w}) \neq H(j, u', v', w')$. This is true since we assumed that $i \neq j$ and H is chosen from a family of collision resistant functions. Here we require collision resistance of H since the adversary's choice of i, j is not known in advance.

Thus, (15) is linearly independent from all previous constraints and thus the ciphertext is rejected except with negligible probability at most $1/q$ (again, the k^{th} such query is rejected except with probability at most $1/(q - k + 1)$).

This completes the proof of (t, N)-key-insulated security. The proof of *strong* key-insulated security follows exactly the same arguments given above except the constraints (11)–(14) now have $\lambda = -1$, as the adversary knows $(x_{i,1} - x_{j,1})$, etc. from SK^*. ■

CRHF's vs. UOWHF's. In the proof we use the fact that H is collision resistant, while in the basic Cramer-Shoup scheme [12], a universal one-way hash function suffices. We note that this does not introduce an extra assumption as collision-resistant hash families can be constructed based on the DDH assumption [34, 36] (in fact, the discrete logarithm assumption is enough). Second, UOWHF's suffice for our construction as long as the maximum number of periods N is polynomial in the security parameter (since a factor of $1/N$ is lost by "guessing" the period i for which the adversary will submit its encryption oracle request). Third, if the adversary only makes $(t-1)$ key exposure requests and we do not require strong security, we no longer have to include the period i inside H and UOWHF's suffice again. Having said this, using a collision-resistant H seems a small price to pay for the simplicity and additional security of our scheme.

Acknowledgment: Shouhuai Xu was partially supported by an NSF grant to the Laboratory for Information Security Technology at George Mason University.

References

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

[2] M. Abe and M. Kanda. A Key Escrow Scheme with Time-Limited Monitoring for One-Way Communication. ACISP '00.

[3] R. Anderson. Invited lecture. ACM CCCS '97.

[4] M. Bellare, A. Desai, E. Jokipii, and P. Rogaway. A Concrete Security Treatment of Symmetric Encryption: Analysis of the DES Modes of Operation. FOCS '97.

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

[6] M. Bellare and A. Palacio. Protecting against Key Exposure: Strongly Key-Insulated Encryption with Optimal Threshold. Available at `http://eprint.iacr.org/2002/064/`.

[7] G. Blakley and C. Meadows. Security of Ramp Schemes. Crypto '84.

[8] D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. Crypto '01.

[9] V. Boyko. On the Security Properties of the OAEP as an All-or-Nothing Transform. Crypto '99.

[10] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-Resilient Functions and All-Or-Nothing-Transforms. Eurocrypt '00.

[11] B. Chor, A. Fiat, and M. Naor. Tracing Traitors. Crypto '94.

[12] R. Cramer and V. Shoup. A Practical Public-Key Cryptosystem Provably Secure against Adaptive Chosen-Ciphertext Attacks. Crypto '98.

[13] A. De Santis, Y. Desmedt, Y. Frankel, and M. Yung. How to Share a Function Securely. STOC '94.

[14] Y. Desmedt and Y. Frankel. Threshold cryptosystems. Crypto '89.

[15] W. Diffie, P. van Oorschot and M. Wiener. Authentication and Authenticated Key Exchanges. *Designs, Codes and Cryptography*, 2:107–125, 1992.

[16] Y. Dodis, J. Katz, S. Xu and M. Yung. Strong Key-Insulated Signature Schemes. Manuscript, 2002.

[17] A. Dyachkov and V. Rykov. A Survey of Superimposed Code Theory. In *Problems of Control and Information Theory*, vol. 12, no. 4, 1983.

[18] T. El Gamal. A Public-Key Cryptosystem and a Signature Scheme Based on the Discrete Logarithm. *IEEE Transactions of Information Theory*, 31(4): 469–472, 1985.

[19] P. Erdos, P. Frankl, and Z. Furedi. Families of Finite Sets in which no Set is Covered by the Union of r Others. In *Israel J. Math.*, 51(1–2): 79–89, 1985.

[20] M. Franklin, M. Yung. Communication Complexity of Secure Computation. STOC '92.

[21] E. Gafni, J. Staddon, and Y. L. Yin. Efficient Methods for Integrating Traceability and Broadcast Encryption. Crypto '99.

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

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

[24] 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).

[25] P. Indyk. Personal communication.

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

[27] S. Jarecki and A. Lysyanskaya. Concurrent and Erasure-Free Models in Adaptively-Secure Threshold Cryptography. Eurocrypt '00.

[28] H. Krawczyk. Secret Sharing Made Short. Crypto '93.

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

[30] R. Kumar, S. Rajagopalan, and A. Sahai. Coding Constructions for Blacklisting Problems without Computational Assumptions. Crypto '99.

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

[32] T. Malkin, D. Micciancio, and S. Miner. Efficient Generic Forward-Secure Signatures With an Unbounded Number of Time Periods. Eurocrypt '02, to appear.

[33] R. Ostrovsky and M. Yung. How to Withstand Mobile Virus Attacks. PODC '91.

[34] T. Pedersen. Non-Interactive and Information-Theoretic Secure Verifiable Secret Sharing. Crypto '91.

[35] R. Rivest. All-or-Nothing Encryption and the Package Transform. FSE '97.

[36] A. Russell. Necessary and Sufficient Conditions for Collision-Free Hashing. *Journal of Cryptology* 8(2): 87–100 (1995).

[37] A. Shamir. How to share a secret. *Comm. ACM*, 22(11):612–613, 1979.

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

[39] W.-G. Tzeng and Z.-J. Tzeng. Robust Key-Evolving Public-Key Encryption Schemes. Available at `http://eprint.iacr.org/2001/009/`.