Cryptography Resilient to Continual Memory Leakage

Zvika Brakerski* Yael Tauman Kalai† Jonathan Katz‡ Vinod Vaikuntanathan§

Abstract

In recent years, there has been a major effort to design cryptographic schemes that remain secure even if part of the secret key is leaked. This is due to a recent proliferation of side channel attacks which, through various physical means, can recover part of the secret key. We explore the possibility of achieving security even with continual leakage, i.e., even if some information is leaked each time the key is used.

We show how to securely update a secret key while information is leaked: We construct schemes that remain secure even if an attacker, *at each time period*, can probe the entire memory (containing a secret key) and "leak" up to a $(1 - o(1))$ fraction of the secret key. The attacker may also probe the memory during the updates, and leak $O(\log k)$ bits, where k is the security parameter (relying on subexponential hardness allows k^ϵ bits of leakage during each update process). All of the above is achieved without restricting the model as is done in previous works (e.g. by assuming that "only computation leaks information" [Micali-Reyzin, TCC04]).

Specifically, under the decisional linear assumption on bilinear groups (which allows for a leakage rate of $(1/2 - o(1))$) or the symmetric external Diffie-Hellman assumption (which allows for a leakage rate of $(1 - o(1))$), we achieve the above for public key encryption, identity-based encryption, and signature schemes. Prior to this work, it was not known how to construct public-key encryption schemes even in the more restricted model of [MR].

The main contributions of this work are (1) showing how to securely update a secret key while information is leaked (in the more general model) and (2) giving a public key encryption (and IBE) schemes that are resilient to continual leakage.

*Weizmann Institute of Science and Microsoft Research, `zvika.brakerski@weizmann.ac.il`.

†Microsoft Research, `yael@microsoft.com`.

‡University of Maryland, `jkatz@cs.umd.edu`.

§IBM Research, `vinodv@alum.mit.edu`.

1 Introduction

Modern cryptography, starting from the foundational work in the late 70s, has been a huge success, leading to the definition and construction of various cryptographic primitives such as digital signatures, public-key encryption, identity-based encryption, and so forth. The working assumption in most of modern cryptography is that secret keys are generated using perfectly random bits, and once they are generated, they are perfectly secret. However, it has been observed that this assumption does not hold in practice, where attackers use different types of *side-channel attacks* to leak information about the secret key. These attacks exploit the physical characteristics of the execution of a cryptographic device, such as timing, power consumption, electro-magnetic radiation, and so forth (see [Koc96, KJJ99, OST06, HSH$^+$08] and the references therein). What's worse, many constructions that were proven to be secure under traditional cryptographic security definitions are known to be completely insecure under many such attacks (see, e.g., [Koc96] for how to break the RSA algorithm under timing attacks, and [HSH$^+$08] for how to break RSA as well as the Advanced Encryption Standard (AES) under cold-boot attacks). The focus of this work is constructing cryptographic protocols that are provably secure in the presence of side-channel attacks.

Recently, starting with the work of Ishai, Sahai and Wagner [ISW03] and Micali and Reyzin [MR04], there has been a major effort by the cryptographic community to construct such protocols. The influential work of Micali and Reyzin proposed to construct cryptographic protocols under the assumption that

"any computation, and only computation, leaks information".

Indeed, the vast variety of side-channel attacks lead to a myriad of different ways in which information can leak *every time* the cryptographic device performs some computation. Thus, it seems prudent to allow *any computation* to leak information – we will refer to this as the *continual leakage* model. Furthermore, even though different side-channel attacks leak vastly different types of information, the *amount* of information thus leaked is always assumed to be "bounded". In short, to account for the different side-channel attacks, we will allow the attacker to obtain any polynomial-time computable function of the current internal state of the device in every time-period, as long as the information thus obtained is "bounded". Of course, it is easy to observe that achieving security in this model requires that the secret key (or, its representation) be refreshed periodically.

The second clause of the Micali-Reyzin proposal turns out to be problematic. Roughly speaking, the "only computation leaks information" assumption means that the parts of memory that are not "touched" during a certain time-period do not leak during that time-period. Is it really true that (temporarily) inactive memory does not leak information? Some recently proposed attacks such as the cold-boot attack [HSH$^+$08] leak information from the memory even when no computation is taking place. In addition, various memory architectures leak information when there is no explicit computation going on; for example, a dynamic random-access memory (DRAM) needs to be automatically refreshed every once in a while (just to maintain the state of the memory), hard-disks might leak information because of the electro-magnetic polarizations, and so on. This raises the question:

Can we protect against continual *leakage of information from* all parts of the memory?

In other words, we would like to remove the "only computation leaks" assumption, and allow *all parts of memory* – including the secret key, the random bits generated by the system, and the intermediate bits of computation – to leak during all time-periods. This powerful model of leakage, which we refer to as the *continual memory leakage* model, is the focus of our work. Our main goal will be to construct a variety of cryptographic schemes – public-key and identity-based encryption schemes, and digital signature schemes – that are secure in the continual memory leakage model.

Before going any further, let us review two important lines of work that are vitally related to this discussion.

Previous Work. The first line of work constructs cryptographic schemes that are proven secure in the Micali-Reyzin model, that is, assuming that *only computation leaks information.* There have been a few such constructions, including the work of Dziembowski and Pietrzak who constructed stream ciphers [DP08, Pie09], and the work of Faust et al. who constructed digital signature schemes [FKPR10]. In addition, the recent works of Faust et al. [FRR$^+$10], Juma and Vahlis [JV10], and Goldwasser and Rothblum [GR10] shows how to do general leakage-resilient cryptography assuming access to secure (perfectly leakage-proof) hardware. We elaborate on these results in Section 1.2, but here, we emphasize that there are several very basic cryptographic tasks, such as public-key encryption schemes, for which we do not know of any construction that is secure against continual leakage *even assuming that "only computation leaks".*

A complementary line of work – initiated by Akavia, Goldwasser and Vaikuntanathan [AGV09] – deals with the so-called *bounded memory-leakage model.* Here, the attacker chooses an arbitrary, efficiently computable function f (possibly as a function of the public parameters of the system), and receives the result of f applied to the secret key sk. Clearly, to have some secrecy left, we must restrict the attacker to choose a function f that does not fully reveal the secret. This (apparently necessary) condition has been modeled in several ways, leading to more and more general definitions, by [AGV09], Naor and Segev [NS09] and Dodis, Kalai and Lovett [DKL09]. The important point to note is that the information leakage here is *one-shot* – that is, the total amount of information that the attacker obtains is fixed, *regardless of the amount of computation that is going on within the device.* Many cryptographic schemes were proven secure in these bounded leakage models, including weak pseudorandom functions, signature schemes, and public-key and identity-based encryption schemes [DKL09, NS09, ADW09, KV09]. We elaborate on these in Section 1.2.

The Best of Both Worlds – the Continual Memory Leakage Model. In this work, we propose the continual memory leakage model that combines the best features of both the continual leakage model of [MR04, DP08, Pie09, FKPR10, JV10, GR10], and the bounded memory leakage model of [AGV09, NS09, KV09, ADW09]. In particular, in the continual memory leakage model, the leakage of information is unrestricted in *time* (thus, information leaks continuously over time) and unrestricted in *space* (thus, every bit of the cryptographic device is liable to leak, regardless of whether it is being used in computation or not). Let us pin down a little more precisely what we mean by this.

For concreteness, let us illustrate the continual memory leakage model in the context of digital signature schemes. We view the signing device as consisting of two types of memory:

1. *Public memory* that stores the public verification key, the public randomness used by the system, and the inputs and outputs of the computation.

2. *Secret memory* that stores the secret key, the secret randomness used by the system, and the intermediate steps in the (signature) computations.

The attacker can see the contents of the public memory in its entirety, while (as we will see) he is allowed to obtain a limited amount of information about the secret memory.

More specifically, we partition time to discrete time periods, where during each time period, the signature device may sign many messages. During each time period (and during each signature computation), we allow the attacker to choose an arbitrary (efficiently computable) leakage function, and obtain as a result the leakage function applied to the internal state (namely, the secret key and the internal randomness and the bits of the intermediate computation) of the device. Following [AGV09], the only restriction we place on the leakage functions is that their *total output length during each time-period* is at most a pre-specified fraction of the number of bits of the internal state.[1]

At the end of each time-period, we "update" or "refresh" the secret key. Secret-key update is a randomized procedure that takes as input a secret key sk corresponding to a public key pk, and outputs a uniformly random secret key sk' *for the same public key.* As we argued above, tolerating continual leakage necessitates such an update of the secret key. We emphasize that information leakage could happen during the update of the secret key as well, and such leakage potentially depends on the randomness used for the update procedure (among other things).

We refer the reader to Section 3 for a full-fledged and formal description of the model.

1.1 Our Results

We construct public-key and identity-based encryption schemes, as well as a signature scheme secure in the continual memory leakage model, under the decisional linear assumption over bilinear groups. More specifically, for any (arbitrarily small) constant $\epsilon > 0$, we prove that our schemes are resilient to continual memory leakage, where at each time period, any bounded (poly-time computable) function of the secret-key sk can be leaked, as long as the leakage-rate is at most $\frac{1}{4} - \epsilon$; i.e., the size of the leakage is $(\frac{1}{4} - \epsilon) \cdot |sk|$. Moreover, our schemes are resilient to leakage during each *secret-key update* process, as long as this leakage contains only $O(\log k)$ bits, where k is the security parameter. Furthermore, if we are willing to rely on subexponential hardness of the linear assumption over bilinear groups, then we can tolerate up to $O(k^{\epsilon})$ bits of leakage from each update process. We note that while tolerating a logarithmic number of bits of leakage in *non-continual, one-shot* memory attack model is quite easy (since this leakage can be guessed), this is not the case in the continual leakage model (since in this model, guesses need to be made over and over again).

We emphasize that when leakage occurs during a computation (such as during the update procedure), a bounded function of all the memory content can leak; this includes the secret key, but also all the *randomness* used by the procedure. It is the latter that makes it difficult to tolerate leakage during a computation.

Our Main Results: Public-key and Identity-based Encryption Schemes. The cornerstone of all our results is the construction of a public-key encryption scheme secure against continual memory leakage with the leakage parameters as described above. We then show the following.

- We show how to modify the construction to create an *identity-based encryption* scheme secure against continual memory leakage of the secret keys of individual users (see Section 2.3 and Section 7; also, see below for more details).
- Given an encryption scheme that is resilient to continual memory leakage, we show how to (relatively easily) construct a signature scheme that is resilient to continual memory leakage;

[1]We can in fact extend our results to the more general "leftover entropy" leakage model of Naor and Segev [NS09] which requires only that the secret key has non-trivial min-entropy conditioned on the leakage.

this follows the paradigm of Katz and Vaikuntanathan [KV09] (see Section 2.4 and Section 8.1). The resulting scheme tolerates leakage from the memory, the key-generation and update procedures, but not during the signing process. We then show how to modify this scheme to tolerate leakage from the signing process as well, under additional computational assumptions; see below for details.

We view the encryption (and IBE) schemes as the main results of this paper, and the signature scheme as secondary, for several reasons: First, as mentioned above, it is relatively easy to construct the signature scheme given our encryption scheme (at least without tolerating leakage from the signing algorithm). Second, and more importantly, prior to this work, it was known how to construct signature schemes that are secure in a continual leakage model assuming that *only computation leaks information* [FKPR10]; however, it was not known how to construct encryption schemes that are secure in a continual leakage model, *even with the Micali-Reyzin assumption.*

Finally, we would like to note that our public key encryption scheme gives rise to a symmetric encryption scheme resilient against continual memory leakage (where the secret key contains both the secret and public keys), where two parties can interact over an insecure channel, while *individually and independently* updating their secret keys. In other words, at any point of time, the two communicating parties might have completely different secret keys, and still they will be able to communicate meaningfully. What's more, this requires no interaction whatsoever! We defer an elaborate treatment of this connection, and more, to the full version.

Remarks on the Identity-based Encryption Result. We note that in our identity based encryption scheme, we can only tolerate leakage from the signing keys of the users, but cannot tolerate leakage from the master secret key. On the one hand, one could argue that since the master secret key is a much more appealing target (as it is related to the security of *all* the users), it may attract more attention from attackers and thus deserves more protection. On the other hand, the trusted authority has much more resources than the individual users, and can afford to implement *iron-clad* counter-measures at the implementation level to prevent key leakage. That said, we view the problem of finding identity-based encryption schemes that are resilient to the leakage of the master secret-key as a very interesting open question.

Remarks on the Signature Scheme. In the case of signature schemes, tolerating leakage from the signing process (which is typically a randomized process) is highly non-trivial (this situation should be contrasted with encryption schemes, where the decryption operation is deterministic, and leakage from the decryption process can be interpreted as memory leakage). Indeed, our first signature scheme (that we described above) cannot tolerate any leakage from the signing process.

However, we construct another signature scheme that can tolerate leakage from the signing process relying on two additional assumptions. The first is that *short* non-interactive arguments exist. We note that known constructions of such arguments, were proven to be secure only in the random oracle model [Kil92, Mic00, BG08]. The second assumption is that there exists a family of collision resistant hash functions that map strings in $\{0,1\}^*$ to strings in $\{0,1\}^{\alpha(k)}$, for some $\alpha(k) \in \mathbb{N}$. Given these constructs, the scheme can then tolerate a leakage-rate of $(\frac{1}{4} - \epsilon) \cdot \frac{1}{\alpha(k)}$ (from both memory, and more significantly, also from the signing algorithm). Thus the smaller the number $\alpha(k)$ is, the better the leakage-rate we get, but the stronger the assumption becomes. We note that (as in the original signature scheme) this scheme can also tolerate leakage during each update process, as long as this leakage contains only $O(\log k)$ bits (or k^ϵ bits if we rely on a subexponential hardness assumption).

Improving the Leakage Rate. We note that the main drawback of our schemes is that the leakage-rate that we can tolerate is not optimal. For example, we can only tolerate a leakage-rate of $\frac{1}{4} - \epsilon$ between each update procedure. We don't view this as a serious drawback, since it simply means that we should update our secret key more often.

Moreover, we mention that for signature schemes, we have an additional construction that tolerates leakage-rate of $1 - \epsilon$ between update procedures. unfortunately, this scheme relies on the very strong and non-standard assumption, posed by Valiant [Val08], which assumes the existence of a non-interactive CS proof-of-knowledge. Also, this scheme cannot tolerate any leakage from the update procedure or the signature procedure. Since we view this result as inferior to the others (due to the strong assumption it relies on and due to the fact that the update and signing processes cannot tolerate any leakage), we omit the details of this construction from this version.

Finally, we note that the fact that our schemes can only tolerate $\log k$ bits of leakage from the update procedure (or k^ϵ bits if we rely on subexponential hardness), is a drawback which we cannot overcome. We leave the problem of improving this leakage parameter as an important open problem.

Finally, we mention that one of the technical tools we develop to achieve these results is a new linear algebraic theorem (which is unconditional, and requires no computational assumptions), stating that "random subspaces are leakage-resilient". See Section 2.1 and Section 5 for details.

1.2 Related Work

It should not come as a surprise that the theory of cryptography community has been cognizant of the issue of *inadvertent* information-leakage for quite a while. A number of early results address this issue, although most of them do not explicitly mention side-channel attacks as a motivating factor. Most notably, the line of work on exposure-resilient cryptography, initiated by Rivest and Boyko [Riv97, Boy99], and continued in a number of other works [CDH$^+$00, ISW03, IPSW06], considers simple leakage functions that reveal a subset of the bits of the secret key or the internal memory of the cryptographic device. This line of research culminated in the works of Ishai, Prabhakaran, Sahai and Wagner [ISW03, IPSW06] who show how to "compile" any cryptographic algorithm into one that tolerates such types of leakage. In contrast to these works, that considers leakage functions that act locally, the focus of later works has been to consider powerful leakage functions that can perform some *global computation* on the secret key. In addition, several works on "cryptography with imperfect randomness" [DOPS04] can also be construed as dealing with information leakage.

The modern approach to dealing with leakage starts with the ground-breaking conceptual work of Micali and Reyzin [MR04] who propose to construct and study *formal models* that capture *general types of leakage*. This study has led to two distinct strands of work, which we describe below.

The "Computational Leakage" Model. As described earlier in this section, the work of Micali and Reyzin proposed to study side-channel attacks under the assumption that *only computation leaks information*. In other words, information leakage occurs whenever computation takes place on the secrets; furthermore, the parts of memory that are not involved in computation, during a certain time-period, are not subject to leakage during that time-period. A number of works design cryptographic schemes that are resilient to leakage under this assumption. In particular, the beautiful work of Dziembowski and Pietrzak [DP08, Pie09] constructs stream ciphers secure in this model. Subsequently, Faust et al. [FKPR10] show a construction of signature schemes in this model as well. Notably, there have been no constructs of public-key *encryption* schemes that tolerate continual leakage (even under the Micali-Reyzin axiom).

More recently, Juma and Vahlis [JV10] and Goldwasser and Rothblum [GR10] show how to "compile" any cryptographic algorithm into one that is secure against continual leakage. These works rely on the Micali-Reyzin axiom and make use of a (very simple) *perfectly leakage-proof hardware device* (this device simply samples from a fixed distribution). In particular, these results can be used to come up with leakage-resilient variants of popular cryptographic algorithms such as RSA and AES. In contrast, the focal point of our work is to do away with the Micali-Reyzin axiom, and avoid any type of secure hardware assumption; however, we construct specific cryptographic primitives in this work, as opposed to [JV10, GR10] whose (more ambitious) goal is general leakage-resilient computation.

The "Bounded Memory Leakage" Model. The second line of work considers the bounded leakage model that allows the adversary to obtain a function f of his choice applied the secret key sk, as long as the output $f(sk)$ "does not reveal the entire secret key". This latter condition has been formalized in many different ways, starting from the work of Akavia, Goldwasser and Vaikuntanathan [AGV09] who restrict the leakage function f to have output length $\ell(|sk|) \ll |sk|$.[2] A number of generalizations have been proposed, including the "leftover entropy model" of Naor and Segev who consider the class of all functions f, for which sk has non-trivial min-entropy conditioned on $f(sk)$. An even more general model (and arguably, the most general model along these lines) was given by Dodis, Kalai and Lovett [DKL09], who considered the class of all functions f which are computationally hard to invert. The work on bounded retrieval model, initiated by [CLW06], and explored in [ADW09, ADN$^+$10] is also closely related to this line of research.

Quite significantly, in all these works (unlike this paper), the attacker can only obtain an a-priori bounded amount of leakage from the secret key, regardless of the amount of computation taking place in the system

Other Models and Results. In addition to these works, there have been a small number of models that capture specific classes of side-channel attacks. The work of Faust et al. [FRR$^+$10] show how to compile any cryptographic algorithm into one that resists continual leakage, where the leakage functions are computable by constant-depth AC^0 circuits. Their result assumes the use of a simple leakage-proof hardware device as well. Standaert et al. [PSP$^+$08] construct pseudorandom generators that are secure against specific, naturally occuring, classes of leakage such as the Hamming weight leakage.

Contemporaneous Work of Dodis et al. [DHLAW10]. We finally mention the concurrent work of Dodis et al., which constructs efficient signature schemes, identification schemes and authenticated key agreement that are secure in the continual leakage model (and similarly to us, do not rely on the *only computation leaks information* assumption). Their signature scheme is secure with leakage-rate $\frac{1}{2} - \epsilon$ in each time period, under the SXDH assumption, or $\frac{1}{3} - \epsilon$ under the linear assumption. Unlike this work, they do not address the issue of leakage from the update process and, furthermore, they do not construct public-key or IBE encryption schemes that are secure in the continual leakage model.

[2]In this work, we follow the [AGV09] approach, and restrict the leakage function to have bounded output length *during every time period.*

2 Overview of Our Results

In this section, we provide an extended overview of our results, constructions, proofs and techniques. Our notational conventions are mostly standard. The notation $g^{\mathbf{X}}$, where $\mathbf{X}$ is a matrix and g is a group element (typically a generator), denotes a matrix where $(g^{\mathbf{X}})_{i,j} = g^{(\mathbf{X})_{i,j}}$. We rely on (the matrix form of) the linear assumption in bilinear groups stating that the ensembles $g^{\mathbf{X}}$, $g^{\mathbf{Y}}$ are computationally indistinguishable if $\mathbf{X}, \mathbf{Y}$ have the same dimensions and if $\mathbf{X}$ is a random rank-d matrix ($d \geq 2$) and $\mathbf{Y}$ is a random rank-$(d+t)$ matrix ($t \geq 0$). This assumption is widely assumed to hold even in groups with bilinear maps. We refer the reader to Section 4 for more details on conventions and assumptions.

We start by describing our linear-algebraic tool in Section 2.1 and then move on to describe our public-key encryption scheme in Section 2.2, our identity based encryption scheme in Section 2.3, and finally our signature scheme in Section 2.4.

2.1 Random Subspaces are Leakage Resilient

We first provide an algebraic tool that is crucial to our leakage resilient constructions. More specifically, we give two algebraic theorems that essentially say that random subspaces are resilient to continual leakage. These are information theoretic theorems (i.e., they do not rely on any cryptographic assumption). We believe that these theorems are interesting on their own, and may find further applications.

Consider a random subspace $\mathbf{X}$ of dimension at least 2 of a given linear space. For concreteness, think of $\mathbf{X}$ as a random subspace of $\mathbb{Z}_q^m$ of dimension $\ell \geq 2$. For notational convenience, $\mathbf{X}$ is represented as a rank-ℓ matrix in $\mathbb{Z}_q^{m\times\ell}$, whose columns form a basis of the subspace. A leakage on $\mathbf{X}$ is given in the following form: an arbitrary leakage function f (that must not depend on $\mathbf{X}$) is applied to a random vector $\mathbf{v}$ in the subspace $\mathbf{X}$. We show a bound linking the size of the range of the leakage function f (i.e. the amount of information it reveals on $\mathbf{v}$), to the statistical distance between the pair $(\mathbf{X}, f(\mathbf{v}))$ and the pair $(\mathbf{X}, f(\mathbf{u}))$, where $\mathbf{u}$ is a random vector in $\mathbb{Z}_q^m$. In particular, we show that if the leakage $f(\mathbf{v})$ reveals "bounded" information about $\mathbf{v}$, then the pairs $(\mathbf{X}, f(\mathbf{v}))$ and $(\mathbf{X}, f(\mathbf{u}))$ are statistically close. In the latter pair, the leakage function reveals nothing about the subspace $\mathbf{X}$, and therefore we conclude that applying a "bounded" leakage function (such as a length-restricted leakage function) to a random vector in the subspace $\mathbf{X}$ is leakage resilient in the sense it does not reveal information about $\mathbf{X}$.

We also consider the case where the leakage function is applied to *two* random vectors in $\mathbf{X}$. In other words, the leakage function is applied to a random dimension-2 subspace, $\mathbf{Y}$, of $\mathbf{X}$ (using a matrix notation, $\mathbf{Y} = \mathbf{X} \cdot \mathbf{T}$, where $\mathbf{T}$ is a random rank-2 matrix in $\mathbb{Z}_q^{\ell\times 2}$). In this case, we need to assume that the subspace $\mathbf{X}$ is of dimension $\ell \geq 4$.

More generally, the dimension of $\mathbf{X}$ needs to be at least twice the dimension of the subspace leaked. Namely, if the leakage function is applied to a random subspace of $\mathbf{X}$ of dimension d then we would need $\mathbf{X}$ to be of dimension $\ell \geq 2d$. This requirement stems from the fact that random dimension-d subspaces of $\mathbf{X}$ are (almost) pairwise independent assuming $\mathbf{X}$ is of dimension $\ell \geq 2d$.

We formally state the two theorems below. The first theorem considers the case where the leakage function is applied to one random vector in $\mathbf{X}$ (or a random one-dimensional subspace of $\mathbf{X}$), and the second theorem considers the case where the leakage function is applied to a random two-dimensional subspace of $\mathbf{X}$. In what follows, we denote by dist the statistical distance, and denote by $\mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})$ the set of all rank-2 matrices in $\mathbb{Z}_q^{\ell\times 2}$.

Theorem 2.1. *Let $m, \ell \in \mathbb{N}$, $m \geq \ell \geq 2$ and let q be a prime. Let $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$, let $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^{\ell}$ and let*

$\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$. *Let* $f : \mathbb{Z}_q^m \to W$ *be some function. Then,*

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{r})),\ (\mathbf{X}, f(\mathbf{u}))\big) \le \epsilon,$$

as long as

$$|W| \le q^{\ell-1}\cdot\epsilon^2.$$

Theorem 2.2. *Let* $m, \ell \in \mathbb{N}$, $m \ge \ell \ge 4$ *and let* q *be a prime. Let* $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$, *let* $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})$ *and let* $\mathbf{Y} \xleftarrow{\$} \mathbb{Z}_q^{m\times 2}$. *Let* $f : \mathbb{Z}_q^{m\times 2} \to W$ *be some function. Then,*

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{Y}))\big) \le \epsilon,$$

as long as

$$|W| \le q^{\ell-3}\cdot\epsilon^2.$$

We note that our original proofs resulted in parameters that are slightly worse than those given in Theorems 2.1 and 2.2. The proofs which achieve the improved parameters (as stated in Theorems 2.1 and 2.2) were pointed to us independently by Gil Segev, Yevgeniy Dodis and Daniel Wichs, and follow the proof of the "generalized crooked leftover hash lemma" [DS05, BFO08].

