6.889 New Developments in Cryptography Apr. 26, 2011

Cryptography Against Continual Memory Leakage

Lecturer: Yael Kalai *Scribe: Jing Chen*

1 Recap

Recall from last lecture that we have several ways to model leakage. One model is "only computation leaks" by Micali and Reyzin [11], which assumes a form of secure memory that does not leak as long as no computation is done on the data. Another one is "memory leakage" by Akavia, Goldwasser, and Vaikuntanathan [1], which assumes that everything can leak information.

From an orthogonal dimension we can talk about bounded leakage and continual leakage, where the former assumes that leakage is just one shot, while the latter allows leakage to happen repeatedly during computation. The table below summarizes all four possible combinations of leakage models.

	only computation leaks	memory leakage
bounded	This combination is not really meaningful.	Most previous work falls in this model, e.g., [1, 12, 2, 9].
continual	E.g., [8]	E.g., [4], which we will discuss today.

In last two lectures we have studied bounded memory leakage, and today we are going to study continual memory leakage. In particular, we are going to discuss signature schemes and encryption schemes that are secure against continual memory leakage —CML from now on.

2 Definition of Security Against CML

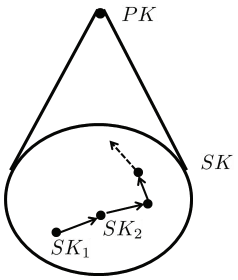
Let us start by defining public-key encryption schemes that are secure against CML.

Definition 1. *A public-key encryption scheme $\mathcal{E}$ is said to be* `semantically secure against CML` *if, $\forall$ PPT adversary $\mathcal{A}$, the advantage of $\mathcal{A}$ against challenger $\mathcal{C}h$ in the following game is negligible (in the security parameter n):*

$\mathcal{A}$		$\mathcal{C}h$
	$\xleftarrow{PK}$	$(PK, SK_1) \leftarrow Gen(1^n)$
	$\xrightarrow{L_1:\{0,1\}^n \to \{0,1\}^\lambda}$	
	$\xleftarrow{L_1(SK_1)}$	
	$\xrightarrow{\texttt{update},\ L_2}$	
	$\xleftarrow{L_2(SK_2)}$	$SK_2 \leftarrow Update(PK, SK_1)$
	$\vdots$	
	$\xrightarrow{\texttt{update},\ L_t}$	
	$\xleftarrow{L_t(SK_t)}$	$SK_t \leftarrow Update(PK, SK_{t-1})$
	$\xrightarrow{m_0, m_1}$	
	$\xleftarrow{c}$	$b \leftarrow_R \{0,1\}, c = Enc(PK, m_b)$
	$\xrightarrow{b'}$	$\mathcal{A}$ *wins if* $b' = b$

Remarks. In the above game, λ is the leakage bound as we have seen in previous lectures, and t is polynomial in n.

The function $Update$ has to be poly-time computable, and is what we are going to use to update the secret key when implementing $\mathcal{E}$. As demonstrated by the figure below, this function implies that there are many *valid* secret keys associated with the same public key, and starting from one of them we can efficiently find some others without ever changing the public key —an advantage of such a scheme.



The adversary can ask for as many secret keys as needed, but at each time point there is only one secret key stored in the system (say in a smart card) and all leakage is about the current key.

Finally, it is easy to see that $Update$ has to be randomized, otherwise the adversary can ask for leakage about SK_t at time 1, 2, ..., and ultimately will collect enough information to recover SK_t.

Signature Schemes. The definition for signature schemes that are secure against CML is very similar, and thus skipped in the lecture. The main difference is that the adversary $\mathcal{A}$ is also given a $Sign$ oracle to generate signatures for messages of his choice, and $\mathcal{A}$ wins if he manages to generate a valid signature for a message never queried to the oracle.

3 Construction of Signature Schemes Secure Against CML

Today we focus on schemes that do not allow leakage of the randomness used by the $Update$ procedure or the $Sign$ procedure (for signature schemes only). Very recently, [10] proposed signature and public-key encryption schemes that further allow leakage on the randomness used by $Update$; and [3] proposed signature schemes that further allow leakage on the randomness used by $Sign$.

3.1 From One-Way Relations to Signature Schemes

For simplicity, instead of constructing cryptographic schemes that are secure against CML as defined in previous section, below we consider a different goal. That is, to construct schemes that are secure against an adversary $\mathcal{A}$ who wins the game only if he succeeds in outputting a valid secret key SK^* for PK. Such a scheme is called a "one-way relation (OWR from now on) secure against CML", introduced by [6].

