# Fully Leakage-Resilient Signatures

Elette Boyle[*] Gil Segev[†] Daniel Wichs[‡]

September 14, 2010

**Abstract**

A signature scheme is *fully leakage resilient* (Katz and Vaikuntanathan, ASIACRYPT '09) if it is existentially unforgeable under an adaptive chosen-message attack even in a setting where an adversary may obtain bounded (yet arbitrary) leakage information on *all intermediate values that are used throughout the lifetime of the system*. This is a strong and meaningful notion of security that captures a significantly wide range of side-channel attacks.

One of the main challenges in constructing fully leakage-resilient signature schemes is dealing with leakage that may depend on the random bits used by the signing algorithm, and constructions of such schemes are known only in the random-oracle model. Moreover, even in the random-oracle model, known schemes are only resilient to leakage of less than half the length of their signing key.

In this paper we construct the first fully leakage-resilient signature schemes without random oracles. We present a scheme that is resilient to any leakage of length $(1 - o(1))L$ bits, where $L$ is the length of the signing key. Our approach relies on generic cryptographic primitives, and at the same time admits rather efficient instantiations based on specific number-theoretic assumptions. In addition, we show that our approach extends to the continual-leakage model, recently introduced by Dodis, Haralambiev, Lopez-Alt and Wichs (FOCS '10), and by Brakerski, Tauman Kalai, Katz and Vaikuntanathan (FOCS '10). In this model the signing key is allowed to be refreshed, while its corresponding verification key remains fixed, and the amount of leakage is assumed to be bounded only in between any two successive key refreshes.

[*]Department of Mathematics, Massachusetts Institute of Technology, Cambridge MA 02139, USA. Email: `eboyle@mit.edu`. Research supported by the US National Defense Science and Engineering Graduate Fellowship. This work was partially completed while visiting the Weizmann Institute of Science.

[†]Microsoft Research, Mountain View, CA 94043, USA. Email: `gil.segev@microsoft.com`. Most of this work was completed while the author was a Ph.D. student at the Weizmann Institute of Science, and supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities.

[‡]Department of Computer Science, New York University, New York NY 10012, USA. This work was partially completed while visiting the Weizmann Institute of Science.

## 1 Introduction

One of the main goals of research in the foundations of cryptography is designing systems that withstand adversarial behavior. Given a cryptographic task, such as public-key encryption, one must formalize an attack model specifying a class of adversaries, and define a notion of security capturing what it means to break the system. Within such a framework, it is then possible to rigorously analyze the security of cryptographic systems.

Starting with the seminal work of Goldwasser and Micali [GM84], various and increasingly strong attack models and notions of security have been proposed. Over the years, however, theoreticians and practitioners began to notice that a large class of realistic attacks, called *side-channel attacks*, are not captured by the existing models. In such attacks, the adversary may learn some additional information about the internal secret state of a system, by measuring various properties resulting from specific *physical* implementations (e.g., timing information, detection of internal faults, electromagnetic radiation, power consumption etc. [BS97, BDL97, Koc96, KJJ99]). As a result, it has become an important research agenda to extend the standard models to capture such side-channel attacks, and to design cryptographic systems whose security guarantees can be rigorously analyzed and clearly stated in these stronger models. Our work focuses on the *bounded-leakage* and *continual-leakage* models described below (several other models are described in Section 1.2).

**The bounded-leakage and continual-leakage models.** A prime example for such a modeling is the *bounded-leakage model* formalized by Akavia, Goldwasser, and Vaikuntanathan [AGV09]. This model is motivated by the "cold-boot attacks" of Halderman et al. [HSH$^+$08], which exploit DRAM remanence effects to retrieve information on cryptographic keys stored in memory. In the bounded-leakage model, a cryptographic scheme is secure if it retains its standard notion of security even when bounded (but yet arbitrary) information on the secret key is leaked. For instance, in the setting of public-key encryption, a scheme must remain semantically secure even when adversarially chosen functions of the secret key are leaked in an adaptive fashion, as long as the total amount of leakage is bounded. A great deal of research has taken part since then to devise various cryptographic primitives in this model, such as public-key and identity-based encryption schemes, signature schemes, and more (see [NS09, KV09, ADW09, ADN$^+$10, LPS10, BG10]).

A drawback of the bounded-leakage model is that the overall amount of leakage is bounded (in particular, less than the length of the secret key), and this is clearly an inherent limitation as long as the secret key remains fixed throughout the lifetime of the system. Recently, Dodis et al. [DHL$^+$10a] and Brakerski et al. [BTK$^+$10] suggested the *continual-leakage model*, in which secret keys are allowed to be refreshed, while their corresponding public keys remain fixed. In this model, security must hold even against an adversary that can continually leak arbitrary information about the current secret key of the scheme, as long as the amount of leaked information is bounded in between any two successive key refreshes.

We note that in both the bounded-leakage model and the continual-leakage model the adversary may be able to learn partial, but yet *arbitrary*, information on the *entire* secret key. This is in contrast with other models, where either the leakage is assumed to be of "low complexity" (such as $AC^0$ circuits) [ISW03, FRR$^+$10], or certain secret values are assumed to be leak-free [MR04, DP08, Pie09, FKP$^+$10, GR10, JV10].