In what follows we give an outline of our original analysis. The outline is for Theorem 2.1, however the outline for Theorem 2.2 is essentially the same.

Proof outline. We note that the (marginal) distributions $f(\mathbf{X}\cdot\mathbf{r})$ and $f(\mathbf{u})$ are identical. We call this component w, and concentrate on the conditional distribution of $\mathbf{X}$ conditioned on w. In the case where $w = f(\mathbf{u})$, the conditional distribution of $\mathbf{X}$ is of course uniform. We will prove that this is also the case when $w = f(\mathbf{X}\cdot\mathbf{r})$ (up to ϵ statistical distance), and the theorem will follow.

The proof uses the fact that for almost all fixed values of $\mathbf{r}_1, \mathbf{r}_2$, it holds that $\mathbf{X}\cdot\mathbf{r}_1$, $\mathbf{X}\cdot\mathbf{r}_2$ are independent. Consider the distribution of $\mathbf{X}$ conditioned on $f(\mathbf{X}\cdot\mathbf{r}) = w$. The probability of a specific $\mathbf{X}$ is proportional to the number of vectors $\mathbf{r}$ for which $f(\mathbf{X}\cdot\mathbf{r}) = w$. Since this can be written as a sum of (almost) pairwise independent random variables, it follows that the number of such $\mathbf{r}$'s is well concentrated around its mean value. Therefore, the said conditional distribution of $\mathbf{X}$ is close to uniform.

For more details, see Section 5.

2.2 A Continual-Leakage Secure Encryption Scheme

We next present our public-key encryption schemes $\mathcal{L}$ and $\mathcal{L}^*$ that are secure against continual leakage attacks. These schemes are parameterized by a polynomially bounded integer ℓ that allows for a tradeoff between the sizes of keys and ciphertexts, and the maximal tolerable leakage rate. The scheme $\mathcal{L}$ is secure under the decisional linear assumption in bilinear groups, as long as at most $1/2 - o(1)$ fraction of the memory leaks between any two consecutive secret-key updates. The security of $\mathcal{L}^*$ relies on the less standard SXDH assumption (described below), however it can tolerate leakage of almost the entire secret key (in particular, a $1 - o(1)$ fraction thereof) between consecutive key-updates. See below for a more detailed explanation on parameters. The scheme $\mathcal{L}^*$ is somewhat simpler than $\mathcal{L}$ and one can think of it as a "toy example" for $\mathcal{L}$. However, since it is based on a less standard assumption, we will focus our attention mostly on $\mathcal{L}$. We stress that the analyses of $\mathcal{L}$ and $\mathcal{L}^*$ are very similar and differ almost only in the use of the correct linear algebraic tool (Theorem 2.2 is used for $\mathcal{L}$ and Theorem 2.1 is used for $\mathcal{L}^*$).

As we described above, the amount of tolerable leakage is measured by the ratio between the total number of bits being leaked at each time period and the total amount of "secret memory" being used at that time period. We distinguish between the leakage rates in 3 phases of the algorithm, and thus characterize the scheme's leakage resiliency using 3 parameters: (1) the leakage rate during the key generation, denoted by ρ_G, which is the total amount of leakage during the key generation divided by the total amount of secret randomness required for this procedure; (2) the leakage rate during key updates, denoted by ρ_U, which is the total amount of leakage during a key-update procedure, divided by the size of the secret-key (which is a "secret input" to the procedure) and the size of the secret randomness used; and (3) the memory leakage rate between updates, denoted by ρ_M, which is the leakage during "normal operation" of the scheme, this is the total number of bits leaked between consecutive updates divided by the size of the secret-key (which is the only secret information during this phase).

We show that our schemes are resilient to leakage of constant rate from the memory and very small leakage (of sub-constant rate) from the key-update procedure. The same amount of leakage can also be tolerated in the key-generation procedure. Recalling our notation, this corresponds to a case where $\rho_M = \Omega(1)$, $\rho_G = \rho_U = o(1)$.

The leakage parameters for $\mathcal{L}[\ell]$. In the scheme $\mathcal{L}[\ell]$, the value of ρ_M for a general ℓ can be as high as $\frac{\ell-6-\gamma}{2\ell}$, for all $\gamma > 0$. For example, taking $\ell = 12$ guarantees resiliency to leakage rate of $1/4 - \gamma$, while the keys and ciphertexts only contain a constant number of group elements. Taking ℓ to be asymptotically increasing, on the other hand, makes the keys and ciphertexts sizes asymptotically increase but makes the tolerable leakage rate $1/2 - o(1)$ (as mentioned above, based on the SXDH assumption, a rate of $1 - o(1)$ can be achieved).

The value of ρ_U for general ℓ is $\omega\left(\frac{\log k}{\ell \log p}\right)$, where k is the security parameter and p is the order of the group relative to which we work. We cannot provide an explicit expression for ρ_U, but rather we show that for all $c > 0$, the scheme is resilient in the case where $\rho_U = \frac{c \cdot \log k}{\ell \log p}$. This translates to $O(\log k)$ bits of leakage during each update procedure. In fact, our argument can be generalized in a straightforward manner to imply resiliency of $\rho_U = \Omega\left(\frac{\log T(k)}{\ell \log p}\right)$ (corresponding to $\Omega(\log T(k))$ bits of leakage during each update procedure) if one is willing to assume that the linear assumption is hard for adversaries that run in time (roughly) $T(k)$. Similar techniques show that the same amount of leakage (i.e., any $O(\log k)$ bits under a standard assumption or $\Omega(\log T(k))$ bits under a strengthened one) can be tolerated from the key-generation procedure, implying $\rho_G = \omega\left(\frac{\log k}{\ell \log p}\right)$ in this case as well.

To put these results in context, we recall that tolerating a leakage of a logarithmic number of bits in *non-continual* memory attacks, while not completely trivial, is fairly simple: the leakage value can be guessed and then the adversary has to be simulated in order to make sure that the value guessed is indeed correct (since a success in the security game for encryption is not verifiable). It is important to notice that this is no longer the case in a model with memory leakage and key updates. Here, even a *one time* leakage of logarithmic size is not necessarily trivial: the reduction cannot simulate the adversary by itself to check the validity of the leakage value guessed. Therefore, even just logarithmic leakage from the key-generation, for example, is non-trivial. The situation becomes even worse when considering leakage from the key-update procedure, since the reduction now needs to make guesses over and over again, and will almost for sure guess wrong at some point. Therefore we view the ability to withstand even such minimal amounts of leakage as an accomplishment.

The Scheme $\mathcal{L}[\ell]$. Let us overview the main ideas behind the structure of our scheme. We want to use the fact that under the linear assumption, random rank-2 matrices in the exponent are indistinguishable from random rank-3 matrices.

We start by setting the public-key to $g^{\mathbf{A}}$, where $\mathbf{A}$ is a random $2 \times \ell$ matrix in $\mathbb{Z}_p^{2\times\ell}$, and where p is the order of the group element g. The ciphertexts are of the form $g^{\mathbf{v}^T}$, where $\mathbf{v} \in \mathbb{Z}_p^{\ell}$. Namely, the public-key and the ciphertext together form a $(3 \times \ell)$ matrix in the exponent. Our scheme has the property that this $(3 \times \ell)$ matrix is random rank-2 for encryptions of 0, and random rank-3 for encryptions of 1, and thus we can use the indistinguishability argument to achieve (semantic) security. To get this property, our encryption algorithm operates as follows: To encrypt 0 set $g^{\mathbf{v}^T}$ to be a linear combination of the rows of $g^{\mathbf{A}}$; to encrypt 1 set $g^{\mathbf{v}^T}$ to be a random vector. One can see that the resulting distributions are statistically close to the prescribed ones. Thus, security is achieved.

The next question, of course, is figuring out how to decrypt such ciphertexts. We notice that any non-zero vector $\mathbf{y}$ in the kernel of $\mathbf{A}$ can be used to decrypt: simply compute $g^{\mathbf{v}^T\cdot\mathbf{y}}$ and see if the result equals to g^0. This will always be the case for encryptions of 0 and will only happen with negligible probability for encryptions of 1. This suggests that such $\mathbf{y}$ can be used as a secret-key. However, it is not clear how such a secret-key can be safely updated.

In order to allow for updates, we use a secret key of the form $g^{\mathbf{Y}}$, where $\mathbf{Y}$ has two column vectors, $\mathbf{Y} = [\mathbf{y}_1 \| \mathbf{y}_2] \in \mathbb{Z}_p^{\ell\times 2}$, and $\mathbf{y}_1, \mathbf{y}_2 \in \mathbb{Z}_p^{\ell}$ are random vectors in the kernel of $\mathbf{A}$. The fact that the vectors are given in the exponent means that we cannot compute the product $g^{\mathbf{v}^T\cdot\mathbf{y}_i}$ as suggested before. We thus work over groups with bilinear maps, which enable computing $e(g,g)^{\mathbf{v}^T\cdot\mathbf{Y}}$ instead.

The key update operation is done by "rotating" the matrix $\mathbf{Y}$: sample a random 2×2 full rank matrix $\mathbf{R} \in \mathbb{Z}_p^{2\times 2}$, and set the new secret key to $g^{\mathbf{Y}\cdot\mathbf{R}}$. Intuitively, since everything is done in the exponent, the update operation is indistinguishable from sampling a fresh random secret-key, which turns out to be a useful property.

The scheme $\mathcal{L}[\ell]$ is formally presented in Figure 1.

Encryption scheme $\mathcal{L}[\ell]$

Parameters. The scheme is parameterized by groups $\mathbb{G}, \mathbb{G}_T$ of prime order p, such that there exists a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let g be a generator of $\mathbb{G}$ (and so $e(g,g)$ is a generator for $\mathbb{G}_T$). An additional parameter $\ell \geq 7$ is polynomial in the security parameter. Setting different values for ℓ will enables a tradeoff between efficiency and the rate of tolerable leakage.

Key-generation. The key-generation algorithm samples $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_p^{2\times\ell}$ and $\mathbf{Y} \xleftarrow{\$} \ker^2(\mathbf{A})$ (i.e. $\mathbf{Y} \in \mathbb{Z}_p^{\ell\times 2}$) (note that this is easy to sample since $\mathbf{A}$ is known). It sets $pk = g^{\mathbf{A}}$ and $sk = g^{\mathbf{Y}}$.

Key-update. To update the secret key $sk = g^{\mathbf{Y}} \in \mathbb{G}^{\ell\times 2}$, sample $\mathbf{R} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{2\times 2})$ and then set $sk' = g^{\mathbf{Y}\cdot\mathbf{R}}$.

Encryption. Given a public-key $pk = g^{\mathbf{A}} \in \mathbb{G}^{2\times\ell}$, encrypting the bit 0 is done by sampling $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^2$ and setting the output ciphertext to $c = g^{\mathbf{r}^T\cdot\mathbf{A}}$. Encrypting the bit 1 is done by setting the ciphertext to $c = g^{\mathbf{u}^T}$ where $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^{\ell}$.

Decryption. Given a ciphertext $c = g^{\mathbf{v}^T}$ and a secret-key $sk = g^{\mathbf{Y}}$, the decryption process computes $e(g,g)^{\mathbf{v}^T\cdot\mathbf{Y}}$. If the result is equal to $e(g,g)^{\mathbf{0}}$, then output 0 and otherwise output 1.

Figure 1: Encryption scheme in the CML model.

Theorem 2.3. *Under the linear assumption, for every $\ell \geq 7$, the encryption scheme $\mathcal{L}[\ell]$ (described* *in Figure 1) has the following security guarantee: For all constants $\gamma, c > 0$, $\mathcal{L}[\ell]$ is secure in the continual leakage model with leakage rate*

$$(\rho_G, \rho_U, \rho_M) = \left(\frac{c \cdot \log k}{4\ell \cdot \log p}, \frac{c \cdot \log k}{(2\ell+4) \cdot \log p}, \frac{\ell - 6 - \gamma}{2\ell} \right) .$$

We refer the reader to Section 6 for details. The proof overview of Theorem 2.3 can be found in Section 6.2, and the formal proof can be found in Section 6.3.

The Scheme $\mathcal{L}^*[\ell]$: a simpler scheme with better guarantees under the SXDH assumption. It was recently pointed out to us by Daniel Wichs that a simpler variant of our scheme, which we initially used for explanatory purposes, can in fact be proven secure under the symmetric external Diffie-Hellman (SXDH) assumption in bilinear groups. Moreover, the simplified scheme enjoys better leakage resilience – it achieves security against a leakage rate of $1 - o(1)$ from memory between secret-key updates (the amount of sustainable leakage during the key generation and key updates remains the same as in our original scheme).

Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be groups of equal prime order p such that there exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. The SXDH assumption says that the decisional Diffie-Hellman (DDH) assumption holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$. Translating into matrix notation, as we use in our original scheme, the SXDH assumption says that it is hard to distinguish random rank-1 matrices from random rank-2 matrices (of the same dimensions) in the exponent, in both the groups $\mathbb{G}_1$ and $\mathbb{G}_2$.

Since the SXDH assumption is not as widely used as the linear assumption, and to avoid repeating essentially the same proof twice, we present our SXDH based scheme $\mathcal{L}^*[\ell]$ in Figure 2 and state its properties below, without proof.

Encryption scheme $\mathcal{L}^*[\ell]$

Parameters. The scheme is parameterized by groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order p, such that there exists a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Let g_1, g_2 be generators of $\mathbb{G}_1, \mathbb{G}_2$ respectively (and so $e(g_1, g_2)$ is a generator for $\mathbb{G}_T$). An additional parameter $\ell \geq 3$ is polynomial in the security parameter. Setting different values for ℓ will enables a tradeoff between efficiency and the rate of tolerable leakage.

Key-generation. The key-generation algorithm samples $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_p^\ell$ and $\mathbf{y} \xleftarrow{\$} \ker(\mathbf{a})$ (i.e. $\mathbf{y} \in \mathbb{Z}_p^\ell$) (note that this is easy to sample since $\mathbf{a}$ is known). It sets $pk = g_1^{\mathbf{a}}$ and $sk = g_2^{\mathbf{y}}$.

Key-update. To update the secret key $sk = g_2^{\mathbf{y}} \in \mathbb{G}_2^\ell$, sample $r \xleftarrow{\$} \mathbb{Z}_p$ and set $sk' = g_2^{r\mathbf{y}}$.

Encryption. Given a public-key $pk = g_1^{\mathbf{a}} \in \mathbb{G}_1^\ell$, encrypting the bit 0 is done by sampling a number $r \xleftarrow{\$} \mathbb{Z}_p$ and setting the output ciphertext to $c = g_1^{r\mathbf{a}}$. Encrypting the bit 1 is done by setting the ciphertext to $c = g_1^{\mathbf{u}^T}$ where $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^\ell$.

Decryption. Given a ciphertext $c = g_1^{\mathbf{v}^T}$ and a secret-key $sk = g_2^{\mathbf{y}}$, the decryption process computes $e(g_1, g_2)^{\mathbf{v}^T \cdot \mathbf{y}}$. If the result is equal to $e(g_1, g_2)^0$, then output 0 and otherwise output 1.

Figure 2: Encryption scheme in the CML model, based on the SXDH assumption.

Theorem 2.4. *Under the* SXDH *assumption, for every $\ell \geq 3$, the encryption scheme $\mathcal{L}^*[\ell]$ (described in Figure 2) has the following security guarantee: For all constants $\gamma, c > 0$, $\mathcal{L}^*[\ell]$ is secure in the continual leakage model with leakage rate*

$$(\rho_G, \rho_U, \rho_M) = \left(\frac{c \cdot \log k}{2\ell \cdot \log p}, \frac{c \cdot \log k}{(\ell+2) \cdot \log p}, \frac{\ell - 2 - \gamma}{\ell} \right) .$$

The proof of this theorem is almost identical to that of Theorem 2.3 above (see Section 6 for more details and proof of the latter), except for the usage of Theorem 2.1 instead of Theorem 2.2 inside the proof.

2.3 A Continual-Leakage Secure IBE Scheme

We present an identity based encryption (IBE) scheme that is secure against continual leakage. This scheme can be seen as a combination of our encryption scheme described above and the d-linear IBE scheme of Brakerski and Kalai [BK10].

We recall that in the context of IBE there are two types of "entities" holding secrets: a trusted authority holding a *master secret-key* that enables producing specific secret-keys per user; and individual users, each holding their own secret-key. In this work, we only allow leakage from the memory of the individual users. For a more detailed discussion on identity based encryption schemes and the associated security notions under continual leakage, we refer the reader to Section 7.

In our construction, we consider an IBE scheme where the set of identities is $\{0,1\}^m$. The public parameters of the scheme is a set of $2m+1$ matrices of dimension 2×2 in the exponent ($g^{\mathbf{A}_0}$, $\{g^{\mathbf{A}_{i,b}}\}_{i \in [m], b \in \{0,1\}}$). Identity $id \in \{0,1\}^m$ is associated with the matrix $\mathbf{A}_{id} = [\mathbf{A}_0 \| \mathbf{A}_{id_1} \| \cdots \| \mathbf{A}_{id_m}]$. Specifically, we use $g^{\mathbf{A}_{id}}$ in order to encrypt messages for id. To decrypt, the user corresponding to id uses the secret-key $sk_{id} {:=} g^{[\mathbf{y}_1 \| \mathbf{y}_2]}$, where $\mathbf{y}_1, \mathbf{y}_2 \xleftarrow{\$} \ker(\mathbf{A}_{id})$. In other words, each user is associated with a pair of keys corresponding to $\mathcal{L}[\ell]$, defined by its identity and the public parameters (recall that $\mathcal{L}[\ell]$ is our public-key encryption scheme defined in Section 2.2). We notice that the trusted authority only needs to know $msk = \mathbf{A}_0$ in order to produce secret-keys for all users. The encryption and decryption algorithms are similar to the ones in $\mathcal{L}[\ell]$.

The proof of security is by reduction to the security of $\mathcal{L}[\ell]$ for $\ell = 2m+2$. Specifically we prove that our scheme is *selectively secure* with continual leakage (then, standard techniques can be used to achieve other, stronger, notions of security). In selective security we assume that the attacker decided on the identity it wants to attack before seeing the public-parameters. Let id^* be that identity. This enables generating the public parameters such that all matrices $\mathbf{A}_{i,1-id_i^*}$ (i.e. all matrices that do not "play" in $\mathbf{A}_{id^*}$) are explicitly known, and a public key for $\mathcal{L}[2m+2]$ is "embedded" in the public parameters as the matrix $g^{\mathbf{A}_{id^*}}$. This enables the adversary to know everything about all other identities (since it is sufficient to explicitly know one sub-matrix of $\mathbf{A}_{id}$ in order to generate a corresponding secret-key) while knowing nothing about id^*. Thus, an adversary that breaks the security of id^*, even with continual leakage, actually breaks the security of $\mathcal{L}[2m+2]$.

For full details on our construction, see Section 7.

2.4 A Continual-Leakage Secure Signature Scheme

We show how to use any encryption scheme secure with continual leakage (such as the encryption scheme $\mathcal{L}[\ell]$ presented in Section 2.2) to construct a signature scheme that is secure with continual leakage. When discussing continual leakage in signature schemes, we must also take into account the signing operation. This operation involves the secret-key and additional secret randomness and may potentially leak extra information. Therefore, we are concerned with an additional parameter of leakage resiliency, namely the leakage rate during the signing process, which we denote by ρ_S. This is defined as the total amount of information leaked during signing, divided by the total size of the secret-key and secret-randomness used by the signing procedure. For a more detailed discussion and for the full constructions, see Section 8.

Our first construction is essentially applying the paradigm of Katz and Vaikuntanathan [KV09] to our encryption scheme, to obtain a signature scheme that preserves the leakage guarantees of the underlying encryption scheme, but does not allow any leakage during the signing process ($\rho_S = 0$). The signing key now contains a secret-key sk for the encryption scheme, and the verification key contains the public-key pk, as well as an additional public-key pk' for a (not necessarily leakage-resilient) public-key encryption scheme, and a common random string crs for a non-interactive simulation-sound zero-knowledge (NIZK) proof system. A signature for a message m contains an encryption of sk using pk' and a NIZK proof that the contents of the ciphertext are indeed a valid secret-key for the leakage resilient encryption scheme. The dependence on the message m comes from properly defining the language for the simulation-sound NIZK proof, as is done in [KV09].

We show that we can use a successful forger for this scheme to break the security of the leakage resilient encryption scheme. In the security reduction, we simulate all NIZK proofs and sample pk' together with a respective secret-key sk'. We can then provide valid looking signatures for a successful forger without it being able to tell the difference. Then we can take its successful forged signature, which must contain an encryption of a valid secret-key for the leakage resilient encryption scheme, decrypt it using sk', and obtain a valid sk that enables breaking the security of the leakage resilient encryption scheme. For more details on this construction, see Section 8.1.

Our second construction does tolerate leakage from the signing algorithm. In this case, the signatures cannot be simulated. However, we construct the scheme such that each signature (information theoretically) reveals very little information about the secret key, and therefore can be obtained as leakage queries. To this end, we rely on the existence of *short* non-interactive arguments (see Definition 4.3). We note that all known constructions of the latter, were proven to be secure only in the random oracle model [Kil92, Mic00, BG08]. For more details on this construction, see Section 8.2.

3 A Model for Continual Memory Leakage

In this section, we discuss several ways to model cryptography with continual memory leakage. We begin by explaining the basic principles that underlie the choice of our model, and based on these principles, derive formal models for secure encryption and digital signatures in the context of continual memory leakage. While the discussion below is rather general and could apply to many cryptographic primitives, we use digital signatures as the running example.

Suppose that we have some cryptographic application (e.g., a signature scheme) that performs computation on some secret information, and runs in an environment where an adversary may be able to extract some information about the internal secret state by "probing" the memory (e.g., the locations where the signing key is stored). This can be done either when the system is idle or during the computation itself (e.g., during the execution of the signing algorithm). If the adversary's probing is not limited in any way, then it can "copy" the entire internal state of the device, thus breaking any reasonable security requirement (e.g., if the adversary attains the signing key, it can then trivially sign any message). This leads to an understanding that some restrictions need to be imposed on the probing process.

A first such model, suggested by Micali and Reyzin [MR04], introduced the "axiom" that *only computation leaks information.* This axiom asserts that an area in memory that is not accessed at a certain time period, cannot be probed during that time period. This enables "shielding" parts of the secret-state from leaking at certain time periods by not letting the adversary access them at that time. In particular, if at every time-slot, some (different) part of the secret is shielded, then this shielded part is uncorrelated with the leakage that occurred during this time-slot, and therefore can be used to "refresh" the remainder of the secret. Making this assumption, therefore, suggests a methodology for coping with leakage even when it is continual. However, while this axiom may be justifiable in certain situations, there are known attacks that do not adhere to it. Specifically, these include "memory attacks", such as the cold-boot attack [HSH$^+$08], where the content of the memory is "dumped", almost entirely, and probed by the attacker.

One approach to address the issue of "memory attacks" mentioned above, is to allow the secret state of the scheme (e.g., the signing-key) to be leaked in an arbitrary (efficient) way, so long as the amount of information revealed to the attacker is bounded. Hence, we do not assume that parts of the state are untouched during certain periods of time, only that the adversary learns a bounded amount of information – which seems necessary as otherwise the entire secret state can be leaked.

However, since we allow the leakage to be continual (letting the adversary learn additional bounded amount of information at any time period), then the secret-key must be periodically *refreshed*,[3] since otherwise it will eventually be completely leaked. Moreover, we must allow *deletions*, since otherwise the adversary can choose to (gradually) leak the initial secret-state from which everything can be derived. In such case, refreshing the state is useless because the adversary will leak from the original state and not the refreshed one. For example, in the case of digital signatures, the adversary will choose to leak bits from the initial signing key at every time slot, and eventually will gain the ability to sign messages of its choosing. Since the public verification key does not change, the initial key must still be "valid" for signing, even if it has so far been refreshed. It seems that the only way to prevent such an attack, without changing the leakage model (i.e., while still allowing the adversary to select an arbitrary efficient leakage function at every time slot), is to allow information to be safely deleted.

We therefore, consider the model of *continual memory leakage with deletions* (and refer to it as the CML model). A computation in this model can be thought of as a Turing machine that has a special "secret tape", with content s. All other tapes of the machine are readable by the adversary at all times. The deletion operation is implicit in the model as cells in the tape of a Turing machine can be deleted/rewritten. At any time period, the adversary can obtain some "bounded" information $f(s)$ about the secret state s.

There are several ways to define "bounded" in this context:[4] The first, is using the definition of Akavia, Goldwasser, and Vaikuntanathan [AGV09], which says that $f(s)$ is "bounded" if $\frac{|f(s)|}{|s|} \leq \rho$ for some predefined bound $\rho < 1$. The second, is the slightly more general definition due to Naor and Segev [NS09], which says that $f(s)$ is "bounded" if s high min-entropy, conditioned on $f(s)$. The third, is the most general one due to Dodis *et al.* [DKL09], which says that $f(s)$ is "bounded" if f is hard-to-invert with probability 2^{-t}, for some parameter $t = t(k)$.

In our definitions below (both in Section 3.1, and in Section 3.2), we use the [AGV09] definition for "bounded" functions, though all of our results hold also with the slightly stronger definition of [NS09]; we choose to present the definitions using the former only for the sake of simplicity. We note in Section 8.2, we even make use of the fact our encryption scheme from Section 6 is secure even if we use the slightly stronger definition of [NS09] for "bounded" functions. Finally, we note that we do not know how to prove our results using the [DKL09] definition for "bounded" functions, and leave this as an important open problem.