More precisely, a OWR is a scheme consisting of three PPTs $Gen, Update, Verify$ such that: $Gen(1^n) = (PK, SK_0)$, $Update(SK_\ell) = SK_{\ell+1}$, and $Verify(PK, SK_k) = 1$ for all SK_k generated by Gen and $Update$. Formal definitions are given in [6] Definition 2.3.

It turns out that the existence of a OWR secure against CML implies the existence of a signature scheme secure against CML. In particular, we have the following theorem, proved by [9].

Theorem 1. *We can construct a signature scheme secure against CML given any OWR secure against CML.*

Proof. Let the OWR scheme be $(Gen, Update, Verify)$, and let $\mathbb{R} = \{(x, w) : Verify(x, w) = 1\}$. Our signature scheme in addition uses as building blocks a semantically secure public-key encryption scheme $\mathcal{E} = (Gen^{\mathcal{E}}, Enc, Dec)$, and a simulation-sound NIZK scheme $(\ell, P, V, (S_1, S_2))$ for the following language $\mathbb{L}$: for any quadruple $s = (m, c, PK_E, x)$, $s \in \mathbb{L}$ if and only if $(x, Dec_{SK_E}(c)) \in \mathbb{R}$ where SK_E is the secret key corresponding to PK_E. (Notice that language $\mathbb{L}$ does not depend on ℓ, length of the common reference string used by the NIZK.)

The signature scheme works as follows.

KeyGen(1^n):

Generate a pair $(x, w) \in \mathbb{R}$ using Gen, a public key PK_E for $\mathcal{E}$ using $Gen^{\mathcal{E}}$, and a random common-reference string $CRS \in \{0,1\}^{\ell}$ for the NIZK.

Set the verification key $VK = (x, PK_E, CRS)$, and the secret key $SK = w$.

(Notice that $Gen^{\mathcal{E}}$ also generates a corresponding secret key SK_E, but the signature scheme does not need it.)

Sign$_w(m)$:

Let $c = Enc_{PK_E}(w)$. Let π be the NIZK proof for $(m, c, PK_E, x) \in \mathbb{L}$ generated by P using CRS (and the randomness used by Enc as a witness).

Output (c, π) as the signature for message m.

Verify$_{VK}(m, c, \pi)$:

Accept if and only if $V((m, c, PK_E, x), \pi, CRS) = 1$, where V is the verification function of the NIZK scheme.

Update(w):

Output $w' = Update_x(w)$, where $Update$ is the update function for the OWR.

To prove that the signature scheme constructed above is secure against CML, we proceed by contradiction. Assume that there exists a PPT forger $\mathcal{A}$ for the signature scheme. That is, after seeing leakage on polynomially many secret keys and querying the **Sign** oracle for polynomially many times, with non-negligible probability, $\mathcal{A}$ outputs m^* and $\sigma^* = (c^*, \pi^*)$ such that m^* is never queried to the **Sign** oracle and **Verify**$_{VK}(m^*, c^*, \pi^*) = 1$. We show that there exists a PPT $\mathcal{B}$ that breaks the OWR. That is, given $x \leftarrow Gen(1^n)$, after seeing leakage on polynomially many witnesses for x, with non-negligible probability, $\mathcal{B}$ outputs w^* such that $(x, w^*) \in \mathbb{R}$.

Upon receiving x from its challenger $\mathcal{C}h$ for the OWR which also knows the corresponding witness w_1, $\mathcal{B}$ works as follows.