3.1 Public-Key Encryption in the CML Model

An encryption scheme in the CML model contains the following PPT algorithms.

- *Key-generation.* Takes as input 1^k (where k is the security parameter) and produces a secret-

[3]This should be done without modifying the public key.

[4]In what follows, think of the secret key as being truly random.

key sk and a public-key pk. We denote this by $(sk, pk) \leftarrow \mathsf{Gen}(1^k)$.

- *Encryption.* Takes as input a public-key pk and a message m and outputs a ciphertext c. We denote this by $c \leftarrow \mathsf{Enc}_{pk}(m)$.
- *Decryption.* Takes as input the secret-key sk and a ciphertext c and outputs a message m'. We denote this by $m' \leftarrow \mathsf{Dec}_{sk,pk}(c)$.
- *Key-update.* Takes as input a secret-key sk and outputs a "refreshed" secret-key sk' such that $|sk'| = |sk|$.[5] We denote this by $sk' \leftarrow \mathsf{Update}_{pk}(sk)$.

The first three algorithms are identical to the standard definition of public-key encryption schemes. The key-update algorithm is an addition that allows for continual leakage resilience.

The correctness requirement is that for all m and a polynomially bounded $t \in \mathbb{N}$, setting $(sk_0, pk) \leftarrow \mathsf{Gen}(1^k)$, $sk_i \leftarrow \mathsf{Update}_{pk}(sk_{i-1})$ for $i \in [t]$, $c \leftarrow \mathsf{Enc}_{pk}(m)$, $m' \leftarrow \mathsf{Dec}_{sk_t,pk}(c)$, it holds that $m = m'$ with all but negligible probability (where the probability is over all randomness in the experiment).

We next define semantic security in the CML model. Formally, leakage in this model is associated with three leakage parameters (ρ_G, ρ_U, ρ_M), where ρ_G bounds the leakage rate from the key-generation process, ρ_U bounds the leakage rate from the update process, and ρ_M is a "global" (relative) memory leakage bound that is enforced between key updates. Taking $\rho_G = \rho_U = 0$ corresponds to allowing leakage only from the memory and not from the key-generation or update processes.

Definition 3.1. *An encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Update})$ *is semantically secure in the* CML *model with leakage rate* (ρ_G, ρ_U, ρ_M), *if any* PPT *adversary succeeds in the following game with probability* $1/2 + \mathrm{negl}(k)$.

1. Initialize. *The adversary specifies a key-generation leakage circuit f with $|f(r,p)| \le \rho_G \cdot |r|$ for all r, p. The challenger chooses "secret randomness" r and "public randomness" p, generates $(sk_0, pk) \leftarrow \mathsf{Gen}(1^k; r, p)$, sends $(pk, p, f(r,p))$ to the adversary, and sets $i := 0$ and $L_0 := |f(r,p)|$.*
2. Leakage and updates. *The forger makes queries of the following type:*
   - *Update queries (update, f), where f is a circuit with $|f(sk, r, p)| \le \rho_U \cdot (|sk| + |r|)$ for all sk, r, p. The challenger chooses "secret randomness" r and "public randomness" p, and computes $sk_{i+1} := \mathsf{Update}_{pk}(sk_i; r, p)$. If $L_i + |f(sk_i, r, p)| \le \rho_M \cdot |sk_i|$ then the challenger returns $(p, f(sk_i, r, p))$ to the adversary, sets $i := i + 1$, and sets $L_{i+1} := |f(sk_i, r, p)|$. Otherwise, the challenger aborts.*
   - *Leakage queries (leak, f), where f is a circuit. If $L_i + |f(sk_i)| \le \rho_M \cdot |sk_i|$ then the challenger returns $f(sk_i)$ to the adversary and sets $L_i := L_i + |f(sk_i)|$. Otherwise, the challenger aborts.*
3. Challenge. *The adversary sends two messages m_0, m_1 to the challenger. The challenger flips a coin $b \xleftarrow{\$} \{0,1\}$ and computes $c \leftarrow \mathsf{Enc}pk(m_b)$. It sends c to the adversary.*
4. Finish. *The adversary outputs a "guess" $b' \in \{0,1\}$.*

The adversary **succeeds** *if $b' = b$.*

[5]This requirement is in order to prevent the secret-key from "blowing up".

3.2 Digital Signatures in the CML Model

A digital signature scheme in the CML model comprises the following PPT algorithms:

- The *key-generation algorithm* Gen takes as input the security parameter 1^k, and outputs a pair of signing/verification keys sk, vk; we denote this by $(sk, pk) \leftarrow \mathsf{Gen}(1^k)$.
- The *signing algorithm* Sign takes as input a signing key sk, a verification key vk, and a message m in some message space $\mathcal{M}$, and outputs a signature σ. We denote this by $\sigma \leftarrow \mathsf{Sign}_{sk,vk}(m)$.
- The deterministic *verification algorithm* Ver takes as input the verification key vk, a message $m \in \mathcal{M}$ and a signature σ and outputs a bit b denoting acceptance or rejection. We denote this by $b := \mathsf{Ver}_{vk}(m, \sigma)$.
- The *key-update algorithm* Update takes as input a signing key sk and a verification key vk, and outputs a "refreshed" signing key sk' with $|sk'| = |sk|$. We denote this by $sk' \leftarrow \mathsf{Update}_{vk}(sk)$.

The correctness requirement is that for all $m \in \mathcal{M}$ and any polynomial t, setting $(sk_0, vk) \leftarrow \mathsf{Gen}(1^k)$ and then computing $sk_i \leftarrow \mathsf{Update}_{vk}(sk_{i-1})$ for $i \in [t]$, we have $\mathsf{Ver}_{vk}(m, \mathsf{Sign}_{sk_t,vk}(m)) = 1$ except with negligible probability. (where the probability is over all randomness in the experiment).

Next we define security in the CML model. We consider the standard notion of existential unforgeability under adaptive chosen message attack, but in the presence of leakage. Allowable leakage is specified by four parameters ρ_G, ρ_S, ρ_U that specify, respectively, the (relative) leakage allowed from the key-generation, signing, and update processes, as well as a "global" (relative) memory leakage bound ρ_M that is enforced between key updates.

Definition 3.2. *Signature scheme* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ *is existentially unforgeable under adaptive chosen message attack in the* CML *model, with leakage rate* $(\rho_G, \rho_U, \rho_S, \rho_M)$*, if any* PPT *forger succeeds in the following game with only negligible probability.*

1. Initialize. *The forger specifies a circuit* f *with* $|f(r, p)| \leq \rho_G \cdot |r|$ *for all* r, p. *The challenger chooses "secret randomness"* r *and "public randomness"* p*, generates* $(sk_0, vk) \leftarrow \mathsf{Gen}(1^k; r, p)$*, sends* $(vk, p, f(r, p))$ *to the forger, and sets* $i := 0$ *and* $L_0 := |f(r, p)|$.
2. Signatures, leakage, and updates. *The forger makes queries of the following types:*
   - *Update queries* (update, f)*, where* f *is a circuit with* $|f(sk, r, p)| \leq \rho_U \cdot (|sk| + |r|)$ *for all* sk, r, p. *The challenger chooses "secret randomness"* r *and "public randomness"* p*, and computes* $sk_{i+1} := \mathsf{Update}_{vk}(sk_i; r, p)$. *If* $L_i + |f(sk_i, r, p)| \leq \rho_M \cdot |sk_i|$ *then the challenger returns* $(p, f(sk_i, r, p))$ *to the forger, sets* $i := i + 1$*, and sets* $L_{i+1} := |f(sk_i, r, p)|$. *Otherwise, the challenger aborts.*
   - *Signing queries* (sign, m, f)*, where* f *is a circuit with* $|f(sk, r, p)| \leq \rho_S \cdot (|sk| + |r|)$ *for all* sk, r, p. *The challenger chooses "secret randomness"* r *and "public randomness"* p*, and computes* $\sigma := \mathsf{Sign}_{sk_i,vk}(m; r, p)$. *If* $L_i + |f(sk_i, r, p)| \leq \rho_M \cdot |sk_i|$ *then the challenger returns* $(\sigma, p, f(sk_i, r, p))$ *to the forger and sets* $L_i := L_i + |f(sk_i, r, p)|$. *Otherwise, the challenger aborts.*
   - *Leakage queries* (leak, f)*, where* f *is a circuit. If* $L_i + |f(sk_i)| \leq \rho_M \cdot |sk_i|$ *then the challenger returns* $f(sk_i)$ *to the forger and sets* $L_i := L_i + |f(sk_i)|$. *Otherwise, the challenger aborts.*
3. Finish. *Assuming the challenger did not abort, the forger outputs* (m^*, σ^*).

The forger **succeeds** *if it never made the query* (sign, m^*)*, and* $\mathsf{Ver}_{vk}(m^*, \sigma^*) = 1$.

4 Preliminaries

We use the following notational conventions. Bold uppercase denotes matrices ($\mathbf{X} \in \mathbb{Z}_q^{n\times k}$) and bold lowercase denotes vectors ($\mathbf{x} \in \mathbb{Z}_q^n$). All vectors are column vectors, row vectors are denoted by $\mathbf{x}^T$. For a scalar (usually a group element) g and a matrix $\mathbf{X} \in \mathbb{Z}^{k\times n}$ (or a vector, as a special case), we let $g^{\mathbf{X}}$ denote a $k \times n$ matrix such that $(g^{\mathbf{X}})_{i,j} = g^{(\mathbf{X})_{i,j}}$.

Let X be a probability distribution over a domain S, we write $x \overset{\$}{\leftarrow} X$ to indicate that x is sampled from the distribution X. The uniform distribution over a set S is denoted $U(S)$. We use $x \overset{\$}{\leftarrow} S$ as abbreviation for $x \overset{\$}{\leftarrow} U(S)$. For any function f with domain S we let $f(X)$ denote the random variable (or corresponding distribution) obtained by sampling $x \overset{\$}{\leftarrow} X$ and outputting $f(x)$. The *min-entropy* of a (discrete) random variable X is $\mathbf{H}_\infty(X) = \min_{x\in S}\{-\log \Pr[X = x]\}$.

We write $\mathrm{negl}(k)$ to denote an arbitrary *negligible* function, i.e. one that vanishes faster than the inverse of any polynomial.

The *statistical distance* between two distributions X, Y (or random variables with those distributions) over a common domain S is defined as $\mathsf{dist}(X, Y) \doteq \max_{A\subseteq S} |\Pr[X \in A] - \Pr[Y \in A]|$. Two ensembles $X = \{X_k\}_k$, $Y = \{Y_k\}_k$ are $\epsilon = \epsilon(k)$-*close* if the statistical distance between them is at most $\epsilon(k)$, this is also denoted by $X \overset{\epsilon}{\equiv} Y$ (where $X \equiv Y$ means that X, Y are identically distributed). They are called *statistically indistinguishable* if $\epsilon(k) = \mathrm{negl}(k)$. An ensemble $X = \{X_k\}_k$ over domains $S = \{S_k\}_k$ is $\epsilon = \epsilon(k)$-*uniform* in S if it is ϵ-close to the uniform ensemble over S (we sometimes omit S when it is clear from the context). $X = \{X_k\}_k$, $Y = \{Y_k\}_k$ are *computationally indistinguishable* if every $\mathrm{poly}(k)$-time adversary $\mathcal{A}$ has negligible distinguishing advantage:

$$|\Pr[\mathcal{A}(X_k) = 1] - \Pr[\mathcal{A}(Y_k) = 1]| = \mathrm{negl}(k)\ .$$

4.1 Linear Algebra

We make extensive use of linear algebra over finite fields of the form $\mathbb{Z}_q$ for a prime q. Some of our results apply to any finite field.

For a linear subspace of column vectors $S \subseteq \mathbb{Z}_q^n$, we let $S^k \subseteq \mathbb{Z}_q^{n\times k}$ denote the set of matrices whose every column is an element in S. If $S \subseteq \mathbb{Z}_q^n$ is a row subspace then $S^k \subseteq \mathbb{Z}_q^{k\times n}$ is defined symmetrically.

The *dimension* of a linear subspace $S \subseteq \mathbb{Z}_q^n$ is the maximal number of linearly independent vectors in S. The *basis* of a linear subspace $S \subseteq \mathbb{Z}_q^n$ with dimension $d \leq n$ is represented by a matrix $\mathbf{B} \in \mathbb{Z}_q^{n\times d}$ whose columns are the basis vectors, i.e. S is the span (see below) of the columns of $\mathbf{B}$. We sometimes consider linear subspaces of row vectors, in such case, the basis will be a matrix $\mathbf{B} \in \mathbb{Z}_q^{d\times n}$ whose rows are the basis vectors.

The *rank* of a matrix is the maximal number of linearly independent rows (or columns) in the matrix. The set of $(n\times k)$-matrices having rank-d is denoted by $\mathrm{Rk}_d(\mathbb{Z}_p^{n\times k})$. Given a set of matrices $M \subseteq \mathbb{Z}_p^{n\times k}$, we use $\mathrm{Rk}_d(M)$ to denote the set of all rank-d matrices in M. The dimension of the row span and of the column span (see below) of a matrix are equal to its rank.

We use the term *span* of a matrix to indicate its row span, i.e. for $\mathbf{A} \in \mathbb{Z}_q^{n\times k}$, we denote $\mathrm{Span}(\mathbf{A}) = \{\mathbf{z}^T \cdot \mathbf{A} : \mathbf{z} \in \mathbb{Z}_q^n\}$. The *kernel* of a matrix is the linear space that is orthogonal to its span, this corresponds to defining $\ker(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}_q^k : \mathbf{A} \cdot \mathbf{x} = \mathbf{0}\}$.

We will also use the following simple fact (see e.g. [BK10]), we provide the proof for completeness.

Lemma 4.1. *Consider a finite field $\mathbb{F}$ of order q. For any $n, m \in \mathbb{N}$ such that $m \geq n$, the distance between the distributions $U(\mathbb{F}^{n\times m})$ and $U(\mathrm{Rk}_n(\mathbb{F}^{n\times m}))$ is at most $1/(q^{m-n} \cdot (q - 1))$.*

Proof. Consider sampling the matrix row by row. The probability that row i is a linear combination of previous rows is at most $q^{-m} \cdot q^{i-1}$. Applying the union bound gives the result. $\square$

4.2 Hardness Assumptions in Bilinear Groups

Consider multiplicative groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, all of prime order p, and let g_1, g_2 be generators for $\mathbb{G}_1, \mathbb{G}_2$ respectively. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ has the following properties. *Bilinearity:* for all $x \in \mathbb{G}_1, y \in \mathbb{G}_2$, $a, b \in \mathbb{Z}$ it holds that $e(x^a, y^b) = e(x, y)^{ab}$; *Non-degeneracy:* $e(g_1, g_2) \neq 1$.

The symmetric external Diffie-Hellman (SXDH**) assumption.** For $i = 1, 2$, let $\mathbb{G}_i = \{\mathbb{G}_{i,k}\}_{k \in \mathbb{N}}$ be a family of groups, where each group $\mathbb{G}_{i,k}$ is of order p, and where p is a k-bit prime. Fix (arbitrary) generators $g_i = g_{i,k}$ for $\mathbb{G}_{i,k}$ (we omit the subscript k to avoid cluttering of notation).

The SXDH assumption is that the DDH problem is hard in $\mathbb{G}_1$ and in $\mathbb{G}_2$. Let us explicitly present a linear algebraic form of this assumption which will allow for a better intuitive understanding of our constructions. Let $\mathbf{x}, \mathbf{x'} \overset{\$}{\leftarrow} \mathbb{Z}_p^n$ be uniformly distributed vectors of any (polynomial) dimension n and let $\alpha \overset{\$}{\leftarrow} \mathbb{Z}_p$ be a uniform scalar. Then the SXDH assumption is equivalent to the assumption that for $i = 1, 2$, the distributions $(g_i^{\mathbf{x}}, g_i^{\alpha \cdot \mathbf{x}})$ and $(g_i^{\mathbf{x}}, g_i^{\mathbf{x'}})$ are computationally indistinguishable.

The linear assumption. This assumption was first introduced in [BBS04], we use a matrix form that was introduced in [NS09] and proven ([NS09, Lemma A.1]) to be implied by the standard form. Throughout this work, the term "linear assumption" refers to the matrix form described below. While this assumption can be stated for the general case of groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ as above, we only present the simpler case where $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$. Our results extend to the general case as well.

Let $\mathbb{G} = \{\mathbb{G}_k\}_{k \in \mathbb{N}}$ be a family of groups as above with a fixed generator g. Consider the distribution D_r ($r \in \{2, 3\}$) defined by the following sampling process: sample $\mathbf{A} \overset{\$}{\leftarrow} \mathrm{Rk}_d(\mathbb{Z}_p^{3 \times 3})$ and output $g^{\mathbf{A}}$. The *linear assumption* is that D_2 and D_3 are computationally indistinguishable. In other words, it is hard to distinguish a random 3×3 rank-2 matrix from a random such rank-3 matrix, if given in the exponent of a group generator.

An immediate corollary of the above definition (see e.g. [NS09, Appendix A]) is that under the linear assumption, for all polynomially-bounded $m, \ell \geq 3$, $r \geq 2$, $t \geq 0$, a random $(m \times \ell)$ rank-r matrix is computationally indistinguishable from a random such rank-$(r + t)$ matrix, when given in the exponent.

4.3 Simulation Sound NIZK Proofs

We define the notion of simulation-sound NIZK, a notion that was introduces by Sahai [Sah99]. Intuitively, a simulation-sound NIZK proof system is a NIZK proof system with the extra property that a polynomially bounded cheating prover is incapable of convincing the verifier of a false statement, even after seeing any (polynomial) number of simulated proofs of her choosing.

Throughout this paper, when we refer to a NIZK proof system we always mean *adaptive* NIZK, as defined below.

Definition 4.1. *[FLS90, BFM88, BSMP91]: $\Pi = (\ell, P, V, S = (S_1, S_2))$ is an* efficient adaptive NIZK proof system *for a language $L \in \mathbf{NP}$ with witness relation $\mathcal{R}$ if ℓ is a polynomial, P, V, S_1, S_2 are all* PPT *algorithms, and there exists a negligible function μ such that for all n the following three requirements hold.*

- **Completeness**: *For all $x \in L$ of length n, and all w such that $\mathcal{R}(x,w) = 1$, and for all strings $r \in \{0,1\}^{\ell(n)}$,*
$$V(x, P(x,w,r), r) = 1.$$
- **Adaptive Soundness**: *For all adversaries $\mathcal{A}$, if $r \in_R \{0,1\}^{\ell(n)}$ is chosen uniformly at random, then the probability that $\mathcal{A}(r)$ will output a pair (x,π) such that $x \notin L$ and yet $V(x,\pi,r) = 1$, is at most $\mu(n)$.*
- **Adaptive Zero-Knowledge**: *For all* PPT *adversaries $\mathcal{A}$,*
$$\left|\Pr[\mathrm{Expt}_{\mathcal{A}}(n) = 1] - \Pr[\mathrm{Expt}^{S}_{\mathcal{A}}(n) = 1]\right| \leq \mu(n),$$
where the experiment $\mathrm{Expt}_{\mathcal{A}}(n)$ is defined by:
$$r \leftarrow \{0,1\}^{\ell(n)}$$
$$\textit{Return } \mathcal{A}^{P(\cdot,\cdot,r)}(r)$$
and the experiment $\mathrm{Expt}^{S}_{\mathcal{A}}(n)$ is defined by:
$$(r,\tau) \leftarrow S_1(1^n)$$
$$\textit{Return } \mathcal{A}^{S'(\cdot,\cdot,r,\tau)}(r),$$
where $S'(x,w,r,\tau) = S_2(x,r,\tau)$.

We next define the notion of simulation-sound NIZK.

Definition 4.2 ([Sah99]). *Let $\Pi = (\ell, P, V, S = (S_1, S_2))$ be an efficient adaptive NIZK proof system for a language $L \in$* **NP**. *We say that Π is* simulation sound *if for all* PPT *adversaries $\mathcal{A}$ we have that*
$$\Pr[\mathrm{Expt}_{\mathcal{A},\Pi}(n) = 1] = \mathrm{negl}(n),$$
where $\mathrm{Expt}_{\mathcal{A},\Pi}(n)$ is the following experiment:
$$(r,\tau) \leftarrow S_1(1^n)$$
$$(x,\pi) \leftarrow \mathcal{A}^{S_2(\cdot,r,\tau)}(r)$$
Let Q be the list of proofs given by S_2 above
Return 1 iff ($\pi \notin Q$ and $x \notin L$ and $V(x,\pi,r) = 1$)

As was shown by Sahai [Sah99], every language in **NP** has a simulation-sound NIZK proof assuming the existence of enhanced trapdoor permutations.

4.4 Short Non-Interactive Arguments

Definition 4.3. *A* **non-interactive argument system** *for an NP language L is defined by a polynomial p and a pair of* PPT *Turing machines (P,V) satisfying the following properties:*

- **Completeness.** *If $(x,w) \in R_L$ then for all $\mathsf{crs} \in \{0,1\}^{p(k)}$ we have $V_{\mathsf{crs}}(x, P_{\mathsf{crs}}(x,w)) = 1$.*
- **Computational (adaptive) soundness.** *For every (non-uniform) polynomial-time P', the following is negligible:*
$$\Pr\left[\mathsf{crs} \leftarrow \{0,1\}^{p(k)}; (x,\pi) \leftarrow P'(\mathsf{crs}) : V_{\mathsf{crs}}(x,\pi) = 1 \bigwedge x \notin L\right].$$

The argument system has **proofs of length** ℓ *if for all* k, *all* $\mathsf{crs} \in \{0,1\}^{p(k)}$, *and all* $(x, w) \in R_L$ *it holds that* $|P_{\mathsf{crs}}(x, w)| \leq \ell(k)$.

The results of [Kil92, Mic00, BG08] imply that for any fixed NP language L, there exists a non-interactive argument system for L with proofs of length $\omega(\log^2 k)$ *in the random oracle model* (and, in fact, with the crs being the emptystring). Nevertheless, use of the random oracle here does not appear to be inherent and one could hope to base the existence of short non-interactive arguments on cryptographic/complexity-theoretic assumptions

4.5 Lossy Trapdoor Functions

The notion of lossy trapdoor functions (LTDF) was first defined by Peikert and Waters [PW08]. Let $n(k) = \mathrm{poly}(k)$ represent the input length of the function and $\ell(k) \leq n(k)$ represent the lossiness of the collection. For convenience, we also define the residual leakage $r(k) := n(k) - \ell(k)$. For all these quantities, we often omit the dependence on k.

In this work, we require a family of lossy trapdoor functions with the additional special property of *oblivious sampling*. Essentially this means that the description of a random function from the family (either the lossy or injective) is computationally indistinguishable from the uniform distribution. It turns out that known constructions indeed have this property.

Definition 4.4 (Lossy trapdoor functions with oblivious sampling)**.** *A collection of* (n, ℓ)*-lossy trapdoor functions is given by a tuple of* PPT *algorithms* $(S_{inj}, S_{loss}, G, G^{-1})$ *having the properties below.*

- Easy to sample an injective function with trapdoor: S_{inj} *takes as input a security parameter* 1^k *and outputs a pair* (s, t), *where* s *is a function index and* t *is the associated trapdoor, such that* $G_s(\cdot)$ *is an injective function over the domain* $\{0,1\}^n$, *with inverse* $G_t^{-1}(\cdot)$.
- Easy to sample a lossy function: S_{loss} *takes as input a security parameter* 1^k *and outputs* $(s, \perp)$, *where* s *is a function index and* $G_s(\cdot)$ *is a function* $g_s(\cdot)$ *over the domain* $\{0,1\}^n$ *whose image has size at most* $2^r = 2^{n-\ell}$.
- Hard to distinguish injective from lossy: *The first outputs of* S_{inj} *and* S_{loss} *are computationally indistinguishable. More formally, let* X_k *denote the distribution of* s *from* S_{inj}, *and let* Y_k *denote the distribution of* s *from* S_{loss}. *Then* $\{X_k\}_{k \in \mathbb{N}} \overset{c}{\approx} \{Y_k\}_{k \in \mathbb{N}}$.

We require an additional property:

- Oblivious sampling: *The first outputs of* S_{inj} *and* S_{loss} *are computationally indistinguishable from the* uniform distribution.

We remark that the families of lossy trapdoor functions presented in [PW08], that are based on the DDH and LWE assumptions have this property, under the same assumptions. In addition, the d-linear based construction of [FGK$^+$09] also has this property.

5 Random Subspaces are Leakage Resilient

In this section, we provide a linear algebraic tool that is crucial to our leakage resilient constructions, but is rather general and may also be usable elsewhere.

Informally, we consider a random subspace $\mathbf{X}$ of dimension at least 2 of a given linear space. A leakage on $\mathbf{X}$ is given in the following form: an arbitrary leakage function (that must not depend on $\mathbf{X}$) is applied to a random vector $\mathbf{v}$ in $\mathbf{X}$. We show a bound linking the size of the range of the leakage function (i.e. the amount of information it reveals about $\mathbf{v}$) to the statistical distance between this experiment and the case where $\mathbf{v}$ is sampled uniformly at random in the entire linear space (independently of $\mathbf{X}$). In the latter case, the leakage function reveals nothing on the subspace $\mathbf{X}$, and therefore we conclude that applying a length-restricted leakage function to a random vector in the subspace is leakage resilient, in the sense it does not reveal information about the subspace that it originated from.

We also consider the case where the leakage function is applied to *two* random vectors in the subspace. In this case, it is required that the subspace $\mathbf{X}$ is of dimension at least 4. More generally, if the leakage function is applied to d random vectors in the subspace, we need the random subspace to be of dimension $2d$. This requirement stems from our analysis, which uses the fact that random dimension-d subspaces of $\mathbf{X}$ are (almost) pairwise independent, which indeed holds in the case where the dimension of $\mathbf{X}$ is at least twice that of the sampled subspaces.

We formally state the two theorem below. The first is for the case that the leakage is applied to *one* random vector, and the second is for the case that the leakage is applied to *two* random vectors.

Theorem 5.1. *Let $m, \ell \in \mathbb{N}$, $m \geq \ell \geq 2$ and let q be a prime. Let $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$, let $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^{\ell}$ and let $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^{m}$. Let $f : \mathbb{Z}_q^m \to W$ be some function. Then,*

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{r})),\ (\mathbf{X}, f(\mathbf{u}))\big) \leq \epsilon,$$

as long as

$$|W| \leq q^{\ell-1}\cdot\epsilon^2.$$

Theorem 5.2. *Let $m, \ell \in \mathbb{N}$, $m \geq \ell \geq 4$ and let q be a prime. Let $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$, let $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})$ and let $\mathbf{Y} \xleftarrow{\$} \mathbb{Z}_q^{m\times 2}$. Let $f : \mathbb{Z}_q^{m\times 2} \to W$ be some function. Then,*

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{Y}))\big) \leq \epsilon,$$

as long as

$$|W| \leq q^{\ell-3}\cdot\epsilon^2.$$

We note that our original proofs result in parameters that are slightly worse than the ones given in Theorems 5.1 and 5.2. The proofs which achieve the improved parameters (as stated in Theorems 5.1 and 5.2) were pointed to us independently by Gil Segev, Daniel Wichs and Yevgeniy Dodis, and follow the proof of the "generalized crooked leftover hash lemma" [DS05, BFO08]. In Section 5.1 we give an outline of our original proofs. The formal proofs (using our original analysis, and thus yielding slightly worse parameters) are given in Section 5.2. A proof with tight parameters appears in Appendix A.

5.1 Proof Overview

The proofs of Theorems 5.1 and 5.2 are essentially the same (both our original proofs which yield worse parameters, and the improved ones). For simplicity, in what follows, we focus on Theorem 5.1.

The two distributions considered in this theorem can be described by the following experiment. First $\mathbf{X}$ is uniformly sampled, then a vector $\mathbf{v} \in \mathbb{Z}_q^m$ is generated by either sampling a random vector $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_q^{\ell}$ and setting $\mathbf{v} = \mathbf{X} \cdot \mathbf{r}$ (in the first distribution) or sampling a random $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$ and setting $\mathbf{v} = \mathbf{u}$ (in the second distribution). Then, the function f is applied to $\mathbf{v}$ to obtain $w = f(\mathbf{v})$. Finally, $(\mathbf{X}, w)$ is output.

We need to prove that even though the distributions of $(\mathbf{X}, \mathbf{v})$ in the two experiments are very distinct in terms of statistical distance, it still holds that the distributions of $(\mathbf{X}, w)$ are very close. To prove this, let us fix the value of $w \in W$ and consider the conditional distribution of $\mathbf{X}$ (since the marginal distribution of w is the same in both experiments, this will enable us to later average on w and get the full result). In the second experiment, $\mathbf{X}$ is independent of $\mathbf{v}, w$ and thus the conditional distribution of $\mathbf{X}$ is uniform. We need to prove, therefore, that the conditional distribution of $\mathbf{X}$ in the first experiment is close to uniform as well. From now on we focus on the first experiment, i.e. $\mathbf{v} = \mathbf{X} \cdot \mathbf{r}$.

Conditioned on w, the probability for any (fixed) subspace $\mathbf{X}$ is proportional to the probability that, for a random $\mathbf{r}$, it holds that $f(\mathbf{X} \cdot \mathbf{r}) = w$. We show that for almost all $\mathbf{X}$, this probability is very close to its expected value (which is, of course, the probability that $f(\mathbf{v}) = w$ for a random $\mathbf{v}$).

To see this, fix any $\mathbf{r}_1, \mathbf{r}_2$. Note that if $\mathbf{r}_1, \mathbf{r}_2$ are linearly independent, then $\mathbf{X} \cdot \mathbf{r}_1$ and $\mathbf{X} \cdot \mathbf{r}_1$ are independent (where the probabilities are now taken only over $\mathbf{X}$). Moreover, note that if $\mathbf{r}_1, \mathbf{r}_2$ were chosen uniformly at random, then they would indeed be linearly independent with very high probability. Thus, the probability that $f(\mathbf{X} \cdot \mathbf{r}) = w$ can be written as a sum (or expectancy) of almost pairwise independent random variables. Applying Chebyshev's inequality implies that such variables are concentrated around their expected value.

We get, therefore, a bound on the distance between the two distributions conditioned on w. As explained above, averaging over all values w and appropriately selecting the parameters implies the theorem.

In what follows, we give the formal proofs which yields slightly weaker parameters than these guaranteed by Theorems 5.1 and 5.2. In the formal treatment we focus on Theorem 5.2 (as opposed to Theorem 5.1). The reason is that Theorem 5.2 is used to obtain our results based on the linear assumption, and Theorem 5.1 is used to obtain our results based on the symmetric external Diffie-Hellman (SXDH) assumption. Since the former assumption appears to be more standard, we give all the formal proofs needed for these results. The proofs needed for our results based on the SXDH assumption are essentially the same.

5.2 Formal Proof

In what follows we give the formal proof of a weaker version of Theorem 5.2, stated below.

Theorem 5.3. *Let $m, \ell \in \mathbb{N}$, $m \geq \ell \geq 4$ and let q be a prime. Let $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$, let $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})$ and let $\mathbf{Y} \xleftarrow{\$} \mathbb{Z}_q^{m\times 2}$. Let $f : \mathbb{Z}_q^{m\times 2} \to W$ be some function. Then it holds that*

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{Y}))\big) \leq \epsilon$$

as long as

$$|W| \leq \frac{q^{\frac{\ell-3}{2}} \cdot \epsilon^{3/2}}{\sqrt{2}}.$$

Relating the formal proof of this theorem to the overview above, Lemma 5.4 gives the Chebyshev-based concentration bound discussed above (where the set S in the lemma statement is the set of all $\mathbf{Z} \in \mathbb{Z}_q^{m\times 2}$ such that $f(\mathbf{Z}) = w$). Lemma 5.5 uses this concentration bound to bound the distance of the conditional distribution of $\mathbf{X}$ and the uniform distribution. The remainder of the proof, namely the weighted average on the values of w is provided in the actual body of the proof of the theorem.

As discussed in the overview above, we provide two required lemmas first and then derive the proof of the theorem. The first lemma shows that the "intersection" of a random subspace $\mathbf{X}$ with any set is well concentrated around its expected value.

Lemma 5.4. *Let $m, \ell \in \mathbb{N}$, $m \geq \ell \geq 4$ and let q be a prime. Consider $S \subseteq \mathbb{Z}_q^{m\times 2}$ and let $\rho = \Pr_{\mathbf{Y} \xleftarrow{\$} \mathbb{Z}_q^{m\times 2}}[\mathbf{Y} \in S]$. Define, for all $\mathbf{X} \in \mathbb{Z}_q^{m\times \ell}$,*

$$d_S(\mathbf{X}) = \Pr_{\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})}[\mathbf{X}\cdot\mathbf{T} \in S] \ .$$

Then for all $\delta > 0$

$$\Pr_{\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m\times \ell}}\left[\left|d_S(\mathbf{X}) - \rho\right| \geq \delta\right] \leq \frac{2\rho}{\delta^2 \cdot q^{\ell-3}} \ .$$

Proof. Let $\mathbf{T}_1, \mathbf{T}_2 \in \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})$ such that $\mathrm{Rk}([\mathbf{T}_1 \| \mathbf{T}_2]) = 4$. Then $\mathbf{X}\cdot\mathbf{T}_1$ and $\mathbf{X}\cdot\mathbf{T}_2$ are independent random variables, when $\mathbf{X}$ is a random matrix. This holds since if we define $\mathbf{R} = [\mathbf{T}_1 \| \mathbf{T}_2]$, then $\mathbf{X}\cdot\mathbf{R}$ is exactly the first 4 columns of $\mathbf{X}$ under some basis-change. Since $\mathbf{X}$ is uniform, it is also uniform under any basis change, and specifically the first 4 columns are random and therefore independent.

We note that by definition

$$d_S(\mathbf{X}) = \Pr_{\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})}[\mathbf{X}\cdot\mathbf{T} \in S] = \mathbb{E}_{\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}\in S}] \ .$$

Therefore, it holds that $\mathbb{E}[d_S(\mathbf{X})] = \mathbb{E}_{\mathbf{X},\mathbf{T}}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}\in S}] = \rho$. Furthermore, letting F denote the event that $[\mathbf{T}_1 \| \mathbf{T}_2]$ is full rank (and $\bar{F}$ denote its complement),

$$\begin{aligned}
\mathbb{V}[d_S(\mathbf{X})] &= \mathbb{E}_{\mathbf{X}}[(d_S(\mathbf{X}) - \rho)^2] \\
&= \mathbb{E}_{\mathbf{X}}\Big[\big(\mathbb{E}_{\mathbf{T}_1}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S} - \rho]\big)\cdot\big(\mathbb{E}_{\mathbf{T}_2}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_2\in S} - \rho]\big)\Big] \\
&= \mathbb{E}_{\mathbf{X}}\Big[\mathbb{E}_{\mathbf{T}_1,\mathbf{T}_2}\Big[\big(\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S} - \rho\big)\big(\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_2\in S} - \rho\big)\Big]\Big] \\
&= \mathbb{E}_{\mathbf{T}_1,\mathbf{T}_2}\Big[\mathbb{E}_{\mathbf{X}}\Big[\big(\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S} - \rho\big)\big(\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_2\in S} - \rho\big)\Big]\Big] \\
&= \mathbb{E}_{\mathbf{T}_1,\mathbf{T}_2}[\mathrm{Cov}_{\mathbf{X}}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S}, \mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_2\in S}]] \\
&= \Pr[\bar{F}]\cdot\mathbb{E}_{(\mathbf{T}_1,\mathbf{T}_2)|\bar{F}}[\mathrm{Cov}_{\mathbf{X}}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S}, \mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_2\in S}]] + \Pr[F]\cdot\underbrace{\mathbb{E}_{(\mathbf{T}_1,\mathbf{T}_2)|F}[\mathrm{Cov}_{\mathbf{X}}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S}, \mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_2\in S}]]}_{=0 \text{ (since } \mathbf{X}\cdot\mathbf{T}_1 \text{ and } \mathbf{X}\cdot\mathbf{T}_2 \text{ are independent)}} \\
&= \Pr[\bar{F}]\cdot\mathbb{E}_{(\mathbf{T}_1,\mathbf{T}_2)|\bar{F}}[\mathrm{Cov}_{\mathbf{X}}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S}, \mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_2\in S}]] \\
&\leq \Pr[\bar{F}]\cdot\mathbb{E}_{(\mathbf{T}_1,\mathbf{T}_2)|\bar{F}}[\mathbb{V}_{\mathbf{X}}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S}]] \\
&= \Pr[\bar{F}]\cdot\mathbb{E}_{\mathbf{T}_1}[\mathbb{V}_{\mathbf{X}}[\mathbb{1}_{\mathbf{X}\cdot\mathbf{T}_1\in S}]] \\
&= \Pr[\bar{F}]\cdot(\rho - \rho^2) \\
&\leq \Pr[\bar{F}]\cdot\rho \ .
\end{aligned}$$

To bound $\Pr[\bar{F}] = \Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}] = \Pr_{\mathbf{T}_1,\mathbf{T}_2}[\mathrm{Rk}([\mathbf{T}_1 \| \mathbf{T}_2]) < 4]$, fix $\mathbf{T}_1 \in \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})$ and consider $\mathbf{T}_2 \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell\times 2})$. Let $\mathbf{t}_1 \in \mathbb{Z}_q^\ell$ denote the first column of $\mathbf{T}_2$ and let $\mathbf{t}_2 \in \mathbb{Z}_q^\ell$ denote the second column of $\mathbf{T}_2$. Both $\mathbf{t}_1$ and $\mathbf{t}_2$ are uniform in $\mathbb{Z}_q^m \setminus \{\mathbf{0}\}$. By the union bound,

$$\Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}] \leq \Pr_{\mathbf{t}_1 \xleftarrow{\$} \mathbb{Z}_q^\ell\setminus\{\mathbf{0}\}}[\mathbf{t}_1 \in \mathrm{Span}(\mathbf{T}_1)] + \Pr_{\mathbf{t}_2 \xleftarrow{\$} \mathbb{Z}_q^\ell\setminus\{\mathbf{0}\}}[\mathbf{t}_2 \in \mathrm{Span}(\mathbf{T}_1, \mathbf{t}_1)] \leq \frac{q^2}{q^\ell} + \frac{q^3}{q^\ell} \leq \frac{2q^3}{q^\ell}.$$

We can therefore apply Chebyshev's inequality and conclude that

$$\Pr_{\mathbf{X}}[|d_S(\mathbf{X}) - \rho| \geq \delta] \leq \frac{2\rho}{\delta^2 \cdot q^{\ell-3}} \ . \qquad \square$$

The second lemma uses the concentration obtained above to conclude that the conditional distribution of $\mathbf{X}$, given that $f(\mathbf{X} \cdot \mathbf{T}) = w$, is close to uniform.

Lemma 5.5. *Let $m, \ell \in \mathbb{N}$, $m \geq \ell \geq 4$ and let q be a prime. Consider $S \subseteq \mathbb{Z}_q^{m \times 2}$ and let $\rho = \Pr_{\mathbf{Y} \xleftarrow{\$} \mathbb{Z}_q^{m \times 2}}[\mathbf{Y} \in S]$. Consider the distributions $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m \times \ell}$, $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell \times 2})$, conditioned on the event $\mathbf{X} \cdot \mathbf{T} \in S$. Then*

$$\mathsf{dist}(\mathbf{X}, U(\mathbb{Z}_q^{m \times \ell})) \leq \left(\frac{2}{\rho^2 \cdot q^{\ell-3}} \right)^{1/3} \ .$$

Proof. For convenience, let us denote $\alpha = \left| \mathbb{Z}_q^{m \times \ell} \right|^{-1} = q^{-m\ell}$. Consider the conditional distribution of $\mathbf{X}$, it holds that for all $\mathbf{A} \in \mathbb{Z}_q^{m \times \ell}$

$$\Pr[\mathbf{X} = \mathbf{A} | \mathbf{X} \cdot \mathbf{T} \in S] = \frac{\Pr[(\mathbf{X} = \mathbf{A}) \wedge (\mathbf{X} \cdot \mathbf{T} \in S)]}{\Pr[\mathbf{X} \cdot \mathbf{T} \in S]} = \frac{\alpha \cdot \Pr_{\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_q^{\ell \times 2})}[\mathbf{A} \cdot \mathbf{T} \in S]}{\rho} = \frac{\alpha \cdot d_S(\mathbf{A})}{\rho} \ .$$

Where d_S is as defined in Lemma 5.4.

The statistical distance from uniform is therefore

$$\mathsf{dist}(\mathbf{X}, U(\mathbb{Z}_q^{m \times \ell})) = \frac{1}{2} \cdot \sum_{\mathbf{A} \in \mathbb{Z}_q^{m \times \ell}} \left| \frac{\alpha \cdot d_S(\mathbf{A})}{\rho} - \alpha \right| = \frac{1}{2\rho} \cdot \mathbb{E}_{\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times \ell}} \left[|d_S(\mathbf{A}) - \rho| \right] \ .$$

Using Lemma 5.4, together with Bayes' theorem, we get that for all $\delta > 0$,

$$\mathsf{dist}(\mathbf{X}, U(\mathbb{Z}_q^{m \times \ell})) \leq \frac{1}{2\rho} \cdot \left(\delta + \frac{2\rho}{\delta^2 \cdot q^{\ell-3}} \right) \ .$$

Setting $\delta = \left(\frac{2\rho}{q^{\ell-3}} \right)^{1/3}$, the result follows. $\square$

We can now prove the theorem by averaging on the values of w.

Proof of Theorem 5.3. For all $w \in W$, denote $S_w = \{\mathbf{Y} : f(\mathbf{Y}) = w\}$, $\rho_w = \Pr_{\mathbf{Y}}[\mathbf{Y} \in S_w]$. Using Lemma 5.5, the statistical distance, conditioned on the second element being w, is at most $\left(\frac{2}{\rho_w^2 \cdot q^{\ell-3}} \right)^{1/3}$. Averaging over all values $w \in W$, we get

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X} \cdot \mathbf{T})),\ (\mathbf{X}, f(\mathbf{Y}))\big) \leq \sum_{w \in W} \rho_w \cdot \left(\frac{2}{\rho_w^2 \cdot q^{\ell-3}} \right)^{1/3} =$$

$$= \left(\frac{2}{q^{\ell-3}} \right)^{1/3} \cdot \sum_{w \in W} \rho_w^{1/3} \leq \left(\frac{2}{q^{\ell-3}} \right)^{1/3} \cdot |W|^{2/3} = \left(\frac{\sqrt{2} \cdot |W|}{q^{\frac{\ell-3}{2}}} \right)^{2/3} \leq \epsilon \ ,$$

assuming

$$|W| \leq \frac{q^{\frac{\ell-3}{2}} \cdot \epsilon^{3/2}}{\sqrt{2}} \ . \qquad \square$$

6 An Encryption Scheme Secure Against Continual Leakage

In this section, we present the first public-key encryption scheme $\mathcal{L}$ that is secure against continual leakage attacks. Our scheme is based on the linear assumption in bilinear groups. It is parameterized by a polynomially bounded integer $\ell \geq 7$ that allows for a tradeoff between the sizes of keys and ciphertexts and the maximal tolerable leakage rate: the public and secret keys contain 2ℓ group elements and a ciphertext contains ℓ group elements.

We show that our scheme is resilient to leakage of constant rate from the memory and leakage of sub-constant rate from the key-update procedure and from the key generation procedure. Recalling Definition 3.1, this corresponds to a case where $\rho_M = \Omega(1)$ and $\rho_G = \rho_U = o(1)$.

More specifically, the value of ρ_M for a general ℓ can be as high as $\frac{\ell-6-\gamma}{2\ell}$, for all $\gamma > 0$. For example, taking $\ell = 12$ guarantees resiliency to leakage rate of $1/4 - \gamma$, while the keys and ciphertexts only contain a constant number of group elements. Taking ℓ to be asymptotically increasing, on the other hand, makes the keys and ciphertexts sizes asymptotically increase but makes the tolerable leakage rate $1/2 - o(1)$.

The value of ρ_G, ρ_U for general ℓ is $\omega\left(\frac{\log k}{\ell \log p}\right)$, where k is the security parameter and p is the order of the group relative to which we work. We cannot provide explicit expressions for ρ_G, ρ_U, but rather we show that for all $c > 0$, the scheme is resilient in the case where $\rho_G, \rho_U = \frac{c \cdot \log k}{\ell \log p}$. In fact, our argument can be generalized in a straightforward manner to imply resiliency of $\Omega\left(\frac{\log T(k)}{\ell \log p}\right)$ if one is willing to assume that the linear assumption is hard for adversaries that run in time (roughly) $T(k)$. We note that while tolerating such leakage in non-continual memory attacks is trivial, this is not the case in the continual model.

We remind the reader that a simpler scheme with better parameters can be achieved based on the less standard SXDH assumption, as described in Section 2.2.

Our scheme is described in Section 6.1. We provide a high-level overview of the security proof in Section 6.2 and then present the formal proof in Section 6.3.

6.1 The Scheme $\mathcal{L}[\ell]$

Let us overview the main ideas behind the structure of our scheme. We want to use the fact that under the linear assumption, random rank-2 matrices in the exponent are indistinguishable from random rank-3 matrices.

We therefore set the public-key to $g^{\mathbf{A}}$, where $\mathbf{A}$ a random $2 \times \ell$ matrix. The ciphertexts are of the form $g^{\mathbf{v}^T}$, where $\mathbf{v}$ is a vector of length ℓ. Namely, the public-key and the ciphertext together form a $(3 \times \ell)$ matrix in the exponent. If this matrix is random rank-2 for encryptions of 0 and random rank-3 for encryptions of 1, then we could use the indistinguishability argument to achieve (semantic) security. We do this by encrypting 0 by setting $g^{\mathbf{v}^T}$ to be a linear combination of the rows of $g^{\mathbf{A}}$; and encrypting 1 by setting $g^{\mathbf{v}^T}$ to be a random vector. One can see that the resulting distributions are statistically close to the prescribed ones. Thus, security is achieved.

The next question, of course, is figuring out how to decrypt such ciphertexts. We notice a non-zero vector $\mathbf{y}$ in the kernel of $\mathbf{A}$ can be used to distinguish the two cases: we can compute $g^{\mathbf{v}^T \cdot \mathbf{y}}$ and see if the result equals to g^0. This will always be the case for encryptions of 0 and will only happen with negligible probability for encryptions of 1. This suggests that such $\mathbf{y}$ can be used as a secret-key, but it is not clear how such a secret-key can be safely refreshed.

In order to be able to refresh, we use a secret key of the form $g^{\mathbf{Y}}$, where $\mathbf{Y} = [\mathbf{y}_1 \| \mathbf{y}_2]$ and $\mathbf{y}_1, \mathbf{y}_2$ are random vectors in the kernel of $\mathbf{A}$. The fact that the vectors are given in the exponent means that we cannot compute the product $g^{\mathbf{v}^T \cdot \mathbf{y}_i}$ as suggested before. We thus work over groups with bilinear maps, which enable computing $e(g,g)^{\mathbf{v}^T\cdot\mathbf{Y}}$ instead.

The key refresh operation is done by "rotating" the matrix $\mathbf{Y}$: we sample a 2×2 full rank matrix $\mathbf{R}$, and set the new secret key to $g^{\mathbf{Y}\cdot\mathbf{R}}$. Intuitively, since everything is done in the exponent, the update operation is indistinguishable from sampling a fresh random secret-key, which turns out to be a useful property.

The scheme $\mathcal{L}[\ell]$ is formally presented in Figure 3. Correctness (with all but negligible probability) follows almost immediately by the explanations above.

Encryption scheme $\mathcal{L}[\ell]$

Parameters. The scheme is parameterized by groups $\mathbb{G}, \mathbb{G}_T$ of prime order p, such that there exists a bilinear map $e : \mathbb{G}\times\mathbb{G}\to\mathbb{G}_T$. Let g be a generator of $\mathbb{G}$ (and so $e(g,g)$ is a generator for $\mathbb{G}_T$). An additional parameter $\ell\geq 7$ is polynomial in the security parameter. Setting different values for ℓ will enable us a tradeoff between efficiency and the rate of tolerable leakage.

Key-generation. The key-generator runs as follows. It samples $\mathbf{A}\xleftarrow{\$}\mathbb{Z}_p^{2\times\ell}$ and $\mathbf{Y}\xleftarrow{\$}\ker^2(\mathbf{A})$ (i.e. $\mathbf{Y}\in\mathbb{Z}_p^{\ell\times 2}$) (note that this is easy to sample since $\mathbf{A}$ is known). It sets $pk = g^{\mathbf{A}}$ and $sk = g^{\mathbf{Y}}$.

Key-update. To update the secret key $sk = g^{\mathbf{Y}}\in\mathbb{G}^{\ell\times 2}$, sample $\mathbf{R}\xleftarrow{\$}\mathrm{Rk}_2(\mathbb{Z}_p^{2\times 2})$ and then set $sk' = g^{\mathbf{Y}\cdot\mathbf{R}}$.

Encryption. Given a public-key $pk = g^{\mathbf{A}}\in\mathbb{G}^{2\times\ell}$, encrypting the bit 0 is done by sampling $\mathbf{r}\xleftarrow{\$}\mathbb{Z}_p^2$ and setting the output ciphertext to $c = g^{\mathbf{r}^T\cdot\mathbf{A}}$. Encrypting the bit 1 is done by setting the ciphertext to $c = g^{\mathbf{u}^T}$ where $\mathbf{u}\xleftarrow{\$}\mathbb{Z}_p^{\ell}$.

Decryption. Given a ciphertext $c = g^{\mathbf{v}^T}$ and a secret-key $sk = g^{\mathbf{Y}}$, the decryption process computes $e(g,g)^{\mathbf{v}^T\cdot\mathbf{Y}}$. If the result is equal to $e(g,g)^{\mathbf{0}}$, then output 0 and otherwise output 1.

Figure 3: Encryption scheme in the CML model, based on the linear assumption.

6.2 Overview of the Security Proof

As explained above, our scheme is resilient to leakage from memory, from the key-update procedure and from the key generation procedure. We will leave handling with leakage from key-generation to the end. Thus assume from this point that $\rho_G = 0$ until we mention otherwise.