- It generates $(PK_E, SK_E) \leftarrow Gen^{\mathcal{E}}(1^n)$ and $(CRS, \tau) \leftarrow S_1(1^n)$. It gives $\mathcal{A}$ the verification key $VK = (x, PK_E, CRS)$, and keeps SK_E and τ for its own use.
- Upon receiving the first leakage function L_1 from $\mathcal{A}$, $\mathcal{B}$ forwards L_1 to $\mathcal{C}h$, and forwards the reply $L_1(w_1)$ of $\mathcal{C}h$ back to $\mathcal{A}$.
- For any $k > 1$, upon receiving the update request and the k-th leakage function L_k from $\mathcal{A}$, $\mathcal{B}$ forwards both to $\mathcal{C}h$ so that $\mathcal{C}h$ updates w_{k-1} to w_k, and forwards the reply $L_k(w_k)$ of $\mathcal{C}h$ back to $\mathcal{A}$.
- Upon receiving a query to the **Sign** oracle from $\mathcal{A}$ with message m, $\mathcal{B}$ first generates $c = Enc_{PK_E}(0^n)$, and then generates $\pi = S_2((m, c, PK_E, x), CRS, \tau)$. It then answers $\mathcal{A}$'s query with (c, π).
- Once $\mathcal{A}$ outputs a message m^* and a signature (c^*, π^*), $\mathcal{B}$ outputs $w^* = Dec_{SK_E}(c^*)$ if **Verify**$_{VK}(m^*, c^*, \pi^*) = 1$ and m^* is never queried to the **Sign** oracle, and outputs $\perp$ (as a sign of failure) otherwise.

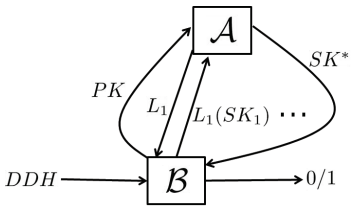
Notice that because $\mathcal{E}$ is semantically secure, and because of the zero-knowledge property of the NIZK, when $\mathcal{A}$ asks for a signature for some message, it can never distinguish whether it is interacting with $\mathcal{B}$ or the real **Sign** oracle. (Otherwise, using a hybrid argument we can either break the security of $\mathcal{E}$ or the zero-knowledge property of the NIZK.) In addition, because $\mathcal{B}$ answers $\mathcal{A}$'s leakage queries honestly, $\mathcal{A}$'s view when interacting with $\mathcal{B}$ is computationally indistinguishable from $\mathcal{A}$'s view when interacting with a honest challenger in its own game for breaking the signature scheme. Therefore by assumption, when interacting with $\mathcal{B}$, with non-negligible probability, $\mathcal{A}$ output a valid message-signature pair $(m^*, (c^*, \pi^*))$ without ever querying m^*.

Accordingly, with non-negligible probability, $\mathcal{B}$ outputs w^* instead of $\perp$. Because of the simulation soundness of the NIZK scheme, we have that $(x, Dec_{SK_E}(c^*)) \in \mathbb{R}$, i.e., $(x, w^*) \in \mathbb{R}$, and $\mathcal{B}$ succeeds in finding a witness for x. This certainly contradicts the hypothesis that the OWR is secure against CML. Therefore forger $\mathcal{A}$ does not exist, and the theorem holds. □

3.2 Construction of OWR Secure Against CML

Given the relation between OWRs and signature schemes secure against CML, we now show how to construct such OWRs. Before talking about any formal construction, let us think about how we are going to prove that some OWR is secure against CML. The idea of such proof is going to guide us through the construction.

Say we would like to prove that a given OWR is secure under the DDH assumption. The proof technique we usually use is a black-box reduction: Suppose there exists a PPT $\mathcal{A}$ that breaks the security of the OWR in the CML game, we use $\mathcal{A}$ as a black-box to construct a PPT $\mathcal{B}$ that breaks the DDH assumption, as shown in the following picture.



Because $\mathcal{A}$ is free to choose how to represent the leakage functions, intuitively, it seems that $\mathcal{B}$ can only use these leakage functions as black-boxes, and thus will need to generate $PK, SK_1, SK_2, \ldots$ on his own, and answer $\mathcal{A}$'s queries truthfully. However, if $\mathcal{B}$ itself is able to generate secret keys, what he can get from a secret key generated by $\mathcal{A}$? The idea is that the OWR should have the following structure: given SK_1, it is easy for $\mathcal{B}$ to generate other secret keys *in a small neighborhood of* SK_1, but hard to generate any secret key outside this neighborhood; and given the leakage, $\mathcal{A}$ can not tell what this neighborhood is, and thus the secret key SK^* he generates falls outside of this neighborhood with very high probability, and $\mathcal{B}$ can use such a key to break DDH. With such an idea in mind, let us proceed to the construction.

We start by introducing our construction block.

Construction Block: Groups With Bilinear Maps.

Definition 2. *A prime (say, q) order group $\mathbb{G}$ has a bilinear map e if $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ for some multiplicative group $\mathbb{G}_T$, such that $\forall$ generator g of $\mathbb{G}$, the following two properties hold.*

1. bilinear: $\forall \alpha, \beta \in \mathbb{Z}_q$, $e(g^\alpha, g^\beta) = e(g, g)^{\alpha\beta}$.
2. non-degeneracy: $e(g, g) \neq 1$, *where* 1 *is the unit of group* $\mathbb{G}_T$.