Let us start by considering a possible proof for standard CPA security of $\mathcal{L}[\ell]$ (as already sketched in Section 6.1 above). The public-key $g^{\mathbf{A}}$ contains a random $2\times\ell$ matrix (in the exponent), and the challenge ciphertext is either a linear combination of the rows of $\mathbf{A}$ or a random vector of length ℓ. The joint distribution of the public-key and ciphertext (we can consider this to be a matrix $\mathbf{V}\in\mathbb{Z}_p^{3\times\ell}$ whose first 2 rows are the matrix $\mathbf{A}$ and its last row is the challenge $\mathbf{v}^T$) is statistically close to either a random rank-2 matrix or a random rank-3 matrix (in the exponent), respectively. An adversary that distinguishes these distributions, therefore, immediately breaks the linear assumption.

Trying to make the argument above leakage resilient, we will need to somehow simulate the leakage from the secret-key. This seems self defeating, as having a secret-key for the scheme should mean that the CPA adversary becomes useless (as we can use the secret-key and decrypt the ciphertext ourselves). We show that the above discouraging intuition is not accurate. To this end, we rely on the fact that our scheme does not have perfect correctness, but rather has a negligible probability of being incorrect.

Consider, at this point, the case where no update queries are made, i.e. the same secret-key is used throughout the attack. At a very high level, what we will do is show how to generate keys and a challenge ciphertext in such a way that the secret-key, while being appropriately distributed respective to the public-key, is useless against our challenge ciphertext. Namely, the distributions (pk, sk) and (pk, c) are proper, but (pk, sk, c) is far from being proper, in the sense that the secret key sk is useless in decrypting the specific ciphertext c (c will always decrypt to 0 using sk).[6] Hence, a CPA adversary that decrypts c correctly can actually be useful in breaking the linear assumption.

We then show that if the amount of leakage is sufficiently bounded, then an adversary cannot gain sufficient information about the secret key sk to discriminate the ciphertext c from a random ciphertext, and thus should indeed decrypt c correctly (with non-negligible probability). We explain this in detail below and then explain how to extend this to the continual leakage scenario where key-updates occur (and thus multiple secret-keys are used and are leaked during the attack).

Let us explain how to generate keys and a challenge ciphertext, such that the keys are properly distributed but are still useless against the challenge ciphertext, so that a successful CPA adversary can be used to break the linear assumption. Consider an instance of the linear assumption, this is a matrix (in the exponent) $g^{\mathbf{C}}$, where $\mathbf{C} \in \mathbb{Z}_p^{3\times 3}$ is either rank-2 or rank-3. We will use a successful CPA adversary $\mathcal{A}$ to construct a PPT algorithm $\mathcal{B}$ that distinguishes between the case that $\mathbf{C}$ is of rank 2 and the case that $\mathbf{C}$ is of rank 3.

The algorithm $\mathcal{B}$, on input $g^{\mathbf{C}}$ does the following. It samples $\ell - 3$ random vectors $\{\mathbf{x}_i\} \xleftarrow{\$} \mathbb{Z}_p^{\ell}$ (with all but negligible probability, these vectors are linearly independent), and generates a $3 \times \ell$ matrix (in the exponent) $g^{\mathbf{V}}$ that has the following properties: (1) If $\mathbf{C}$ is a random rank-2 matrix, then $\mathbf{V}$ is a random rank-2 matrix; (2) If $\mathbf{C}$ is a random rank-3 matrix, then $\mathbf{V}$ is a random rank-3 matrix; (3) The vectors $\{\mathbf{x}_i\}$ are uniformly distributed in the kernel of $\mathbf{V}$.

This can be done using linear algebra, as follows. Let $\mathbf{X}$ denote the $\ell \times (\ell - 3)$ matrix whose i^{th} column is $\mathbf{x}_i$. Let $\mathbf{B}$ be a basis for the linear subspace $\{\mathbf{w}^T \in \mathbb{Z}_p^{\ell} : \mathbf{w}^T \cdot \mathbf{X} = \mathbf{0}\}$. Note that $\mathbf{B}$ is efficiently computable given $\mathbf{X}$. Setting $g^{\mathbf{V}} = g^{\mathbf{C}\cdot\mathbf{B}}$, it holds that $\mathbf{V}$ has the same rank as $\mathbf{C}$. By symmetry, the vectors $\{\mathbf{x}_i\}$ are uniformly distributed in $\ker(\mathbf{V})$ as required.

The algorithm $\mathcal{B}$ feeds the adversary $\mathcal{A}$ with the first two rows of $g^{\mathbf{V}}$ as the public-key, denoting it by $g^{\mathbf{A}}$. It generates a secret-key $g^{\mathbf{y}_1}, g^{\mathbf{y}_2}$ by choosing at random $\mathbf{y}_1, \mathbf{y}_2 \xleftarrow{\$} \{\mathbf{X} \cdot \mathbf{t} : \mathbf{t} \in \mathbb{Z}_p^{\ell-3}\}$, i.e. randomly sampling from the column span of $\mathbf{X}$. Then, it feeds $\mathcal{A}$ with the challenge ciphertext $g^{\mathbf{v}^T}$, which is the last row of $g^{\mathbf{V}}$. Notice that our secret-key will always decrypt the challenge ciphertext $g^{\mathbf{v}^T}$ to 0, even if $\mathbf{v}^T$ is independent of the rows of $\mathbf{A}$ and hence should be decrypted to 1 (thus, our secret key is "crippled" w.r.t. the ciphertext $g^{\mathbf{v}^T}$). A CPA adversary that succeeds in decrypting c correctly (in spite of getting leakage from a "crippled" secret-key) will, therefore, break the linear assumption by determining if $\mathbf{V}$ (and hence $\mathbf{C}$) is rank-2 or rank-3.

Next, we explain why the adversary $\mathcal{A}$, which is given leakage from our "crippled" secret key, should nevertheless decrypt c correctly (with non-negligible probability). That is, we explain why it cannot distinguish the "crippled" secret key distribution from the genuine one (given that the leakage is small enough). To this end, we use the algebraic tool that we develop in Section 5, to show that a small enough leakage *statistically* hides the subspace $\mathbf{X}$ that the secret-key is sampled from, making it statistically hard to distinguish our secret-key distribution and the legal distribution where $\mathbf{y}_1, \mathbf{y}_2 \xleftarrow{\$} \ker(\mathbf{A})$. Thus, the adversary cannot use the leakage to distinguish between our "crippled" secret-key and a genuine one. This completes the proof for the non-continual problem.

We now address the fact that the leakage is continual, namely that there is a sequence of phases in which leakage occurs, each followed by a refresh operation. At this point, however, we still

[6]It is here that we use the fact that our scheme does not have perfect completeness.

assume that the key update procedure does not leak (i.e. $\rho_U = 0$). While our "legal" key-update operation never changes the column span of the secret-key,[7] we notice that it is computationally indistinguishable from one that re-samples new vectors $\mathbf{y}_1, \mathbf{y}_2$ from the column span of $\mathbf{X}$ at every update operation.[8] Indistinguishability holds as one can consider the legal key distribution as setting $sk' = g^{\mathbf{X}' \cdot \mathbf{T}}$ where $\mathbf{X}'$ is random rank-2, distinguishing $g^{\mathbf{X}}$ and $g^{\mathbf{X}'}$ is hard by the linear assumption, and this holds even in the presence of the matrix $\mathbf{A}$. (Jumping ahead, it is this indistinguishability argument that is troublesome when leakage from the key-update procedure is allowed.) Note that this "improper" update can be simulated, since we explicitly know $\mathbf{X}$. This enables us to apply the above argument consecutively: at each phase, we will leak from new vectors in the column span of $\mathbf{X}$, which will be statistically close to leaking from $\mathbf{y}_1, \mathbf{y}_2 \xleftarrow{\$} \ker(\mathbf{A})$, which is, in turn, indistinguishable from the "legal" key distribution.

Lastly, we are left with treating leakage from the update process (i.e. $\rho_U > 0$). Any amount of such leakage seems to completely break the indistinguishability argument of the previous paragraph. Intuitively, the adversary is getting leakage from the input to the update procedure (including the random tape used), and thus may have some information on what the legal output should be. Then, once the adversary sees some memory leakage from an improper secret-key, used for the security reduction, it can compare it to what it knows about the legal secret-key, thus catching any attempt for a switch of distributions as above. In order to overcome this barrier, we must come up with a way to simulate the leakage from the update, in such a way that will ease the mind of the adversary $\mathcal{A}$ and make it behave properly in spite of the discrepancy between the distributions.

The key observation is that whether or not $\mathcal{A}$'s mind is at ease is an efficiently checkable event. We can simulate the remainder of the security game, using the proposed improper secret key and some candidate leakage value (as if everything from this point and on is done legitimately) and see if $\mathcal{A}$'s success probability decreases or not. If the number of bits being leaked is small enough, specifically $O(\log k)$, we can efficiently go over all possible values until we find the one that works.

This looks good and well, but what if *no* leakage value works? In this case, we recall again that the improper key distribution used in our reduction is indistinguishable from the original one. Therefore, since in the original distribution there is always a good leakage value — the legal value,[9] this should also be the case with the improper one. This holds since the event of not finding an acceptable leakage value is also efficiently checkable (again, by simulating $\mathcal{A}$). A change in behavior in this respect yields a distinguisher between the real and improper distributions, which is impossible under our hardness assumption.

We get, therefore, that a leakage of $O(\log k)$ bits from the update procedure can indeed be tolerated. This can be generalized in a straightforward manner to imply a tolerable leakage of $O(\log T(k))$ bits, if we assume that the linear assumption is hard for (roughly) $T(k)$-time adversaries.

The last thing to do is handle leakage from the key-generation. At this point, however, it is clear how this can be done (for $O(\log k)$ bits of leakage). We use the same technique we use for updates: after generating our initial secret-key, we go over all possible values of key-generation leakage, and find one for which $\mathcal{A}$ works well.

[7]Recall that the update procedure takes $sk = g^{\mathbf{Y}}$ and outputs $sk' = g^{\mathbf{Y} \cdot \mathbf{R}}$ for an invertible $\mathbf{R}$.

[8]The new key distribution sets $sk' = g^{\mathbf{X} \cdot \mathbf{T}}$, for a properly sampled matrix $\mathbf{T}$, regardless of the previous sk.

[9]In fact, this is only true with noticeable probability over the sampling of the secret key, and therefore we may need to try a few keys before we find a good one.

6.3 Proof of Security of Encryption Scheme $\mathcal{L}[\ell]$

Theorem 6.1. *Under the linear assumption, for every $\ell \geq 7$, the encryption scheme $\mathcal{L}[\ell]$ (described in Figure 3) has the following security guarantee: For all constants $\gamma, c > 0$, $\mathcal{L}[\ell]$ is secure in the* CML *model with leakage rate*

$$(\rho_G, \rho_U, \rho_M) = \left(\frac{c \cdot \log k}{4\ell \cdot \log p}, \frac{c \cdot \log k}{(2\ell+4) \cdot \log p}, \frac{\ell - 6 - \gamma}{2\ell}\right) .$$

Proof. Suppose, towards contradiction, that there exist constants $c, \gamma > 0$ and a PPT adversary $\mathcal{A}$ such that $\mathcal{A}$ succeeds in breaking security in the CML model with leakage rate

$$(\rho_G, \rho_U, \rho_M) = \left(\frac{c \cdot \log k}{4\ell \cdot \log p}, \frac{c \cdot \log k}{(2\ell+4) \cdot \log p}, \frac{\ell - 6 - \gamma}{2\ell}\right) .$$

That is, $\mathcal{A}$ succeeds in the security game with probability $\frac{1}{2} + \epsilon$, for $\epsilon \geq 1/\text{poly}(k)$. Let t be some (polynomial) upper bound on $\mathcal{A}$'s running time. This, specifically, is also an upper bound on the number of update queries made by $\mathcal{A}$.

We construct a PPT algorithm $\mathcal{B}$ that runs in time $\text{poly}(k, t, 1/\epsilon, k^c) = \text{poly}(k)$ and breaks the linear assumption with probability $\frac{1}{2} + \frac{\epsilon^2}{32} - \text{negl}(k)$.

First, let us explicitly bound the absolute number of bits that can leak at every update operation (as opposed to the relative bound ρ_U). The total bit-length of the secret-key and randomness used in the update process is $2\ell \log p + 4 \log p$. Thus the absolute number of bits that can be leaked is at most $\rho_U \cdot (2\ell+4) \log p = c \cdot \log k$. This means that the image of the leakage function has cardinality at most $k^c = \text{poly}(k)$. Similarly, the total number of bits that can leak during the key generation process is also bounded by $c \log k$.

The algorithm $\mathcal{B}$ takes as input a matrix (in the exponent) $g^{\mathbf{C}}$, where $\mathbf{C} \in \mathbb{Z}_p^{3\times 3}$, and where the distribution of $\mathbf{C}$ is the following: a bit $b \xleftarrow{\$} \{0,1\}$ is sampled uniformly; if $b = 0$ then $\mathbf{C} \xleftarrow{\$} \text{Rk}_2(\mathbb{Z}_p^{3\times 3})$ (i.e., $\mathbf{C}$ is a random rank 2 matrix); if $b = 1$ then $\mathbf{C} \xleftarrow{\$} \text{Rk}_3(\mathbb{Z}_p^{3\times 3})$ (i.e., $\mathbf{C}$ is a random rank 3 matrix). $\mathcal{B}$ will output a guess b' as to the value of the bit b, such that

$$\Pr[b' = b] \geq \frac{1}{2} + \frac{\epsilon^2}{32} - \text{negl}(k) \geq \frac{1}{2} + 1/\text{poly}(k) ,$$

and thus break the assumption.

We actually assume that $\mathbf{C}$ is slightly differently distributed: The first two rows of $\mathbf{C}$ always form a random rank-2 matrix, and the last row is either a random linear combination of the first two rows, in the case that $b = 0$, or is a truly random vector, in the case that $b = 1$. We can assume this w.l.o.g. since this distribution is statistically indistinguishable from the original one.

The algorithm $\mathcal{B}$, on input $g^{\mathbf{C}}$, makes a guess b' as to the value of the bit b, as follows:

1. Generate a matrix $\mathbf{X}$, and matrices $g^{\mathbf{V}}, g^{\mathbf{A}}$, where $\mathbf{X} \in \mathbb{Z}_p^{\ell\times(\ell-3)}$, $\mathbf{V} \in \mathbb{Z}_p^{3\times\ell}$, $\mathbf{A} \in \mathbb{Z}_p^{2\times\ell}$, such that
    (a) $\mathbf{X}$ is uniformly distributed in $\text{Rk}_{\ell-3}(\mathbb{Z}_p^{\ell\times(\ell-3)})$.
    (b) $\mathbf{A}$ is a random full-rank matrix such that $\mathbf{A} \cdot \mathbf{X} = \mathbf{0}$.
    (c) The first two rows of $\mathbf{V}$ are identical to $\mathbf{A}$, and the last row, denoted by $\mathbf{v}^T$, is distributed as follows: If $\mathbf{C}$ is a rank-2 matrix then $\mathbf{v}^T$ is a random linear combination of the rows of $\mathbf{A}$; and if $\mathbf{C}$ is a rank-3 matrix then $\mathbf{v}^T$ is a random vector such that $\mathbf{v}^T \cdot \mathbf{X} = 0$.

This is done by sampling $\mathbf{X} \xleftarrow{\$} \mathrm{Rk}_{\ell-3}(\mathbb{Z}_p^{\ell\times(\ell-3)})$, letting $g^{\mathbf{V}} = g^{\mathbf{C}\cdot\mathbf{B}}$, where $\mathbf{B} \in \mathbb{Z}_p^{3\times\ell}$ is a matrix whose rows form a basis of the subspace $\{\mathbf{w}^T : \mathbf{w}^T \cdot \mathbf{X} = 0\}$, and letting $g^{\mathbf{A}}$ be the first two rows of $g^{\mathbf{V}}$.

2. Let $pk = g^{\mathbf{A}}$, and compute a corresponding secret key by choosing a random full-rank matrix $\mathbf{T}_0 \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{(\ell-3)\times 2})$ and setting $sk_0 \doteq g^{\mathbf{Y}_0} := g^{\mathbf{X}\cdot\mathbf{T}_0}$. The distribution of the key pair (pk, sk_0) is statistically close to that of a "legal" key pair, generated by the original key generation algorithm.

3. Test the success probability of $\mathcal{A}$ on the key pair (sk_0, pk) with all possible values of leakage from the key generation, as follows. Define $M \doteq \frac{k\cdot t^2}{\epsilon^2}$. For all $\alpha \in \{0,1\}^{c\log k}$, emulate the security game with $\mathcal{A}$, conditioned on (sk_0, pk) and on α's being the answer to the key generation leakage, for M times. For each $\kappa \in [M]$, let Z_κ be an indicator variable for the event that $\mathcal{A}$ succeeded in the κ^{th} emulation. Set $\bar{Z}_{0,\alpha} := \frac{1}{M}\sum_{i\in[M]} Z_\kappa$.

   If $\bar{Z}_{0,\alpha} \geq \frac{1}{2} + \frac{3\epsilon}{4} - \frac{\epsilon}{2(t+1)}$ then set $\eta_0 := \bar{Z}_{0,\alpha}$ and continue to the next step. If no α has this property, then $\mathcal{B}$ aborts and returns a random bit $b' \xleftarrow{\$} \{0,1\}$.

4. Feed the adversary $\mathcal{A}$ with the public key $g^{\mathbf{A}}$.

5. In what follows we explain how $\mathcal{B}$ answers $\mathcal{A}$'s queries. To this end, $\mathcal{B}$ maintains a value for the current secret-key sk_i, which he uses to answer leakage queries. Jumping ahead, $\mathcal{B}$ emulates update queries differently than the legal Update procedure, which results in the distribution of the sk_i's being different from the distribution of the secret-keys in the original game. As a result, the emulation of the leakage from the update procedure will require special care to prevent $\mathcal{A}$ from noticing the change.

   - The leakage queries of $\mathcal{A}$ are answered by computing the leakage function on the current secret-key sk_i. Namely, if in time period i the adversary $\mathcal{A}$ sends a leakage query (leak, f_i), then $\mathcal{B}$ returns the answer $f_i(sk_i)$.
   - To answer update queries, $\mathcal{B}$ does the following. Consider the distribution $D_{\mathbf{X}}$ defined by sampling a random full-rank matrix $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{(\ell-3)\times 2})$, and outputting $g^{\mathbf{X}\cdot\mathbf{T}}$ (we remark that sk_0 was sampled in the same way).

     Upon receiving the i^{th} update query (update, f_i), $\mathcal{B}$ updates the secret key as follows. It samples $g^{\mathbf{Y}_i} \xleftarrow{\$} D_{\mathbf{X}}$.[10] Then, to compute a corresponding leakage, $\mathcal{B}$ goes over all the possible leakage values (note that there are only k^c possibilities). For each possible leakage value $\alpha \in \{0,1\}^{c\cdot\log k}$, it tests if it is a "good" leakage w.r.t. $sk_i = g^{\mathbf{Y}_i}$. A good leakage is one that (almost) preserves the success probability of $\mathcal{A}$ in spite of using the wrong key-distribution. If no such good leakage is found, we try sampling $g^{\mathbf{Y}_i} \xleftarrow{\$} D_{\mathbf{X}}$ anew. We will have to show below that with high probability, a good $g^{\mathbf{Y}_i}$ is found after a sufficiently small number of trials.

     In what follows, we show how $\mathcal{B}$ estimates the success probability of $\mathcal{A}$, conditioned on $(g^{\mathbf{Y}_i}, \alpha)$, in order to determine whether α is a good leakage w.r.t. $g^{\mathbf{Y}_i}$. This is done similarly to the way $\mathcal{B}$ tested the success probability of $\mathcal{A}$ (conditioned on the key pair (sk_0, pk)) in Step 3. Recall our definition of $M = \frac{k\cdot t^2}{\epsilon^2}$.

[10]We only refer to the distribution $D_{\mathbf{X}}$ explicitly in the proof of Claim 6.2 below. Until then, one can just consider setting $g^{\mathbf{Y}_i} := g^{\mathbf{X}\cdot\mathbf{T}_i}$ for $\mathbf{T}_i \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{(\ell-3)\times 2})$.

(a) For each $\kappa \in [M]$, emulate a (fresh) continuation of $\mathcal{A}$ while simulating the rest of its queries legally.
That is, upon receiving an update query (update, f_j) in time period $j > i$, sample $\mathbf{R}_j \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{2\times 2})$, set $sk_j{:=}\mathsf{Update}(sk_{j-1}; \mathbf{R}_j)$, and leak the prescribed value $f_j(sk_{j-1}, \mathbf{R}_j)$. Upon receiving a leakage query (leak, f_j) in time period j, feed $\mathcal{A}$ with the answer $f_j(sk_j)$. Finally, when $\mathcal{A}$ asks for the challenge ciphertext, choose a random bit $\beta \xleftarrow{\$} \{0,1\}$, and give $\mathcal{A}$ an encryption $c \leftarrow \mathsf{Enc}(g^{\mathbf{A}}, \beta)$.
If $\mathcal{A}$'s output is equal to the bit β, then set $Z_\kappa = 1$; otherwise set $Z_\kappa = 0$.

(b) Compute $\bar{Z}_{(i,\alpha,g^{\mathbf{Y}_i})}{:=}\frac{1}{M}\sum_{\kappa=1}^{M} Z_\kappa$.

(c) If $\bar{Z}_{(i,\alpha,g^{\mathbf{Y}_i})} \geq \eta_{i-1} - \frac{\epsilon}{2(t+1)}$ then we say that α is "good". Otherwise we say that it is "bad".

After a good leakage value α has been found w.r.t. some secret key $g^{\mathbf{Y}_i}$, $\mathcal{B}$ will set $sk_i{:=}g^{\mathbf{Y}_i}$, answer the update query of $\mathcal{A}$ with the leakage value α, and will continue the simulation of $\mathcal{A}$.

To ensure that a good leakage value is found with high probability, $\mathcal{B}$ repeats this trial several times, as follows. Recall the value of η_0 set above. Similarly to η_0, the value of η_i (in item (b) below) is an estimation of the success probability of $\mathcal{A}$, conditioned on its state after the first i updates.

(a) Sample $g^{\mathbf{Y}_i} \xleftarrow{\$} D_{\mathbf{X}}$

(b) Go over all $\alpha \in \{0,1\}^{c\log k}$ in some arbitrary order (say lexicographic), and test whether α is "good" with respect to the secret key $g^{\mathbf{Y}_i}$. If a "good" α is found then the response to the i^{th} leakage query is set to α and the new secret key is set to $sk_i{:=}g^{\mathbf{Y}_i}$. In addition, set $\eta_i{:=}\bar{Z}_{(i,\alpha,g^{\mathbf{Y}_i})}$.

(c) Otherwise, if all the possible α's are "bad", then go back to step (a). If more than $J \doteq \frac{tk}{\epsilon}$ values of $g^{\mathbf{Y}_i}$ have been tried, then abort and output a random guess $b' \xleftarrow{\$} \{0,1\}$.

6. When $\mathcal{A}$ asks for the challenge ciphertext, $\mathcal{B}$ sends $g^{\mathbf{v}^T}$ as the challenge ciphertext.

7. Finally, when $\mathcal{A}$ outputs a guess b', then $\mathcal{B}$ outputs the same value b'.

We argue that $\mathcal{B}$ guesses the bit b (i.e., $b' = b$) with probability $\frac{1}{2} + \frac{\epsilon^2}{32} - \mathrm{negl}(k)$.

This will follow in a straightforward manner by combining the following 3 claims (the event Λ will be defined below).

Claim 6.2. *Under the linear assumption,*

$$\Pr[\mathcal{B} \textit{ doesn't abort } |\Lambda] \geq \epsilon/4 - \mathrm{negl}(k) .$$

Claim 6.3. *It holds that*

$$\Pr[b' = b|\mathcal{B} \textit{ doesn't abort } ,\Lambda] \geq \frac{1}{2} + \epsilon/8 - \mathrm{negl}(k) .$$

Claim 6.4. *It holds that*

$$\Pr[\Lambda] = 1 - \mathrm{negl}(k) .$$

We start by defining the event Λ and proving Claim 6.4. For all $i \in \{0\} \cup [t]$, $\alpha \in \{0,1\}^{c\log k}$, and for every possible value $g^{\mathbf{Y}_i}$ sampled during the running of $\mathcal{B}$, we define $P_{(i,\alpha,g^{\mathbf{Y}_i})}$, as follows.

Fix the coin tosses of $\mathcal{B}$ until it receives the i'th update query from $\mathcal{A}$. Conditioned on this fixing, $P_{(i,\alpha,g^{\mathbf{Y}_i})}$ is the probability of $\mathcal{A}$'s success in an emulated game in which the first i updates are emulated by $\mathcal{B}$ with the i^{th} secret key set to $g^{\mathbf{Y}_i}$ and leakage value for the i^{th} update set to α, and the remainder of the game is emulated as perscribed (i.e. key updates are performed using the procedure Update and leakage values are computed by definition). We note that $P_{(i,\alpha,g^{\mathbf{Y}_i})}$ is not a well defined value in its own right, but rather a function of the coin tosses of $\mathcal{B}$ until (and including) the i^{th} update. The value $P_{0,\alpha}$ is defined to be the success probability of $\mathcal{A}$ in a game, fixing (sk_0, pk) and taking α as the value of the key-generation leakage. We also denote by P_i the value of $P_{(i,\alpha,g^{\mathbf{Y}_i})}$ for the $(\alpha, g^{\mathbf{Y}_i})$ that are finally selected by $\mathcal{B}$ to be used as leakage value and sk_i respectively (P_0 is defined analogously to be $P_{0,\alpha}$ for the α that is actually used).

Throughout $\mathcal{B}$'s emulation of the security game with $\mathcal{A}$, it tries to estimate the values of P_i by sampling M continuations of $\mathcal{A}$ and checking the success rate. The event Λ indicates that all such estimations are sufficiently close to the correct value. Formally, the event Λ holds if and only if both for all $\alpha \in \{0,1\}^{c\log k}$,

$$\left|\bar{Z}_{0,\alpha} - P_{0,\alpha}\right| \le \frac{\epsilon}{8t}$$

and for every $i \in \{0\} \cup [t]$, every $\alpha \in \{0,1\}^{c\log k}$, and all the $\{g^{\mathbf{Y}_i}\}$ chosen by $\mathcal{B}$,

$$\left|\bar{Z}_{(i,\alpha,g^{\mathbf{Y}_i})} - P_{(i,\alpha,g^{\mathbf{Y}_i})}\right| \le \frac{\epsilon}{8t} .$$

Note that as a special case, Λ implies that for all $i \in \{0\} \cup [t]$ it holds that $|\eta_i - P_i| \le \frac{\epsilon}{8t}$.

Claim 6.4, therefore, is a straightforward application of the Chernoff bound. The formal argument follows.

Proof of Claim 6.4. By Chernoff,[11] for all $i, \alpha, g^{\mathbf{Y}_i}$,

$$\Pr\left[\left|\bar{Z}_{(i,\alpha,g^{\mathbf{Y}_i})} - P_{(i,\alpha,g^{\mathbf{Y}_i})}\right| > \frac{\epsilon}{8t}\right] \le 2e^{-\epsilon^2 M/(32t^2)} = 2e^{-k/32} ,$$

and the same also holds for $\bar{Z}_{0,\alpha}$ and $P_{0,\alpha}$. Since the total number of such $(i, \alpha, g^{\mathbf{Y}_i})$ (and $\bar{Z}_{0,\alpha}$) is at most $t \cdot k^c \cdot J + k^c = \text{poly}(k)$, the claim follows by the union bound. ∎

We move on to proving Claim 6.3.

Proof of Claim 6.3. Recall that this claim considers a probability space where $\mathcal{B}$ does not abort and the event Λ occurs (i.e. all η_i's are close to the respective P_i's). In such case, by definition

$$\eta_t \ge \eta_{t-1} - \frac{\epsilon}{2(t+1)} \ge \eta_{t-2} - \frac{2\epsilon}{2(t+1)} \ge \cdots \ge \eta_0 - \frac{t\epsilon}{2(t+1)} \ge \left(\frac{1}{2} + \frac{3\epsilon}{4} - \frac{\epsilon}{2(t+1)}\right) - \frac{t\epsilon}{2(t+1)} = \frac{1}{2} + \frac{\epsilon}{4} .$$

Since Λ holds, it implies that $P_t \ge \eta_t - \frac{\epsilon}{8t} \ge \frac{1}{2} + \frac{\epsilon}{4} - \frac{\epsilon}{8t} \ge \frac{1}{2} + \frac{\epsilon}{8}$.

However, we are not done because P_t refers to the success probability of $\mathcal{A}$ given a legal challenge ciphertext, while $\mathcal{B}$ feeds it with $g^{\mathbf{v}^T}$ which is malformed. For example, in the case where $b = 1$, a legal challenge ciphertext is uniformly distributed, while the ciphertext $g^{\mathbf{v}^T}$ always satisfies $\mathbf{v}^T \cdot \mathbf{Y}_i = \mathbf{0}$, for any secret key of the form $g^{\mathbf{Y}_i} = g^{\mathbf{X}\cdot\mathbf{T_i}}$.

It remains to show, therefore, that using such malformed challenge can only change the success probability of $\mathcal{A}$ by a negligible amount. If $b = 0$ (i.e. the matrix $\mathbf{C}$ is of rank 2) then this

[11]Recall that the (additive) Chernoff bound implies that if $X_1, X_2, \ldots, X_M$ are indicator random variables such that $\mathbb{E}(X_i) = \mu$. Then, $\Pr[|\bar{X} - \mu| > \epsilon] \le 2e^{-2\epsilon^2 M}$.

immediately follows: in such case, the distribution of $\mathbf{v}^T$ is identical to that of a valid challenge ciphertext. Proving this for $b = 1$ (where $\mathbf{C}$ has rank 3) is highly non-trivial, and is proved using the algebraic tool developed in Section 5, as follows.

Consider the linear space $\ker(\mathbf{A})$. Notice that $\mathbf{X}$ and $g^{\mathbf{A}}$ are sampled in such a way that the columns of $\mathbf{X}$ are within negligible statistical distance from a set of $\ell - 3$ random vectors in $\ker(\mathbf{A})$. We apply Theorem 5.3 inside the linear space $\ker(\mathbf{A})$, one can think of the representation of $\mathbf{X}$ and all other elements relative to some basis of $\ker(\mathbf{A})$. The amount of total allowed leakage from each $g^{\mathbf{Y}_i}$ consists of the leakage α from the update procedure, and of the leakage $f_i(g^{\mathbf{Y}_i})$ from memory. This amounts to

$$\rho_M \cdot |sk| + \rho_U(|sk| + |r|) = \rho_M \cdot 2\ell \cdot \log p + c \cdot \log k \ .$$

An important delicate point is that we also need to consider the "leakage" from the $g^{\mathbf{Y}_i}$ that ended up not being used, because the very fact that they are not used affected the view of the adversary (since another value was used instead). One can consider these values as leaking a single bit saying that they are "not good" (which is clearly less than what leaks from the ones that are actually used, so we will just use the bound above). Therefore, we need to consider leakage from $J \cdot t + 1$ values of the form $g^{\mathbf{Y}_i} = g^{\mathbf{X} \cdot \mathbf{T}_i}$ (recall that sk_0 also leaks).

Let $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_p^\ell$ and let $\mathbf{Z}_i \xleftarrow{\$} \ker^2(\mathbf{A})$ for all i. Applying Theorem 5.3 with parameters $\hat{m} = \ell - 2$ (this is the dimension of $\ker(\mathbf{A})$), $\hat{\ell} = \ell - 3$ (this is the number of random vectors in the subspace that are contained in $\mathbf{X}$), and with functions h_i,[12] whose range is of cardinality $|W| = p^{2\ell \cdot \rho_M} \cdot k^c$, we get

$$\left(\mathbf{A}, \mathbf{v}, h_0(g^{\mathbf{X}\cdot\mathbf{T}_0}), h_1(g^{\mathbf{X}\cdot\mathbf{T}_1}), \ldots\right) \overset{\delta}{\equiv} \tag{1}$$
$$\left(\mathbf{A}, \mathbf{v}, h_0(g^{\mathbf{Z}_0}), h_1(g^{\mathbf{Z}_1}), \ldots\right) \equiv \tag{2}$$
$$\left(\mathbf{A}, \mathbf{u}, h_0(g^{\mathbf{Z}_0}), h_1(g^{\mathbf{Z}_1}), \ldots\right) \overset{\delta}{\equiv} \tag{3}$$
$$\left(\mathbf{A}, \mathbf{u}, h_0(g^{\mathbf{X}\cdot\mathbf{T}_0}), h_1(g^{\mathbf{X}\cdot\mathbf{T}_1}), \ldots\right) \ . \tag{4}$$

The transitions from (1) to (2) and from (3) to (4) follow immediately from Theorem 5.3 (in fact, these also hold if $\mathbf{X}$ is a part of the distribution). The transition from (2) to (3) follows from the fact that $b = 1$, and thus the distribution of $\mathbf{v}$, even conditioned on $\mathbf{A}$, is uniform in $\mathbb{Z}_p^\ell$.

The value of δ is derived from Theorem 5.2

$$\delta \doteq J \cdot t \cdot \sqrt{\frac{|W|}{p^{\ell-6}}} + \text{negl}(k) = J \cdot t \cdot 2^{1/3} \cdot \sqrt{\frac{p^{2\ell\cdot\rho_M} \cdot k^c}{p^{\ell-6}}} + \text{negl}(k) \ ,$$

where the additional $\text{negl}(k)$ comes from $\mathbf{X}$'s being only statistically close to uniform. Assigning our value of $\rho_M = \frac{\ell-6-\gamma}{2\ell}$, we get $\delta = Jt \cdot k^{c/2} \cdot p^{-\gamma/2} + \text{negl}(k) = \text{negl}(k)$.

This concludes the proof of the claim. ■

Finally, we prove Claim 6.2.

Proof of Claim 6.2. We need to prove that $\mathcal{B}$ does not abort with sufficient probability. To see this, we use the fact that an abort is an efficiently recognizeable event. Our strategy, therefore, is to replace $\mathcal{B}$ with an appropriate $\mathcal{B}'$ that produces a computationally indistinguishable distribution and bounding the abortion probability of $\mathcal{B}'$. The claim on $\mathcal{B}$ will then immediately follow.

[12]Think of h_i as containing the total amount of leakage on sk_i, including memory leakage and update leakage. Namely, $h_i(g^{\mathbf{X}\cdot\mathbf{T}_i}) \doteq (\alpha_i, f_i(g^{\mathbf{X}\cdot\mathbf{T}_i}))$.

Specifically, we define $\mathcal{B}'$ as identical to $\mathcal{B}$ with the exception that the probability $D_{\mathbf{X}}$, that is used in step 5, is replaced by a distribution D' that is defined by the following sampling process: sample $\mathbf{R} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{2\times 2})$ and output $g^{\mathbf{Y}_0\cdot\mathbf{R}}$ (recall that $sk_0 = g^{\mathbf{Y}_0}$). This means that in $\mathcal{B}'$, the updated secret-keys are sampled from the correct distribution, but the leakage value is still computed using the "trial and error" mechanism. It should be clear (as will be formally proven below) that if the keys are sampled legally, then a good leakage value has to exist (with sufficient probability).

The indistinguishability of $\mathcal{B}$ and $\mathcal{B}'$ follows from the indistinguishability of the two distributions $D_{\mathbf{X}}$ and D'. This indistinguishability holds even given $\mathbf{V}$, by the following argument. Let $\mathbf{V}$ be a known matrix. Let $\mathbf{B} \in \mathbb{Z}_p^{\ell\times(\ell-3)}$ be the first $\ell-3$ columns of a random basis for $\ker(\mathbf{V})$ (such basis contains either $\ell-2$ or $\ell-3$ vectors), and let $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{(\ell-3)\times 2})$. Let $g^{\mathbf{M}}$ be such that $\mathbf{M} \in \mathbb{Z}_p^{(\ell-3)\times(\ell-3)}$ and is either random of rank 2 or random of rank $\ell-3$. The linear assumption implies that distinguishing the two cases given only $g^{\mathbf{M}}$ is hard. Note that if $\mathbf{M}$ is of rank 2 then $g^{\mathbf{B}\cdot\mathbf{M}\cdot\mathbf{T}}$ is distributed according to D'. On the other hand, if $\mathbf{M}$ is of rank $\ell-3$ then $g^{\mathbf{B}\cdot\mathbf{M}\cdot\mathbf{T}}$ is distributed according to $D_{\mathbf{X}}$. Indistinguishability follows and thus,

$$\Pr[\mathcal{B}\text{ doesn't abort } |\Lambda] \geq \Pr[\mathcal{B}'\text{ doesn't abort } |\Lambda] - \mathrm{negl}(k)\ .$$

We move on to bound $\Pr[\mathcal{B}'\text{ doesn't abort } |\Lambda]$. From now on, all events are conditioned on Λ. We first notice that a standard Markov argument implies that

$$\Pr\left[P_0 \geq \frac{1}{2} + \frac{3\epsilon}{4}\right] \geq \epsilon/4.$$

It follows that the test in step 3 passes with at least that probability (up to a negligible distance), since (sk_0, pk) are statistically indistinguishable from the proper distribution and therefore, since Λ holds, when trying the "correct" key-generation leakage value, the test will pass.

Let us now recursively bound the probability of abortion in step i conditioned on no abortion in previous steps. No abortion in previous steps implies that η_{i-1} is well defined and $P_{i-1} \geq \eta_{i-1} - \frac{\epsilon}{8t}$. A Markov argument implies that with probability $\frac{\epsilon}{8t}$ over sk_i it holds that $P_i \geq \eta_{i-1} - \frac{\epsilon}{4t}$ when the "real" leakage value is used. Such sk_i will surely pass the test (recall that event Λ guarantees that all Chernoff estimates are close to their respective values). Trying $J = \frac{tk}{\epsilon}$ times guarantees that such $g^{\mathbf{Y}_i}$ is sampled with all but negligible probability.

It follows that

$$\Pr[\mathcal{B}'\text{ doesn't abort } |\Lambda] \geq \frac{\epsilon}{4} - \mathrm{negl}(k)\ ,$$

and the claim follows. ■

This completes the proof of the theorem. □

7 A Continual-Leakage Secure Identity-Based Encryption Scheme

In this section we present a definition for identity based encryption in the CML model. We then show how to extend the scheme presented in Section 6 to obtain an identity based encryption scheme.

First, in Section 7.1, we discuss what the correct definition of IBE in the CML model needs to be, leading, in Section 7.2, to the formal security definition to be used in this work. Our construction is presented at a high level in Section 7.3 and is formally stated in Figure 4. We provide a sketch for the proof of security in Section 7.4 (we do not give a formal proof as it is a straightforward combination of our encryption scheme, from Section 6, and the linear-based IBE of [BK10]).

Interestingly, the simplified scheme $\mathcal{L}^*$ (see Section 2.2) does not seem to extend to an IBE scheme the way our main scheme $\mathcal{L}$ does.

7.1 Modeling IBE in the CML Model

In identity based encryption, a trusted authority holds a master secret-key (msk) and posts a set of public parameters (pp). The public parameters enable anyone to encrypt messages, knowing only the identity (id) of the intended recipient. The trusted authority can use its master secret key to produce a respective secret-key (sk_{id}) for the user whose identity is id. This secret-key enables decrypting messages intended for user id.

When applying the CML model to this framework, we notice a significant difference from the cases of encryption or signatures discussed above. While in the aforementioned cases, only one entity was holding secret information and thus susceptible to leakage (the decryptor and the signer, respectively), in the IBE case, we have a "hierarchy" of secrets: The trusted authority holds msk, naturally breaching its security could imply loss of security for all users; the individual users have their own secret-keys sk_{id} which can also leak just as in the standard context of public-key encryption. We stress that leakage from one user's secret-key should have no effect on the security of another's. This is because even in the standard definition of security for IBE (with no leakage), the attacker can access any secret-key of its choosing, except for that of the user being attacked.

The discussion above suggests that a proper definition of IBE in the CML model should address the issues of leakage from the msk as well as leakage from the individual secret keys sk_{id}. While we present the general model, our solution can only handle the latter leakage type, namely one from the individual secret-keys sk_{id}. On one hand, one could argue that since the msk is a much more appealing target (as it is related to the security of *all* users), it may attract more attention from attackers and thus deserves more protection. This point of view makes our solution seem less appealing. On the other hand, a situation where the trusted authority does not leak can be justified by arguing that in real-life situations, the trusted authority has much more resources than the specific users, and therefore can afford to take counter-measures at the implementation level to prevent key leakage. In any case, finding schemes that are resilient to the more general class of attacks remains an interesting open problem.

Most generally, an IBE scheme in the CML model consists of the algorithms Setup, $\mathsf{Extract}$, $\mathsf{Update}_{\text{master}}$, Enc, Dec, $\mathsf{Update}_{\text{user}}$, where $\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec}$ are identical to those in the standard definition of IBE, the new algorithms $\mathsf{Update}_{\text{master}}, \mathsf{Update}_{\text{user}}$ correspond to the key-update procedures of the trusted authority and of an individual user, respectively. The input-output functionalities of these procedures are similar to that of the update procedure in the encryption case. In the interest of keeping the discussion at an abstract level, we refrain from giving formal definitions.

An attack on an IBE scheme in the CML, therefore, is modeled as follows. The trusted authority uses the Setup procedure to generate (msk, pp). The setup process may leak some information to the adversary. The attacker can then (adaptively) specify as many values of id for which it wants to see sk_{id}. In addition, it can make additional leakage queries to msk ($\mathsf{Update}_{\text{master}}$ is used to refresh msk when required and this update process may also leak). After this query phase, the attacker decides which identity id^* it wants to attack. The trusted authority uses the $\mathsf{Extract}$ procedure to compute sk_{id^*} (this is, of course, not revealed to the adversary). This process may also leak. At this point, the adversary can make more queries as above. It of course cannot ask for the secret-key of id^* but can ask for leakage from it ($\mathsf{Update}_{\text{user}}$ is used to update sk_{id^*} when required, and may also leak). After this second query phase, the adversary decides on messages m_0, m_1 and receives the challenge ciphertext c_b, which is an encryption of m_b, where b is a random bit sampled by the challenger. A third query phase now commences, but now no leakage queries are allowed, only sk_{id}

queries for $id \neq id^*$. After this final query phase, the adversary makes its guess b' as to the value of b.

As explained above, in this work we only present a solution for the case where In the case where no leakage from msk is allowed. This means no leakage from the Setup and $\mathsf{Extract}$ procedures as well. In such case, the above attack becomes simpler (specifically, the types of queries the adversary can make significantly reduces). We will in fact consider an even weaker security definition that corresponds to the CML version of *selective identity security* (or just selective security for short). In selective security, the adversary is required to decide on the value of id^* at the beginning of the experiment (before the public parameters are generated). A trivial reduction shows that a selective-secure scheme with identity space $\{0,1\}^m$, where m is polynomially related to the security parameter ($m = k^\epsilon$), can be extended to a fully-secure scheme with identity space $\{0,1\}^*$ using a family of collision resistant hash function, with a loss of 2^{-m} factor in security. It is easy to see that this transformation carries over to the CML model (at least in the case where there is no leakage from msk). This transformation is rather standard and we will not describe it here in detail.

We give a formal definition of selective security in the CML model in Section 7.2.

7.2 Selective IBE Security in the CML Model — Definition

We formally define the model of attack for selective secure IBE in the CML model. The definition is in the spirit of the discussion in Section 7.1 above.

Definition 7.1. *An* IBE *scheme* $(\mathsf{Setup}, \mathsf{Extract}, \mathsf{Update}_{\text{user}}, \mathsf{Enc}, \mathsf{Dec})$ *is selectively secure under chosen plaintext attack in the* CML *model, with leakage rate* (ρ_M, ρ_U)*, if any* PPT *adversary succeeds in the following game with probability negligibly close to* $\frac{1}{2}$.

1. Identity selection. *The adversary decides on an identity* id^* *for which it wants to break the security of the encryption. It sends* id^* *to the challenger.*

2. Setup. *The challenger runs* $\mathsf{Setup}(1^k)$ *to generate* (msk, pp) *and sends* pp *to the adversary.*

3. Secret-key generation. *The challenger runs* $sk_{id^*,0} \leftarrow \mathsf{Extract}(msk, pp, id^*)$. *It further sets* $i := 0$ *and* $L_0 := 0$.

4. Query 1. *The adversary makes queries of the following types:*

   - *Extraction queries* $(\mathsf{extract}, id)$ *where* $id \neq id^*$. *The challenger generates* $sk_{id} := \mathsf{Extract}(sk, pp, id)$ *and sends this value to the adversary.*
   - *Leakage queries* (leak, f), *where* f *is a circuit. If* $L_i + \left|f(sk_{id^*,i})\right| \leq \rho_M \cdot \left|sk_{id^*,i}\right|$ *then the challenger returns* $f(sk_{id^*,i})$ *to the adversary and sets* $L_i := L_i + \left|f(sk_{id^*,i})\right|$. *Otherwise, the challenger aborts.*
   - *Update queries of the form* (update, f), *where* f *is a poly-size circuit. The challenger chooses randomness* r *for the updating process, and computes* $f(sk_{id^*,i}, r)$. *If* $|f(sk_{id^*,i}, r)| > \rho_U \cdot |sk_{id^*,i}|$ *or if* $L_i + |f(sk_{id^*,i}, r)| > \rho_U \cdot \left|sk_{id^*,i}\right|$ *then it aborts. Otherwise, it returns* $f(sk_{id^*,i}, r)$ *to the adversary, and it sets* $sk_{id^*,i+1} \leftarrow \mathsf{Update}_{\text{user}}(sk_{id^*,i}, pp, r)$, $L_{i+1} \leftarrow |f(sk_{id^*,i}, r)|$, *and* $i \leftarrow i + 1$.

5. Challenge. *The adversary sends two messages* m_0, m_1 *to the challenger. The challenger flips a coin* $b \xleftarrow{\$} \{0,1\}$ *and computes* $c \leftarrow \mathsf{Enc}(pk, m_b)$. *It sends* c *to the adversary.*

6. Query 2. *The adversary makes additional extraction queries* $(\mathsf{extract}, id)$ *(with* $id \neq id^*$*) to which the challenger answers the same as above.*

7. Finish. *The adversary outputs a guess $b' \in \{0,1\}$ as to the value of b.*

The adversary **succeeds** *if $b' = b$.*

7.3 Our Construction

We adapt our CML secure public-key encryption scheme into a CML secure IBE scheme. From now on, we focus our attention on the case of selective security where no leakage from msk is allowed, as described above. Our scheme can be seen as a combination of our basic scheme from Section 6 and the d-linear IBE scheme of [BK10].

Consider the following simplified version of the [BK10] construction. The public parameters are a set of $2m+1$ random 2×2 matrices in the exponent (where $\mathcal{ID} = \{0,1\}^m$ is the identity space): $g^{\mathbf{A}_0}$, $\{g^{\mathbf{A}_{i,b}}\}_{i \in [m], b \in \{0,1\}}$. The "public key" that corresponds to identity id (i.e. the part of the public parameters that is required to encrypt for id) is $g^{\mathbf{A}_{id}}$ where $\mathbf{A}_{id} = [\mathbf{A}_0 \| \mathbf{A}_{id_1} \| \cdots \| \mathbf{A}_{id_m}]$. The corresponding secret key sk_{id} is a random vector in $\ker(\mathbf{A}_{id})$. Encryption and decryption are performed very similarly to our scheme $\mathcal{L}[\ell]$: the ciphertext is a vector in the exponent $g^{\mathbf{v}^T}$ where an encryption of 0 uses $\mathbf{v}^T$ that is uniform in $\mathrm{Span}(\mathbf{A})$ and and an encryption of 1 uses a completely uniform $\mathbf{v}^T$. The master secret key is the matrix $\mathbf{A}_0$ (in explicit representation). It is immediate that using $\mathbf{A}_0$, one can sample uniformly from $\ker(\mathbf{A}_{id})$ for all id.

The proof of security in [BK10] is as follows. Since we consider the case of selective security, the attacker decides on id^* before the public parameters are generated. This enables generating the public parameters such that all matrices $\mathbf{A}_{i,1-id_i^*}$ (i.e. all matrices that do not "play" in $\mathbf{A}_{id^*}$) are explicitly known, the matrix $g^{\mathbf{A}_{id^*}}$ is generated together with the challenge ciphertext $g^{\mathbf{v}^T}$ such that a successful adversary decides whether the matrix whose first two rows are $g^{\mathbf{A}_{id^*}}$ and its third row is $g^{\mathbf{v}^T}$ is of rank 2 or 3, thus breaking the linear assumption. Clearly for all $id \neq id^*$, the matrices that are explicitly known can be used to sample from $\ker(\mathbf{A}_{id})$ and thus one can answer the adversary's queries.

Adapting this idea to achieve key resiliency is straightforward using what we know of $\mathcal{L}[\ell]$. We now rather than generating the secret key as a single vector in $\ker(\mathbf{A}_{id})$, generate it as two such vectors. This will enable $\mathsf{Update}_{\text{user}}$ operations just as we do in $\mathcal{L}[\ell]$. Therefore we can reduce the CML security of the resulting IBE scheme to the CML security of $\mathcal{L}[\ell]$.

We formally define our IBE scheme $\mathcal{IBL}[m]$ in Figure 4. Correctness immediately follows. We discuss its security guarantees below.

7.4 Overview of the Security Proof

The following theorem establishes the selective security of our scheme in the CML model (with no leakage from the master key). Since the proof is just a combination of ideas from Section 6 and from [BK10], we only sketch it here.

Theorem 7.1. *For any polynomially bounded parameter m, the* IBE *scheme $\mathcal{IBL}[m]$ is selective secure under chosen plaintext attack in the* CML *model, such that for all $\gamma, c > 0$ the scheme is secure with leakage rate*

$$(\rho_U, \rho_M) = \left(\frac{c \cdot \log k}{\ell \cdot \log p}, \frac{m - 2 - \gamma}{2(m+1)}\right) .$$

Proof sketch. We prove by reducing the security of $\mathcal{IBL}[m]$ to that of the scheme $\mathcal{L}[\ell]$ presented in Section 6, with parameter $\ell \doteq 2m+2$. Assigning the said value of ℓ into the parameters of Theorem 6.1 immediately implies the leakage parameters in the theorem statement above.

Identity-based encryption scheme $\mathcal{IBL}[m]$

- **Parameters.** The scheme is parameterized by groups $\mathbb{G}, \mathbb{G}_T$ of prime order p such that there exists a bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let g be a canonic generator for $\mathbb{G}$ (and thus $e(g, g)$ is a generator for $\mathbb{G}_T$).

  The identity space of the scheme is $\mathcal{ID} = \{0, 1\}^m$, where $m \in \mathbb{N}$ is polynomially related to the security parameter. We require an exponential identity space so that it is possible to apply known methods for obtaining fully secure IBE from a selective-secure one.