Notice that the DDH assumption is false in any $\mathbb{G}$ with a bilinear map e. Indeed, to decide wether a quadruple (x, y, z, w) is of the form $(g, h, g^\alpha, h^\alpha)$ or $(g, h, g^\alpha, h^\beta)$ with $\alpha \neq \beta$, it suffices to check whether $e(x, w) = e(y, z)$ —equal if the former, not equal if the latter. Therefore we need a new hardness assumption.

The Linear Assumption.

Definition 3. *Let g be a generator of a group $\mathbb{G}$ with prime order q, and $A = (a_{ij})_{m\times n}$ an $m \times n$ matrix over $\mathbb{Z}_q$, then $g^A \triangleq (g^{a_{ij}})_{m\times n}$.*

Definition 4. *$\forall d \geq 2$ and $\forall n \geq d + 1$, the* `linear assumption` *asserts that $g^{U_d^{n\times n}(q)} \cong_C g^{U_{d+1}^{n\times n}(q)}$, where $U_d^{n\times n}(q)$ is the random $n \times n$ matrix over $\mathbb{Z}_q$ of degree d, and $U_{d+1}^{n\times n}(q)$ is defined similarly.*

Typically, n is our security parameter and $|q|$, the length of q, is of $poly(n)$. The requirement that the random matrices are square matrices is not crucial, since we can always trunk a non-square matrix to a square one. Because q is much bigger than n, doing so will not affect much the degree of the matrix. Finally notice that if $d = 1$ and $n = 2$ then the corresponding assumption is precisely DDH.

Construction of OWR Secure Against CML. Now we proceed to construct a OWR $(Gen, Update, Verify)$ as follows.

- $Gen(1^n)$: Given group $\mathbb{G}$ of size q with bilinear map e, let g be a generator and $A \leftarrow \mathbb{Z}_q^{2\times\ell}$, that is, A is a random $2 \times \ell$ matrix over $\mathbb{Z}_q$. Set $PK = g^A$ and $SK = g^B$, where $B \leftarrow \mathbb{Z}_q^{\ell\times 2}$ such that $AB = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$. That is, letting $Ker(A) = \{b : Ab = \begin{pmatrix} 0 \\ 0 \end{pmatrix}\}$, i.e., the kernel of A, then each column of B is in $Ker(A)$. We also say that B is in the kernel of A, without causing any ambiguity.

  Note. Given only g^A and g^B, the relation of A and B can be verified efficiently via the bilinear map. Indeed, letting $C = AB$, we have that $\forall i, j \in \{1, 2\}$, $c_{ij} = \sum_{k=1}^{\ell} a_{ik}b_{kj}$, and thus $e(g, g)^{c_{ij}} = \prod_{k=1}^{\ell} e(g^{a_{ik}}, g^{b_{kj}})$, which can be computed easily given e. Let $e(g^A, g^B) \triangleq e(g, g)^{AB}$, we have that this matrix can be easily computed, and $AB = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$ if and only if $e(g^A, g^B) = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$.

- $Update(g^B)$: Let $R_1 \leftarrow \mathbb{Z}_q^{2\times 2}$. Since each element in matrix BR_1 is a linear combination of elements of B, with coefficients being elements of R_1, g^{BR_1} can be easily computed given g^B and R_1. The function then outputs g^{BR_1}.

  Note. The above update procedure can be easily repeated. That is, given $g^{BR_1R_2\ldots R_k}$, the update procedure choose $R_{k+1} \leftarrow \mathbb{Z}_q^{2\times 2}$ and output $g^{BR_1R_2\ldots R_kR_{k+1}}$.

  Also note that starting from g^B, the neighborhood of g^B within which the witnesses are updated is $g^{SPAN(B)}$, where $SPAN(B) = \{BR : R \leftarrow \mathbb{Z}_q^{2\times 2}\}$ is the span of B.

- $Verify(g^A, g^B)$: As mentioned before, given g^A and g^B it is easy to verify whether $AB = 0$ or not, via the bilinear map.

We have the following theorem.

Theorem 2. *The OWR constructed above is secure against CML under the linear assumption.*

To prove this theorem we first introduce the following claim, whose proof is left as an exercise.