  Our scheme is a bit-encryption scheme, namely its message space is $\mathcal{M} = \{0, 1\}$.

- **Setup.** The setup procedure runs as follows. It samples $\mathbf{A}_0 \xleftarrow{\$} \mathbb{Z}_p^{2\times2}$ and sets $msk \leftarrow \mathbf{A}_0^{-1}$ (if $\mathbf{A}_0$ is not invertible, the procedure fails).

  It then samples $g^{\mathbf{A}_{i,b}} \xleftarrow{\$} \mathbb{G}^{2\times2}$ for all $i \in [m]$ and $b \in \{0, 1\}$ (note that no explicit generation of $\mathbf{A}_{i,b}$ is required). The public parameters are set to $pp{:=}\{g^{\mathbf{A}_0}, \{g^{\mathbf{A}_{i,b}}\}_{i,b}\}$.

  We denote $\mathbf{A}_{id} \doteq [\mathbf{A}_0 \| \mathbf{A}_{id_1} \| \cdots \| \mathbf{A}_{id_m}]$

- **Extract.** To extract a secret key for identity $id \in \{0, 1\}^m$ we sample $\mathbf{x}_1, \mathbf{x}_2 \xleftarrow{\$} \ker(\mathbf{A}_{id})$ and set $g^{\mathbf{X}}{:=}[g^{\mathbf{x}_1} \| g^{\mathbf{x}_2}]$. We then define $sk_{id} \doteq g^{\mathbf{X}}$.

  In order to sample $\mathbf{x} \xleftarrow{\$} \ker(\mathbf{A}_{id}) \subseteq \mathbb{Z}_p^{2m+2}$, we define $\mathbf{B}_{id} \doteq [\mathbf{A}_{id_1} \| \cdots \| \mathbf{A}_{id_m}]$ to be the part of $\mathbf{A}_{id}$ that does not contain $\mathbf{A}_0$ and recall that $\mathbf{A}_0$ is invertible and $msk = \mathbf{A}_0^{-1}$. The marginal distribution of any $2m$ elements in the vector $\mathbf{x}$ is uniform. Therefore we sample $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^{2m}$ and set
$$g^{\mathbf{x}}{:=}\begin{bmatrix} g^{\mathbf{r}} \\ g^{-\mathbf{A}_0^{-1}\cdot\mathbf{B}_{id}\cdot\mathbf{r}} \end{bmatrix} .$$

- **Update (user).** To update a secret-key $sk_{id} = g^{\mathbf{X}}$, we sample $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_2(\mathbb{Z}_p^{2\times2})$ and output $sk'_{id} = g^{\mathbf{X}\cdot\mathbf{T}}$.

- **Encrypt.** To encrypt a message $\mu \in \{0, 1\}$ for identity id, using the public parameters $pp = \{g^{\mathbf{A}_0}, \{g^{\mathbf{A}_{i,b}}\}_{i,b}\}$, we compute the ciphertext $c \doteq g^{\mathbf{v}^T}$ where $g^{\mathbf{v}^T} \in \mathbb{G}^{2m+2}$ is defined such that
$$\mathbf{v}^T \xleftarrow{\$} \begin{cases} \mathrm{Span}(\mathbf{A}_{id}) & \text{if } \mu = 0 \\ \mathbb{Z}_p^{2m+2} & \text{if } \mu = 1 \end{cases} .$$
  Clearly, $g^{\mathbf{v}^T}$ can be efficiently sampled in both cases.

- **Decrypt.** To decrypt a ciphertext $c = g^{\mathbf{v}^T}$ using secret key $sk_{id} = g^{\mathbf{X}}$, we use the bilinear map to compute $e(g^{\mathbf{v}^T}, g^{\mathbf{X}}) = e(g, g)^{\mathbf{v}^T\cdot\mathbf{X}}$ and output $\mu = 0$ if the result is equal to $e(g, g)^{\mathbf{0}}$ and $\mu = 1$ otherwise.

Figure 4: IBE scheme in the CML model.

Consider an adversary $\mathcal{A}$ for the selective IBE security in the CML model of $\mathcal{IBL}[m]$. Then an adversary $\mathcal{B}$ for the semantic security of $\mathcal{L}[2m+2]$, with essentially the same success probability can be defined as follows. The adversary $\mathcal{B}$ will use the challenger of $\mathcal{L}[2m+2]$ to simulate a challenger for $\mathcal{IBL}[m]$.

1. $\mathcal{B}$ simulates $\mathcal{A}$ to receive the value of the target identity id^*.

2. $\mathcal{B}$ gets a public key for $\mathcal{L}[2m + 2]$ from the challenger. This public key is of the form $g^{\mathbf{A}}$ for $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_p^{2\times(2m+2)}$. $\mathcal{B}$ interprets this value as $g^{\mathbf{A}_{id^*}}$. This interpretation defines values for $g^{\mathbf{A}_0}$ and $\{g^{\mathbf{A}_{i,id_i^*}}\}_i$.

3. $\mathcal{B}$ samples matrices $\mathbf{A}_{i,1-id_i^*} \xleftarrow{\$} \mathbb{Z}_p^{2\times 2}$ for all $i \in [m]$. It then sets $pp := \{g^{\mathbf{A}_0},\ \{g^{\mathbf{A}_{i,b}}\}_{i,b}\}$ and sends this to $\mathcal{A}$ as the set of public parameters for the scheme $\mathcal{IBL}[m]$. Note that the distribution of pp here is exactly as prescribed.

4. The first query phase of $\mathcal{A}$ is simulated. For the queries made by $\mathcal{A}$, $\mathcal{B}$ answers as follows.

   - *Extraction queries* of the form $(\mathsf{extract}, id)$ for $id \neq id^*$ can be answered by $\mathcal{B}$ by considering a location $i' \in [m]$ for which $id_{i'} \neq id^*_{i'}$. This implies that $\mathbf{A}_{i',id}$ is explicitly known to $\mathcal{B}$ who can use it to generate sk_{id} (assuming that $\mathbf{A}_{i',id}$ is invertible which is the case with all but negligible probability).
   - *Leakage queries* of the form (leak, f) and update queries of the form (update, f) are forwarded to the challenger of $\mathcal{L}[2m+2]$ as leakage and update queries. The challenger's answer is then forwarded back to $\mathcal{A}$.

     We notice that by definition, the distribution of answers that $\mathcal{A}$ gets in this simulation is identical to that produced by a "legal" challenger for $\mathcal{IBL}[m]$.

5. $\mathcal{B}$ simulates the challenge phase of $\mathcal{A}$ to obtain the messages μ_0, μ_1. It forwards these messages to the challenger to obtain the challenge ciphertext c, which is forwarded to $\mathcal{A}$.[13]

6. The second query phase of $\mathcal{A}$ is simulated exactly as the first one.

7. When $\mathcal{A}$ terminates and returns a guess b', $\mathcal{B}$ terminates as well and returns the same b'.

It is straightforward that the simulation of $\mathcal{A}$'s view is accurate up to a negligible term and also that if $\mathcal{A}$ wins in its experiment then so does $\mathcal{B}$ in its. The result thus follows. □

8 Continual Leakage Resilience: From Encrypting to Signing

In this section, we show how to construct a signature scheme that is secure against continual leakage, using as a building block any encryption scheme secure against continual leakage. Specifically, given an encryption scheme that is semantically secure in the CML model with leakage rate (ρ_G, ρ_U, ρ_M), we construct a signature scheme that is existentially unforgeable under adaptive chosen message attacks in the CML model, with leakage rate $(\rho_G, \rho_U, \rho_S, \rho_M)$, where the value of ρ_S, which is the leakage rate from the signing algorithm, is given below.

In fact, our construction only relies on the ability to verify that a key-pair (sk, pk) is valid, and our security proof reduces the unforgeability of the signature scheme to actually finding a valid sk for the encryption scheme. Thus a much weaker primitive (a "leakage resilient one-way function") could have been used. In the interest of keeping the presentation simple, we do not explicitly define this "middle stage" and work directly with an encryption scheme.

Throughout this section, we assume a stronger leakage resilience property on the CML secure encryption scheme that underlies our construction: We assume that the scheme remains secure even if the size of the leakage is unbounded, so long as the image of the leakage function has bounded cardinality. Namely, we allow the leakage to be arbitrarily long, but require that the secret key sk has "enough" min-entropy left conditioned on the leakage. We stress that our encryption scheme, presented in Section 6, is indeed secure with respect to this (stronger) security definition.

First, in Section 8.1, we construct such a signature scheme with $\rho_S = 0$; i.e., the scheme does not tolerate any leakage from the signing process. In addition to relying on an encryption scheme

[13]Since both schemes encrypt bit by bit, one can just assume that $\mu_0 = 0$ and $\mu_1 = 1$, however we chose to give a more general description.

secure against continual leakage, here we also rely on the existence of an unbounded simulation-sound NIZK proof system (see Definition 4.2). This scheme follows the paradigm of Katz and Vaikuntanathan [KV09] for constructing leakage-resilient signature schemes in the *bounded* leakage model, although we must modify the scheme appropriately so as to tolerate *continual* leakage.

Then, in Section 8.2, we construct such a signature scheme with $\rho_S = \rho_M/\alpha(k)$, where $\alpha(k) = \omega(\log k)$, and the smaller $\alpha(k)$ is, the stronger the underlying assumption is. Namely, we rely on the assumption that there exists a family $\mathcal{H} = \{\mathcal{H}_k\}_{k\in\mathbb{N}}$ of collision resistant hash functions such that for every $h \in \mathcal{H}_k$, $h : \{0,1\}^* \to \{0,1\}^{\alpha(k)}$. Moreover, we rely on the existence of lossy trapdoor functions with oblivious representation (see Definition 4.4), and we rely on the existence of *short* non-interactive arguments (see Definition 4.3). We note that all known constructions of the latter, were proven to be secure only in the random oracle model [Kil92, Mic00, BG08]. Finally, we note that in order to deal with leakage from the signing process we generalize the CML model, as explained in Section 8.2 (see Definition 8.1).

8.1 Continual Leakage Resilience — No Leakage from Signing Process

In this section we construct a signature scheme $\mathcal{T}_1 = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ that is existentially unforgeable under adaptive chosen message attack in the CML model with leakage rate $(\rho_G, \rho_U, 0, \rho_M)$. Our signature scheme relies on the following ingredients:

- A public-key encryption scheme $\mathcal{L} = (\mathcal{L}\text{-}\mathsf{Gen}, \mathcal{L}\text{-}\mathsf{Enc}, \mathcal{L}\text{-}\mathsf{Dec}, \mathcal{L}\text{-}\mathsf{Update})$ that is semantically secure in the CML model with leakage rate (ρ_G, ρ_U, ρ_M). We assume the encryption scheme $\mathcal{L}$ has the following property:[14] there exists a negligible function μ and a *deterministic* polynomial-time predicate T such that $T(pk, sk) = 1$ iff $(pk, sk) \leftarrow \mathcal{L}\text{-}\mathsf{Gen}(1^k)$ or sk correctly decrypts ciphertexts encrypted using pk except with probability $\mu(k)$).

- A *dense* public-key encryption scheme $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ that is semantically secure (with no leakage guarantees). The "dense" property means that the public key is indistinguishable from a random string, and so public keys (or at least strings indistinguishable from valid public keys) can be sampled "obliviously" without knowledge of the corresponding secret key.

  Such schemes are known to exist e.g. under the DDH or the LWE assumptions (e.g. [Gam84, Reg05]). In fact, even our scheme from Section 6 has this property.

- An unbounded simulation-sound NIZK proof system $\Pi = (\ell, P, V, S_1, S_2)$ for the following language L:

$$L = \left\{(m, c, pk', pk) : \exists sk, r \text{ s.t. } c = \mathsf{Enc}_{pk'}(sk; r) \text{ and } T(pk, sk) = 1\right\}.$$

  Namely, $(m, c, pk', pk) \in L$ if and only if c is an encryption (using encryption scheme $\mathcal{E}$ and the public key pk') of a secret key sk that corresponds to pk (with respect to encryption scheme $\mathcal{L}$).[15]

Our signature scheme $\mathcal{T}_1 = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ is defined in Figure 5.

[14]In fact, we can use a slightly weaker property that *all* encryption schemes have, but for the sake of readability, we make this simplifying assumption.

[15]One may notice that the m part of the input does not participate in the verification process. Namely, whether $(m, c, pk', pk) \in L$, or not, is independent of the value of m. As demonstrated in [KV09], having m as a part of the input makes the NIZK proof attached to a specific message, and prevents "cut and paste" attacks.

Signature scheme $\mathcal{T}_1$

Key generation. The key-generation algorithm Gen does as follows:

1. Sample a key-pair for the leakage-resilient encryption scheme $(pk, sk) \leftarrow \mathcal{L}\text{-}\mathsf{Gen}(1^k)$, using secret/public randomness as specified by $\mathcal{L}\text{-}\mathsf{Gen}$.
2. Using public randomness, sample (obliviously) a public key pk' for the (ordinary) encryption scheme $\mathcal{E}$.
3. Using public randomness, sample a string $\mathsf{crs} \xleftarrow{\$} \{0,1\}^{\ell(k)}$.

The signing key is sk, and the verification key is $vk = (pk, pk', \mathsf{crs})$.

Signing. Given inputs m, a secret key sk, and a verification key $vk = (pk, pk', \mathsf{crs})$, do:

1. Choose a random string r and compute $c \leftarrow \mathsf{Enc}_{pk'}(sk; r)$.
2. Compute a proof π for the statement $(m, c, pk', pk) \in L$, with respect to the common random string crs, using (sk, r) as the witness. Namely, compute $\pi \leftarrow P_{\mathsf{crs}}((m, c, pk', pk), (sk, r))$.

Output $\sigma = (c, \pi)$ as a signature for m.

Verifying. To verify a signature $\sigma = (c, \pi)$ on a message m with respect to the verification key $vk = (pk, pk', \mathsf{crs})$, check whether π is a valid proof of the statement $(m, c, pk', pk) \in L$ with respect to the common random string crs.

Updates. The update procedure Update is identical to the update procedure of $\mathcal{L}$, using secret/public randomness as specified by $\mathcal{L}\text{-}\mathsf{Update}$.

Figure 5: Signature scheme in the CML model.

Theorem 8.1. *Let $\mathcal{L}$ be a semantically secure public-key encryption scheme in the* CML *model, with leakage rate (ρ_G, ρ_U, ρ_M); let $\mathcal{E}$ be a semantically secure dense public-key encryption scheme; and let Π be an unbounded simulation-sound NIZK proof system. Then signature scheme $\mathcal{T}_1 = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ described in Figure 5 is existentially unforgeable under adaptive chosen message attacks in the* CML *model, with leakage rate $(\rho_G, \rho_U, 0, \rho_M)$.*

Proof. Let $\mathcal{F}$ be a PPT forger attacking $\mathcal{T}_1$ with the specified leakage rate. We consider a sequence of experiments, and let $\Pr_i[\cdot]$ denote the probability of an event in Experiment i. We let Succ denote the event that $\mathcal{F}$ succeeds, as in Definition 3.2.

Experiment 0. This is the experiment of Definition 3.2.

Experiment 1. We introduce the following differences with respect to the preceding experiment: when setting up the public key, we now generate the common random string crs of the simulation-sound NIZK by computing $(\mathsf{crs}, \tau) \leftarrow S_1(1^k)$. We also generate the public key pk' by running the key-generation algorithm $(pk', sk') \leftarrow \mathsf{Gen}(1^k)$. Note that when answering leakage queries of $\mathcal{F}$, the secret randomness is unchanged and crs, pk' are treated as public randomness just as in the previous experiment. It follows easily that the difference $|\Pr_1[\mathsf{Succ}] - \Pr_0[\mathsf{Succ}]|$ is negligible.

Experiment 2. We modify the preceding experiment by answering signing queries as follows: to sign m, generate $c \leftarrow \mathsf{Enc}_{pk'}(sk)$ as before but then compute π as $\pi \leftarrow S_2((m, c, pk', pk), \mathsf{crs}, \tau)$. The (adaptive) zero-knowledge property of Π implies that $|\Pr_2[\mathsf{Succ}] - \Pr_1[\mathsf{Succ}]|$ is negligible. Note that we rely here on the fact that there is no leakage from the signing process (i.e., $\rho_S = 0$).

Experiment 3. We modify the preceding experiment by answering signing queries as follows: to sign m, now compute $c \leftarrow \mathsf{Enc}_{pk'}(0^{|sk|})$ (and then compute π as in Experiment 2). Semantic security of encryption scheme $\mathcal{E}$ implies that $|\Pr_3[\mathsf{Succ}] - \Pr_2[\mathsf{Succ}]|$ is negligible. Again, we rely here on the fact that there is no leakage from the signing process.

We now prove that $\Pr_3[\mathsf{Succ}]$ is negligible, by reduction to the semantic security of public-key encryption scheme $\mathcal{L}$ in the CML model. Construct the following PPT adversary $\mathcal{A}$ attacking $\mathcal{L}$:

1. Generate $(pk', sk') \leftarrow \mathsf{Gen}(1^k)$ and $(\mathsf{crs}, \tau) \leftarrow S_1(1^k)$.
2. Run $\mathcal{F}$ as a subroutine. Given f as output by $\mathcal{F}$, define $f_{\mathcal{L}}(\cdot,\cdot) \doteq f(\cdot,\cdot,pk',\mathsf{crs})$ and output the leakage function $f_{\mathcal{L}}$. (Note that $f_{\mathcal{L}}(r,p) \leq \rho_G \cdot |r|$ iff $f(r,p,pk',\mathsf{crs}) \leq \rho_G \cdot |r|$.) Obtain in return a public key pk, public randomness p, and leakage $f_{\mathcal{L}}(r,p)$. Give to $\mathcal{F}$ the verification key (pk, pk', crs), public randomness p, pk', crs, and leakage $f_{\mathcal{L}}(r,p) = f(r,p,pk',\mathsf{crs})$.
3. Continue running $\mathcal{F}$, responding to its signing queries as specified in Experiment 3. (Recall that we do not allow leakage during signing queries.)
4. When $\mathcal{F}$ makes a leakage query (leak, f), adversary $\mathcal{A}$ forwards this query to its own leakage oracle and then forwards the response to $\mathcal{F}$. Note that the size of the secret key in $\mathcal{T}_1$ is identical to the size of the secret key in $\mathcal{L}$, so the relative leakage bound ρ_M is respected.
5. When $\mathcal{F}$ makes an update query (update, f), adversary $\mathcal{A}$ forwards this query to own update oracle and then forwards the response to $\mathcal{F}$.
6. Assuming the challenger has not aborted, $\mathcal{F}$ outputs a pair (m, σ) with $\sigma = (c, \pi)$. If $\mathcal{F}$ succeeds, then $\mathcal{A}$ uses the secret key sk' to decrypt the ciphertext c and obtain $sk := \mathsf{Dec}_{sk'}(c)$. If (pk, sk) is not a valid key-pair, then $\mathcal{A}$ aborts. Otherwise, $\mathcal{A}$ outputs two distinct messages m_0, m_1 and receives a challenge ciphertext $c_b \leftarrow \mathcal{L}\text{-}\mathsf{Enc}_{pk}(m_b)$. It then uses the secret key sk to decrypt c_b and thus guess b.

Note that $\mathcal{A}$ provides a perfect simulation of Experiment 3 for $\mathcal{F}$. Therefore, $\mathcal{F}$ succeeds in outputting a valid forgery with probability exactly $\Pr_3[\mathsf{Succ}]$. Simulation soundness of Π, together with the fact that $\sigma = (c, \pi)$ is a valid signature, implies that (pk, sk) is a valid key-pair with all but negligible probability. Thus,

$$\begin{aligned}\Pr[\mathcal{A} \text{ succeeds}] &\geq \Pr[\mathcal{A} \text{ succeeds} \,|\mathcal{F} \text{ succeeds in Experiment 3}] \cdot \Pr[\mathcal{F} \text{ succeeds in Experiment 3}] \\ &\geq (1 - \text{negl}(k)) \cdot \Pr[\mathcal{F} \text{ succeeds in Experiment 3}]\ ,\end{aligned}$$

and so $\Pr_3[\mathsf{Succ}]$ must be negligible, as desired. □

8.2 Continual Leakage Resilience — With Leakage from the Signing Process

The construction in the previous section is not necessarily resilient to leakage that occurs during the signing process. In particular, if part of the randomness used to encrypt sk (during the course of computing a signature) is leaked, then there is no longer any guarantee that sk remains secret. Similarly, if part of the randomness used to compute the NIZK proof is leaked then there is no longer any guarantee that the witness (which includes sk) remains hidden.

Here we show how to modify the signature scheme so as to obtain security against continual leakage, even if it occurs during the signing process. Interestingly, we solve this by making our scheme leak *more*: In our new scheme, the signatures themselves leak information about the signing-key. Thus the scheme does not even conform with the standard definition of security (without leakage). In the CML model, however, this caveat is tolerable, since we have a method for refreshing the key. We have to require, however, that the signing-key is periodically refreshed, even regardless of any adversarial action, resulting in a variant of CML security. In addition, we need to rely on stronger assumptions. We discuss these in turn.

In our modified model of continual leakage resilience, we add an additional ℓ-bit leakage that occurs each time a signature is generated; this leakage is *in addition to* any leakage explicitly specified by the adversary. As explained above, this additional leakage represents information included in the signature itself (that is needed in order to verify). To ensure meaningful security, as usual, we need to ensure that the *total* leakage is some bounded fraction of the secret-key length. As a consequence, the signer must now run the update operation every time it issues a certain number of signatures, even if it is willing to assume that no additional leakage occurred. A formal definition, for the case of general ℓ, follows.

Definition 8.1. *Signature scheme* $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver}, \mathsf{Update})$ *is existentially unforgeable under adaptive chosen message attack in the* CML *model, with leakage rate* $(\rho_G, \rho_U, \rho_S, \rho_M)$ *and signature leakage* $\ell = \ell(k)$*, if any* PPT *forger succeeds in the following game with only negligible probability.*

1. Initialize. *The forger specifies a circuit* f *with* $|f(r, p)| \leq \rho_G \cdot |r|$ *for all* r, p. *The challenger chooses "secret randomness"* r *and "public randomness"* p*, generates* $(sk_0, vk) \leftarrow \mathsf{Gen}(1^k; r, p)$*, sends* $(vk, p, f(r, p))$ *to the forger, and sets* $i := 0$ *and* $L_0 := |f(r, p)|$.

2. Signatures, leakage, and updates. *The forger makes queries of the following types:*

   - *Signing queries* (sign, m, f)*, where* f *is a circuit with* $|f(sk, r, p)| \leq \rho_S \cdot (|sk| + |r|)$ *for all* sk, r, p. *The challenger chooses "secret randomness"* r *and "public randomness"* p*, and computes* $\sigma \leftarrow \mathsf{Sign}_{sk_i, vk}(m; r, p)$. *If* $L_i + |f(sk_i, r, p)| + \ell \leq \rho_M \cdot |sk_i|$ *then the challenger returns* $(\sigma, p, f(sk_i, r, p))$ *to the forger and sets* $L_i := L_i + |f(sk_i, r, p)| + \ell$. *Otherwise, the challenger aborts.*

   - *Update queries* (update, f)*, where* f *is a circuit with* $|f(sk, r, p)| \leq \rho_U \cdot (|sk| + |r|)$ *for all* sk, r, p. *The challenger chooses "secret randomness"* r *and "public randomness"* p*, and computes* $sk_{i+1} := \mathsf{Update}_{vk}(sk_i; r, p)$. *If* $L_i + |f(sk_i, r, p)| \leq \rho_M \cdot |sk_i|$ *then the challenger returns* $(p, f(sk_i, r, p))$ *to the forger, sets* $i := i + 1$*, and sets* $L_{i+1} := |f(sk_i, r, p)|$. *Otherwise, the challenger aborts.*

   - *Leakage queries* (leak, f)*, where* f *is a circuit. If* $L_i + |f(sk_i)| \leq \rho_M \cdot |sk_i|$ *then the challenger returns* $f(sk_i)$ *to the forger and sets* $L_i := L_i + |f(sk_i)|$. *Otherwise, the challenger aborts.*

3. Finish. *Assuming the challenger did not abort, the forger outputs* (m^*, σ^*).

The forger **succeeds** *if it never made the query* (sign, m^*)*, and* $\mathsf{Ver}_{vk}(m^*, \sigma^*) = 1$.

Note that Definition 8.1 generalizes Definition 3.2, and the two coincide for $\ell = 0$.

Our construction, presented in Figure 6, can be viewed as modifying the scheme from the previous section in two ways:

- Recall that in our previous construction, each signature contained an encryption of sk — the secret-key of the underlying leakage-resilient encryption scheme. Now, rather than using a public-key encryption scheme to encrypt sk, we "encrypt" it using a family of lossy trapdoor functions $(S_{inj}, S_{loss}, G, G^{-1})$, in a particular way (see Figure 6).