Claim 1. *The linear assumption implies that* $\forall$ *constant* c *and* $\forall$ *PPT* $\mathcal{B}$*,*

$$\Pr_{\substack{A \leftarrow \mathbb{Z}_q^{2\times \ell} \\ b_1,\ldots,b_c \leftarrow Ker(A)}} \left[\mathcal{B}(g^A, b_1, \ldots, b_c) = g^{b'} \ s.t.\ b' \in Ker(A), b' \notin SPAN(b_1, \ldots, b_c)\right] = neg(n). \tag{1}$$

That is, given g^A and $b_1, \ldots, b_c \leftarrow Ker(A)$, it is hard to find $g^{b'}$ such that $b' \in Ker(A)$ and $b' \notin SPAN(b_1, \ldots, b_c)$.

Now let us prove Theorem 2.

Proof. Suppose the OWR is not secure against CML, that is, there exists PPT $\mathcal{A}$ such that with non-negligible probability (taken over the randomness used by Gen, $Update$, and $\mathcal{A}$),

$$\mathcal{A}(g^A, L_0(g^B), L_1(g^{BR_1}, \ldots, L_t(g^{BR_1\cdots R_t})) = g^{b'},$$

where $t = poly(n)$, $b' \in Ker(A)$, and $e(g,g)^{Ab'} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$.

If $\mathcal{A}$ succeeds and $b' \notin SPAN(B)$, then let us try to break the linear assumption by constructing a PPT $\mathcal{B}$ such that the probability in Equation 1 is not negligible for $c = 2$. In particular, given g^A, b_1, and b_2, $\mathcal{B}$ sets $B = (b_1, b_2)$ and gives g^A to $\mathcal{A}$. When $\mathcal{A}$ queries with leakage functions $L_0, L_1, \ldots, L_t$, $\mathcal{B}$ uses $Update$ to answer $\mathcal{A}$'s queries truthfully. Finally, when $\mathcal{A}$ succeeds and $b' \notin SPAN(B)$, $\mathcal{B}$ outputs $g^{b'}$; otherwise $\mathcal{B}$ outputs $\perp$. Because $\mathcal{B}$ answers $\mathcal{A}$'s query truthfully, $\mathcal{A}$ succeeds with non-negligible probability. Therefore it suffices for us to show that

conditioned on $\mathcal{A}$ *succeeds,* $b' \notin SPAN(B)$ *with non-negligible probability.*

Unfortunately, we do not now how to prove the above statement. To fix this problem, we let our PPT $\mathcal{B}$ break Equation 1 for $c = 4$ instead of $c = 2$. In particular, given g^A and $b_1, \ldots, b_4$, $\mathcal{B}$ feeds $\mathcal{A}$ with g^A, and answers $\mathcal{A}$'s queries $L_0, \ldots, L_t$ with $L_0(B_0), L_1(B_1), \ldots, L_t(B_t)$, where each B_k consists of two random vectors from $SPAN(b_1, \ldots, b_4)$. If $\mathcal{A}$ succeeds and outputs $g^{b'}$ with $b' \notin SPAN(b_1, \ldots, b_4)$, then $\mathcal{B}$ outputs $g^{b'}$, otherwise $\mathcal{B}$ outputs $\perp$.

By the linear assumption, $\mathcal{A}$ can not distinguish whether he is receiving leakage about vectors from a space of dimension 4 (when playing with $\mathcal{B}$) or from a space of dimension 2 (when playing the true game). Therefore when interacting with $\mathcal{B}$, $\mathcal{A}$ still succeeds with non-negligible probability.

Again it is left to show that conditioned on $\mathcal{A}$ succeeds, $b' \notin SPAN(b_1, \ldots, b_4)$ with non-negligible probability. In fact we prove a stronger result, that is, if $\mathcal{A}$ succeeds then it outputs $b' \notin SPAN(b_1, \ldots, b_4)$ with *overwhelming* probability. This is based on the following lemma, for properly chosen leakage bound λ.

Lemma 1. *For properly chosen* λ*, any leakage function* L *with leakage bound* λ*, and any constant* d*, let* $X \subseteq \mathbb{Z}_q^\ell$ *be a random subspace of dimension* d*, then*

$$(L(x_1, \ldots, x_{d/2}), X) \cong (L(u_1, \ldots, u_{d/2}), X),$$

where each x_k *is randomly chosen from* X*, and each* u_k *is uniformly and randomly chosen from* $\mathbb{Z}_q^\ell$*.*

Here "properly chosen λ" means that we can allow leakage for less than one vector, say 0.99 of a vector can be leaked. Therefore with input $d/2$ vectors, the allowed leakage rate is $\frac{0.99}{d/2}$, which is 0.499 with $d = 4$.

Informally, Lemma 1 says that even given leakage on $d/2$ random samples from X, X itself is still information-theoretically hidden, just like given "leakage" on $d/2$ uniform vectors. The proof of Lemma 1 can be found in [4], and follows from the the proof of the generalized "crooked" leftover hash lemma [7, 5].

To use Lemma 1, we take $d = 4$, $X = SPAN(b_1, \ldots, b_4)$, $L = L_k$ and $(x_1, x_2) = B_k$ for any $k \leq t$. Notice that the adversary $\mathcal{A}$ does not even get leakage on B_k directly —he only gets leakage on g^{B_k}, and thus X is information-theoretically hidden after $\mathcal{A}$ has seen any $L_k(g^{B_k})$. Because there are only polynomially many leakage functions, by a hybrid argument we have that X is still hidden even after $\mathcal{A}$ has seen all the leakage. Because X is very small compared with $\mathbb{Z}_q^\ell$, with high probability $\mathcal{A}$ will output $g^{b'}$ such that $b' \notin SPAN(b_1, \ldots, b_4)$. Therefore $\mathcal{B}$ will succeed with non-negligible probability, which contradicts Claim 1, implying that the OWR is secure against CML under the linear assumption. □

4 Encryption Schemes Secure Against CML

Finally, without giving any proof, we claim that the public-key encryption scheme constructed below is secure against CML. For simplicity, this scheme only encrypts single bits.

- $Gen(1^n)$: It is the same as the generation procedure of the OWR in last section. In particular, $PK = g^A$ with $A \leftarrow \mathbb{Z}_q^{2\times\ell}$, and $SK = g^B$ with $B \leftarrow \mathbb{Z}_q^{\ell\times 2}$ such that $AB = 0$.
- $Enc_{PK}(b)$: $Enc_{PK}(0) = g^{rA}$ with $r \leftarrow \mathbb{Z}_q^{1\times 2}$; and $Enc_{PK}(1) = g^U$ with U the uniform $1 \times \ell$ vector.
- $Dec_{SK}(c)$: Interpret c as g^C for some $1 \times \ell$ vector C, and use the bilinear map e to compute $e(g^C, g^B) \triangleq e(g, g)^{CB}$. Output 0 if $e(g, g)^{CB} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, and output 1 otherwise.
- $Update(SK)$: The same as the update procedure of the OWR.

References

[1] A. Akavia, S. Goldwasser and V. Vaikuntanathan. Simultaneous Hardcore Bits and Cryptography against Memory Attacks. TCC 2009, pages 474-495.

[2] J. Alwen, Y. Dodis, and D. Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. Crypto 2009, pages 36-54.

[3] E. Boyle, G. Segev, and D. Wichs. Fully Leakage-Resilient Signatures. To appear in EUROCRYPT 2011.

[4] Z. Brakerski, Y. T. Kalai, J. Katz, and V. Vaikuntanathan. Overcoming the hole in the bucket: Publickey cryptography resilient to continual memory leakage. FOCS 2010, pages 501-510. Full version: ePrint 2010/278. http://eprint.iacr.org/2010/278.

[5] A. Boldyreva, S. Fehr, and A. O'Neill. On notions of security for deterministic encryption, and ecient constructions without random oracle. CRYPTO 2008, pages 335-359.

[6] Y. Dodis, K. Haralambiev, A. Lopez-Alt, and D. Wichs. Cryptography against continuous memory attacks. FOCS 2010, pages 511-520.

[7] Y. Dodis and A. Smith. Correcting errors without leaking partial information. STOC 2005, pages 654-663.

[8] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. FOCS 2008, pages 293-302.

[9] J. Katz and V. Vaikuntanathan. Signature Schemes with Bounded Leakage Resilience. Asiacrypt 2009, pages 703-720.

[10] A. Lewko, M. Lewko, and B. Waters. How to Leak on Key Updates. To appear in STOC 2011.

[11] S. Micali and L. Reyzin. Physically Observable Cryptography. TCC 2004, pages 278-296.

[12] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. Crypto 2009, pages 18-35.