- Instead of using a (simulation-sound) NIZK proof system, we use a (non-interactive) argument system with short proofs; i.e., proofs that are significantly shorter than the witness size. Intuitively, even though we cannot say much about what information might be leaked by a proof of some statement, we *can* bound the leakage by the length of the proof. (For our purposes, we can obtain proofs of size $\omega(\log^2 k)$.) Proofs are with respect to the following **NP** language: $L' = \left\{\left(\{y_i\}_{i=1}^k, \{s_i\}_{i=1}^k, pk\right)\right\}$ for which there exist $sk, \{r_i\}$ such that:

$$y_i = G_{s_i}(r_i) \bigwedge \bigoplus_{i=1}^k r_i = sk \bigwedge T(pk, sk) = 1.$$

The scheme we describe achieves only a weaker notion of security, termed *a-priori unforgeability against chosen-message attacks* (a-cma) in [BK10], where the adversary is given a random "challenge" message m in advance, can request signatures on any messages other than m, and then succeeds only if it outputs a forgery on m. We then apply a recent transformation from [BK10] to obtain a scheme satisfying Definition 8.1.

Signature scheme $\mathcal{T}_2$

Key Generation. The key-generation algorithm Gen does as follows:

1. The message space of the scheme is $\{0,1\}^n$ where $n \in \mathbb{N}$ is related to the security parameter k by some function $n = \alpha(k)$ to be determined later.
2. Sample a pair of keys for the leakage-resilient encryption scheme $(pk, sk) \leftarrow \mathcal{L}\text{-}\mathsf{Gen}(1^k)$, using secret/public randomness as specified by $\mathcal{L}\text{-}\mathsf{Gen}$.
3. For $i \in [n]$ and $b \in \{0,1\}$, sample (obliviously, using public randomness) a key $s_{i,b}$ for a lossy trapdoor function.
4. Using public randomness, choose a string crs for a non-interactive argument system.

The signing key is sk, and the verification key is $vk = (pk, \{s_{i,b}\}, \mathsf{crs})$.

Signing. Given inputs m, a secret key sk, and a verification key $(pk, \{s_{i,b}\}, \mathsf{crs})$, do:

1. Using secret randomness, choose n random strings $r_1, \ldots, r_n$ such that $\bigoplus_{i=1}^n r_i = sk$.
2. Compute $y_i := G_{s_{i,m_i}}(r_i)$ for all i.
3. Using public randomness, compute a non-interactive argument π for the statement $(\{y_i\}_{i=1}^n, \{s_{i,m_i}\}_{i=1}^n, pk) \in L'$, using $sk, \{r_i\}_{i=1}^n$ as the witness.

Output $\sigma = (\{y_i\}_{i=1}^n, \pi)$ as a signature for m.

Verifying. To verify a signature $\sigma = (\{y_i\}, \pi)$ on a message m, with respect to the verification key $(pk, \{s_{i,b}\}, \mathsf{crs})$, check whether π is a valid proof of the statement $(\{y_i\}_{i=1}^n, \{s_{i,m_i}\}_{i=1}^n, pk) \in L'$ with respect to crs.

Updating. The update procedure Update is identical to the update procedure of the underlying leakage-resilient encryption scheme $\mathcal{L}$, using public/secret randomness as directed by $\mathcal{L}\text{-}\mathsf{Update}$.

Figure 6: Signature scheme in the CML model.

Theorem 8.2. *Let $\mathcal{L}$ be a semantically secure public-key encryption scheme in the* CML *model, with leakage rate (ρ_G, ρ_U, ρ_M); let $(S_{inj}, S_{loss}, G, G^{-1})$ be a family of* LTDF *with oblivious sampling (see Definition 4.4) with lossy parameter $\ell/2$; and let Π be a non-interactive argument system with proofs of length $\ell/2$. Then signature scheme $\mathcal{T}_2$ described in Figure 6 is a priori unforgeable under chosen-message attacks in the* CML *model, with leakage rate $(\rho_G, \rho_U, \rho_S, \rho_M)$, where $\rho_S = \rho_M$, and signature leakage ℓ.*

Proof. Let $\mathcal{F}$ be a PPT forger attacking $\mathcal{T}_2$, and let $\Pr[\mathsf{Succ}]$ denote the success probability of $\mathcal{F}$ with respect to the notion of *a priori* unforgeability (and the stated leakage parameters).

We first modify the experiment in the following way: Let $m = m_1 \cdots m_n$ be the random (challenge) message given to the adversary. Instead of generating all seeds $\{s_{i,b}\}$ obliviously, we now generate the seeds $\{s_{i,m_i}\}_{i=1}^n$ using S_{inj}, and generate the remaining seeds using S_{loss}. The properties of the LTDF immediately imply that this has only a negligible effect on the success probability of $\mathcal{F}$. Let $\Pr'[\mathsf{Succ}]$ denote the success probability of $\mathcal{F}$ in this modified experiment.

We show that $\Pr'[\mathsf{Succ}]$ is negligible, by reduction to the semantic security of $\mathcal{L}$ in the CML model. Construct the following PPT adversary $\mathcal{A}$ attacking $\mathcal{L}$:

1. Choose a random message $m \leftarrow \{0,1\}^n$ and give it to $\mathcal{F}$.
2. For all i, compute $(s_{i,m_i}, t_i) \leftarrow S_{inj}(1^n)$ and $(s_{i,\bar{m}_i}, \perp) \leftarrow S_{loss}(1^n)$. Also choose a random crs for the non-interactive argument system.
3. Run $\mathcal{F}$ as a subroutine. Given f as output by $\mathcal{F}$, define $f_{\mathcal{L}}(\cdot,\cdot) \doteq f(\cdot,\cdot,\{s_{i,b}\},\mathsf{crs})$ and output the leakage function $f_{\mathcal{L}}$. (Note that $f_{\mathcal{L}}(r,p) \le \rho_G \cdot |r|$ iff $f(r,p,pk',\mathsf{crs}) \le \rho_G \cdot |r|$.) Obtain in return a public key pk, public randomness p, and leakage $f_{\mathcal{L}}(r,p)$. Give to $\mathcal{A}$ the verification key $(pk, \{s_{i,b}\}, \mathsf{crs})$, public randomness $(p, \{s_{i,b}\}, \mathsf{crs})$, and leakage $f_{\mathcal{L}}(r,p) = f(r,p,\{s_{i,b}\},\mathsf{crs})$.
4. When $\mathcal{F}$ makes a signing query for a message m' with leakage function f, adversary $\mathcal{A}$ answers it as follows.
   (a) Find an index i such that $m'_i \neq m_i$.
   (b) For all $j \neq i$, choose random r_j and compute $y_j := G_{s_{j,m'_j}}(r_j)$.
   (c) Choose randomness r for the non-interactive argument system.
   (d) Make a leakage query (leak, g) where the circuit g has the values $\{r_j\}_{j\neq i}$ hard-wired into it, and on input sk it computes the following:
      - Set $r_i := sk \oplus \bigoplus_{j\neq i} r_j$ and compute $y_i \doteq G_{s_{i,m'_i}}(r_i)$.
      - Compute a non-interactive argument π for the statement $(\{y_i\}_{i=1}^n, \{s_{i,m'_i}\}_{i=1}^n, pk) \in L'$ with respect to crs, using witness $(sk, \{r_i\}_{i=1}^n)$ and randomness r.
      - Compute $f(sk, \{r_i\}_{i=1}^n, r)$.

      Finally, the circuit g on input sk outputs $(y_i, \pi, f(sk, \{r_i\}_{i=1}^n, r))$.
   (e) Give to $\mathcal{F}$ the signature $(\{y_i\}_{i=1}^n, \pi)$, public randomness r, and leakage $f(sk, \{r_i\}_{i=1}^n, r)$.

   The total leakage of $\mathcal{A}$'s query (in an information-theoretic sense) is $|f(sk,\{r_i\},r)| + \ell/2 + \ell/2 = |f(sk,\{r_i\},r)| + \ell$ bits, where $\ell/2$ bits are due to the leakage[16] from y_i and $\ell/2$ bits are due to π.
5. When $\mathcal{F}$ makes a leakage query (leak, f), adversary $\mathcal{A}$ forwards this query to its own leakage oracle and then forwards the response to $\mathcal{F}$. Since the size of the secret key in $\mathcal{T}_2$ is identical to the size of the secret key in $\mathcal{L}$, the global bound on the relative leakage is respected.
6. When $\mathcal{F}$ makes an update query (update, f), adversary $\mathcal{A}$ forwards this query to own update oracle and then forwards the response to $\mathcal{F}$.

[16]Although $|y_i| > \ell/2$, the value y_i only leaks $\ell/2$ bits (with all but negligible probability) in an information-theoretic sense.

7. Assuming the challenger has not aborted, $\mathcal{F}$ outputs a signature $\sigma = (\{y_i\}, \pi)$. If $\mathcal{F}$ succeeds (and so, in particular, the proof π is valid), then $\mathcal{A}$ uses the trapdoors $\{t_i\}$ to compute $\{r_i := G^{-1}_{s_{i,m_i}}(y_i)\}$ and finally $sk = \bigoplus_{i=1}^{n} r_i$. If (pk, sk) is not a valid key-pair, then $\mathcal{A}$ aborts. Otherwise, $\mathcal{A}$ outputs two distinct messages m_0, m_1 and receives a challenge ciphertext $c_b \leftarrow \mathcal{L}\text{-}\mathsf{Enc}_{pk}(m_b)$. It then uses the secret key sk to decrypt c_b and thus guess b.

Note that $\mathcal{A}$ provides a perfect simulation of the modified experiment for $\mathcal{F}$. Therefore, $\mathcal{F}$ succeeds in outputting a valid forgery with probability exactly $\Pr'[\mathsf{Succ}]$. Computational soundness of Π, together with the fact that $\sigma = (\{y_i\}, \pi)$ is a valid signature, implies that (pk, sk) is a valid key-pair with overwhelming probability. The theorem follows easily. $\square$

We now relate security in the model of a priori unforgeability to security against the standard notion of existential unforgeability.

Theorem 8.3. *Assume that there exists a signature scheme $\mathcal{T}_2$ (for messages of length at least $n = \alpha(k)$) that is* a priori unforgeable *under adaptive chosen-message attacks in the* CML *model, with leakage rate $(\rho_G, \rho_U, \rho_S, \rho_M)$ and signature leakage ℓ, and assume the existence of collision-resistant hash functions mapping strings of length k to strings of length $\alpha(k)$. Then there exists a signature scheme $\mathcal{T}_2'$ that is* existentially unforgeable *under adaptive chosen-message attacks with leakage rate $(\rho_G, \rho_U, \frac{\rho_S}{\alpha(k)}, \rho_M)$ and signature leakage $\ell \cdot \alpha(k)$.*

Note that $\alpha(k)$ can be made as small as $\omega(\log k)$, at the price of relying on an exponential hardness assumption.
This theorem follows from a very recent work of [BK10], which shows how to convert any signature scheme that is a priori unforgeable under adaptive chosen message attacks, into one which is existentially unforgeable under adaptive chosen message attacks.[17] Looking into their transformation, it is easy to see that their transformation also works for signature schemes in the CML model, with the following parameters.

Corollary 8.4 ([BK10])**.** *There exists a generic transformation for converting any signature scheme $\mathcal{T}_2$ that is a priori unforgeable under adaptive chosen message attacks in the* CML *model with leakage rate $(\rho_G, \rho_U, \rho_S, \rho_M)$ and signature leakage ℓ, into a new signature scheme $\mathcal{T}_2'$ that is existentially unforgeable under adaptive chosen message attacks in the* CML *model with leakage rate $(\rho_G, \rho_U, \frac{\rho_S}{\alpha(k)}, \rho_M)$ and signature leakage $\ell \cdot \alpha(k)$, assuming the message space of $\mathcal{T}_2$ is contained in $\{0,1\}^{\alpha(k)}$.*

Very loosely speaking, the new signing algorithm signs all the prefixes of the message to be signed (actually, it signs a universal hash function applied to each prefix, where the universal hash function is added to the verification key). This intuitively explains the loss in the leakage rate from the signing algorithm and the loss in the signature leakage. The key generation of $\mathcal{T}_2'$ is almost identical to that of $\mathcal{T}_2$, except that $\mathcal{T}_2'$ has the additional universal hash function in its verification key, but the private randomness used by the key generation of $\mathcal{T}_2'$ is identical to that of $\mathcal{T}_2$ (and the secret keys in both schemes are identical). Moreover, the update algorithm of $\mathcal{T}_2'$ is identical to that of $\mathcal{T}_2$. This explains why the rest of the parameters ρ_G, ρ_U, ρ_M are left unchanged.

Proof of Theorem 8.3. First, using the collision resistant hash family $\mathcal{H}$, one can assume without loss of generality that the message space of $\mathcal{T}_2$ is contained in $\{0,1\}^{\alpha(k)}$, by first hashing the message

[17] Actually, the transformation of [BK10] converts any signature scheme that is a priori unforgeable under *static* message attacks into one which is existentially unforgeable under adaptive chosen message attacks. But we will only apply it to schemes which are a priori unforgeable under adaptive chosen message attacks.

and then signing it. Then, Corollary 8.4 immediately implies that there exists a signature scheme $\mathcal{T}_2'$ that is existentially unforgeable under adaptive chosen message attacks in the CML model with leakage rate $(\rho_G, \rho_U, \frac{\rho_S}{\alpha(k)}, \rho_M)$ and signature leakage $\ell \cdot \alpha(k)$, as desired. $\square$

Finally, we combine Theorem 8.2 and Theorem 8.3, to obtain the main result of this subsection.

Theorem 8.5. *Assume the existence of: (1) a semantically secure public-key encryption scheme in the* CML *model, with leakage rate (ρ_G, ρ_U, ρ_M); (2) a family of* LTDF *with oblivious sampling with lossy parameter $\ell/2$; (3) a non-interactive argument system with proofs of length $\ell/2$; (4) a collision-resistant hash functions mapping strings of length k to strings of length $\alpha(k)$. Then there exists a signature scheme that is existentially unforgeable under adaptive chosen message attacks in the* CML *model, with leakage rate $(\rho_G, \rho_U, \rho_S, \rho_M)$, where $\rho_S = \frac{\rho_M}{\alpha(k)}$, and signature leakage $\ell \cdot \alpha(k)$.*

References

[ADN$^+$10] Joel Alwen, Yevgeniy Dodis, Moni Naor, Gil Segev, Shabsi Walfish, and Daniel Wichs. Public-key encryption in the bounded-retrieval model. To Appear in Eurocrypt 2010, 2010.

[ADW09] Joël Alwen, Yevgeniy Dodis, and Daniel Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In Halevi [Hal09], pages 36–54.

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 41–55. Springer, 2004.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103–112. ACM, 1988.

[BFO08] Alexandra Boldyreva, Serge Fehr, and Adam O'Neill. On notions of security for deterministic encryption, and efficient constructions without random oracles. In David Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 335–359. Springer, 2008.

[BG08] Boaz Barak and Oded Goldreich. Universal arguments and their applications. *SIAM J. Comput.*, 38(5):1661–1694, 2008.

[BK10] Zvika Brakerski and Yael Tauman Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. Cryptology ePrint Archive, Report 2010/086, 2010. `http://eprint.iacr.org/`.

[Boy99] Victor Boyko. On the security properties of oaep as an all-or-nothing transform. In *CRYPTO*, pages 503–518, 1999.

[BSMP91] Manuel Blum, Alfredo De Santis, Silvio Micali, and Giuseppe Persiano. Noninteractive zero-knowledge. *SIAM J. Comput.*, 20(6):1084–1118, 1991.

[CDH$^+$00] Ran Canetti, Yevgeniy Dodis, Shai Halevi, Eyal Kushilevitz, and Amit Sahai. Exposure-resilient functions and all-or-nothing transforms. In *EUROCRYPT*, pages 453–469, 2000.

[CLW06] Giovanni Di Crescenzo, Richard J. Lipton, and Shabsi Walfish. Perfectly secure password protocols in the bounded retrieval model. In *TCC*, pages 225–244, 2006.

[DHLAW10] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana Lopez-Alt, and Daniel Wichs. Cryptography against continuous memory attacks. Manuscript, 2010.

[DKL09] Yevgeniy Dodis, Yael Tauman Kalai, and Shachar Lovett. On cryptography with auxiliary input. In Michael Mitzenmacher, editor, *STOC*, pages 621–630. ACM, 2009.

[DOPS04] Yevgeniy Dodis, Shien Jin Ong, Manoj Prabhakaran, and Amit Sahai. On the (im)possibility of cryptography with imperfect randomness. In *FOCS*, pages 196–205, 2004.

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302. IEEE Computer Society, 2008.

[DS05] Yevgeniy Dodis and Adam Smith. Correcting errors without leaking partial information. In Gabow and Fagin [GF05], pages 654–663.

[FGK$^+$09] David Mandell Freeman, Oded Goldreich, Eike Kiltz, Alon Rosen, and Gil Segev. More constructions of lossy and correlation-secure trapdoor functions. Cryptology ePrint Archive, Report 2009/590, 2009. `http://eprint.iacr.org/`.

[FKPR10] Sebastian Faust, Eike Kiltz, Krzysztof Pietrzak, and Guy N. Rothblum. Leakage-resilient signatures. In *TCC*, pages 343–360, 2010.

[FLS90] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *FOCS*, volume I, pages 308–317. IEEE, 1990.

[FRR$^+$10] Sebastian Faust, Tal Rabin, Leonid Reyzin, Eran Tromer, and Vinod Vaikuntanathan. Protecting against leakage: the computationally bounded and noisy cases. To Appear in Eurocrypt 2010, 2010.

[Gam84] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18, 1984.

[GF05] Harold N. Gabow and Ronald Fagin, editors. *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*. ACM, 2005.

[GR10] Shafi Goldwasser and Guy Rothblum. How to play mental solitaire under continuous side-channels: A completeness theorem using secure hardware. Manuscript, 2010.

[Hal09] Shai Halevi, editor. *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*. Springer, 2009.

[HSH$^+$08] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: Cold boot attacks on encryption keys. In *USENIX Security Symposium*, pages 45–60, 2008.

[IPSW06] Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and David Wagner. Private circuits ii: Keeping secrets in tamperable circuits. In *EUROCRYPT*, pages 308–327, 2006.

[ISW03] Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *CRYPTO*, pages 463–481, 2003.

[JV10] Ali Juma and Yevgeniy Vahlis. Leakage-resilient key proxies. Manuscript, 2010.

[Kil92] Joe Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *STOC*, pages 723–732. ACM, 1992.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *CRYPTO*, pages 388–397, 1999.

[Koc96] Paul C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In *CRYPTO*, pages 104–113, 1996.

[KV09] Jonathan Katz and Vinod Vaikuntanathan. Signature schemes with bounded leakage resilience. In Mitsuru Matsui, editor, *ASIACRYPT*, volume 5912 of *Lecture Notes in Computer Science*, pages 703–720. Springer, 2009.

[Mic00] Silvio Micali. Computationally sound proofs. *SIAM J. Comput.*, 30(4):1253–1298, 2000.

[MR04] Silvio Micali and Leonid Reyzin. Physically observable cryptography (extended abstract). In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 278–296. Springer, 2004.

[NS09] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. In Halevi [Hal09], pages 18–35.

[OST06] Dag Arne Osvik, Adi Shamir, and Eran Tromer. Cache attacks and countermeasures: The case of aes. In *CT-RSA*, pages 1–20, 2006.

[Pie09] Krzysztof Pietrzak. A leakage-resilient mode of operation. In *EUROCRYPT*, pages 462–482, 2009.

[PSP$^+$08] Christophe Petit, François-Xavier Standaert, Olivier Pereira, Tal Malkin, and Moti Yung. A block cipher based pseudo random number generator secure against side-channel key recovery. In *ASIACCS*, pages 56–65, 2008.

[PW08] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In Richard E. Ladner and Cynthia Dwork, editors, *STOC*, pages 187–196. ACM, 2008.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Gabow and Fagin [GF05], pages 84–93.

[Riv97] Ronald L. Rivest. All-or-nothing encryption and the package transform. In *FSE*, pages 210–218, 1997.

[Sah99] Amit Sahai. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS*, pages 543–553, 1999.

[Val08] Paul Valiant. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency. In *TCC*, pages 1–18, 2008.

A A Linear Algebraic Tool

We prove the following lemma whose proof is a straightforward implementation of the analysis in [BFO08, Lemma 7.1].

Lemma A.1. *Let $m, \ell, d \in \mathbb{N}$, $m \geq \ell \geq 2d$ and let q be a prime. Let $\mathbf{X} \xleftarrow{\$} \mathbb{Z}_q^{m\times\ell}$, let $\mathbf{T} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_q^{\ell\times d})$ and let $\mathbf{Y} \xleftarrow{\$} \mathbb{Z}_q^{m\times d}$. Let $f : \mathbb{Z}_q^{m\times 2} \to W$ be some function. Then,*

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{Y}))\big) \leq \epsilon,$$

as long as

$$|W| \leq 4\cdot(1-1/q)\cdot q^{\ell-(2d-1)}\cdot\epsilon^2.$$

Proof. The proof is an adaptation of the "simple case" in the proof of [BFO08, Lemma 7.1]. The first part of our proof, up to and including Eq. (5), is taken directly from there (with the required changes of notation), the remainder of the proof is a sequence of straightforward derivations.

Define, for random variables X, Y supported on a domain W,

$$\begin{aligned} \mathrm{Col}(X) &\doteq \sum_{w\in W}\Pr[X=w]^2 \\ D(X,Y) &\doteq \sum_{w\in W}\left(\Pr[X=w]-\Pr[Y=w]\right)^2 . \end{aligned}$$

It holds that

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{Y}))\big) \leq \frac{1}{2}\sqrt{|W|\cdot\mathbb{E}_{\mathbf{X}}[D(f(\mathbf{X}\cdot\mathbf{T}), f(\mathbf{Y}))]}\ . \qquad (5)$$

Let us focus, therefore, on $\mathbb{E}_{\mathbf{X}}[D(f(\mathbf{X}\cdot\mathbf{T}), f(\mathbf{Y}))]$, recalling that for all $\mathbf{T}\in\mathrm{Rk}_d(\mathbb{Z}_q^{\ell\times d})$ it holds that $\mathbf{X}\cdot\mathbf{T}$ is uniformly distributed in $\mathbb{Z}_q^{m\times d}$ (i.e. identically distributed to $\mathbf{Y}$).

$$\begin{aligned} \mathbb{E}_{\mathbf{X}}[D(f(\mathbf{X}\cdot\mathbf{T}), f(\mathbf{Y}))] &= \mathbb{E}_{\mathbf{X}}\sum_{w\in W}\big(\Pr_{\mathbf{T}}[f(\mathbf{X}\cdot\mathbf{T})=w]-\Pr_{\mathbf{Y}}[f(\mathbf{Y})=w]\big)^2 \\ &= \mathbb{E}_{\mathbf{X}}\sum_{w\in W}\big(\mathbb{E}_{\mathbf{T}}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T})=w}]-\mathbb{E}_{\mathbf{Y}}[\mathbb{1}_{f(\mathbf{Y})=w}]\big)^2 \\ &= \mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}\big[\sum_{w\in W}\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)=w}\big] - 2\cdot\mathbb{E}_{\mathbf{X},\mathbf{T},\mathbf{Y}}\big[\sum_{w\in W}\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T})=f(\mathbf{Y})=w}\big] \\ &\qquad +\mathbb{E}_{\mathbf{X},\mathbf{Y}_1,\mathbf{Y}_2}\big[\sum_{w\in W}\mathbb{1}_{f(\mathbf{Y}_1)=f(\mathbf{Y}_2)=w}\big] \\ &= \mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}] - 2\cdot\mathbb{E}_{\mathbf{X},\mathbf{T},\mathbf{Y}}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T})=f(\mathbf{Y})}] + \mathbb{E}_{\mathbf{Y}_1,\mathbf{Y}_2}[\mathbb{1}_{f(\mathbf{Y}_1)=f(\mathbf{Y}_2)}] \\ &= \mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}] - \mathrm{Col}(f(\mathbf{Y}))\ . \end{aligned}$$

We are left with analyzing $\mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}]$. Denote by F the event where the matrix $[\mathbf{T}_1\|\mathbf{T}_2]$ is of full rank (and by $\bar{F}$ the complementary event). Note that for all $\mathbf{T}_1, \mathbf{T}_2$ for which F happens, it holds that $\mathbf{X}\cdot\mathbf{T}_1$ and $\mathbf{X}\cdot\mathbf{T}_2$ are uniform and independent. Therefore

$$\mathbb{E}_{\mathbf{X},\mathbf{T}_1,\mathbf{T}_2}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}] \leq \Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}] + \mathbb{E}_{\mathbf{T}_1,\mathbf{T}_2|F}\big[\mathbb{E}_{\mathbf{X}}[\mathbb{1}_{f(\mathbf{X}\cdot\mathbf{T}_1)=f(\mathbf{X}\cdot\mathbf{T}_2)}]\big] = \Pr[\bar{F}] + \mathrm{Col}(f(\mathbf{Y}))\ .$$

To bound $\Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}]$, we use Lemma 4.1 to get

$$\Pr_{\mathbf{T}_1,\mathbf{T}_2}[\bar{F}] = \Pr\big[[\mathbf{T}_1 \| \mathbf{T}_2] \text{ is not full rank}\big] \leq \Pr_{\mathbf{R} \xleftarrow{\$} \mathbb{Z}_q^{\ell \times 2d}}[\mathbf{R} \text{ is not full rank}] \leq \frac{q^{2d-\ell}}{q-1} \ .$$

Putting it all together, we get that

$$\mathsf{dist}\big((\mathbf{X}, f(\mathbf{X}\cdot\mathbf{T})),\ (\mathbf{X}, f(\mathbf{Y}))\big) \leq \sqrt{\frac{|W|}{4\cdot(1-1/q)\cdot q^{\ell-(2d-1)}}}\ ,$$

and the result follows. □