**Leakage-resilient signature schemes.** In this paper we study the security of signature schemes in the bounded-leakage and continual-leakage models. Signature schemes in the bounded-leakage model were proposed by Alwen, Dodis, and Wichs [ADW09] and by Katz and Vaikuntanathan [KV09], who focused mainly on leakage from the signing key of the scheme. Specifically, a signature scheme is leakage-resilient in the bounded-leakage model if it is existentially unforgeable against an adaptive chosen-message attack [GMR88] even when adversarially chosen functions of the signing key are leaked in an adaptive fashion. Signature schemes satisfying this notion of security were constructed both based on generic cryptographic primitives in the standard model [KV09] and based on the Fiat-Shamir transform [FS86] in the random-oracle model [KV09, ADW09].

Although this notion of security models various side-channel attacks, such as the cold-boot attacks [HSH$^+$08], it does not fully capture attacks which rely on leakage that occurs during the signing operation. Unlike in the setting of public-key encryption, where decryption algorithms are typically deterministic (and thus the only secret information is the decryption key), the secret state for a signature scheme consists of much more than just the signing key. Specifically, each invocation of the signing algorithm involves both the signing key and fresh secret randomness[1]. Thus, even if a scheme is secure against leakage from the signing key only, it may be vulnerable to attacks that rely on leakage of intermediate values from the signing process.

In order to capture leakage from the signing operation, Katz and Vaikuntanathan [KV09] put forward the more general notion of a *fully leakage-resilient* signature scheme in the bounded-leakage model. This notion requires a signature scheme to remain existentially unforgeable under an adaptive chosen-message attack even when the adversary obtains bounded leakage information on *all intermediate values* used by the signer throughout the lifetime of the system (this can be naturally extended to the continual-leakage model [DHL$^+$10a, BTK$^+$10]). This, indeed, seems to capture a more realistic class of side-channel attacks that take place *during the execution* of a system, such as attacks that rely on timing information, power consumption patterns, and more.

Currently, however, the known constructions of fully leakage-resilient signature schemes either provide only one-time security[2] [KV09], or are proven secure in the random-oracle model [ADW09, BTK$^+$10, DHL$^+$10a, KV09]. Moreover, even in the random-oracle model, known schemes are either resilient to leakage of at most half the length of the signing key [ADW09, DHL$^+$10a, KV09], or require refreshing the signing key after every few invocation of the signing algorithm even when no leakage occurs [BTK$^+$10] (and this is required also in the bounded-leakage model where refreshing is not part of the typical functionality).

## 1.1 Our Contributions

We construct the first fully leakage-resilient signature schemes without random oracles. We first present a scheme in the bounded-leakage model that is resilient to any leakage of $(1 - o(1))L$ bits, where $L$ is the bit-length of the signing key. Our scheme is based on generic cryptographic primitives, and is inspired by the approach of Katz and Vaikuntanathan [KV09] (although their scheme is resilient to leakage from the signing key only). Moreover, we show that our construction can be instantiated based on specific number-theoretic assumptions to yield a rather efficient scheme. That is, we demonstrate that it is not only of theoretical interest (due to using a certain form of an argument system for general NP languages), but may also be of practical interest.

We then extend our approach to the continual-leakage model by relying on any *continual leakage-resilient one-way relation*, a primitive recently introduced by Dodis, Haralambiev, Lopez-Alt and Wichs [DHL$^+$10a]. The amount of leakage that can be tolerated between any two successive

[1]Without leakage, the signing algorithm of any signature scheme can be made deterministic by using, as its random coins, the output of a pseudorandom function (PRF) applied to the message, where the seed of the PRF is made part of the secret key. However, in the setting of key leakage, this transformation may no longer be secure since the seed to the PRF can also leak.

[2]Katz and Vaikuntanathan in fact generalized their basic one-time scheme to a $t$-time scheme (where $t$ is any pre-determined parameter) that tolerates any leakage of $\Theta(L/t)$ bits.

refreshes of the signing key depends on the leakage resilience of the underlying one-way relation[3]. In turn, instantiating our scheme with existing constructions of such one-way relations yields schemes that are resilient to any leakage of length $(1/2 - o(1))L$ bits based on the Linear assumption, and of length $(1 - o(1))L$ bits based on the Symmetric External Diffie-Hellman assumption [DHL+10a, BTK+10].

## 1.2 Related Work

Various constructions of leakage-resilient signature schemes have been proposed so far. In this section we describe the different leakage models and discuss the security guarantees that are satisfied by these constructions. In what follows we denote by $L$ the bit-length of the signing key.

**The limited-complexity leakage model.** Some of the initial works in leakage-resilient cryptography concentrated on specific and limited forms of leakage. Ishai et al. [ISW03] showed how to securely implement any efficiently computable function in the presence of an adversary who can probe the values of a constant number of wires. Faust et al. [FRR+10] demonstrated a related construction allowing leakage functions with bounded output, belonging to a low complexity class, such as $AC^0$. Unfortunately, to accommodate such generality in the underlying function, constructions of this type so far handled only weak forms of leakage, which seem rather far from capturing many known side-channel attacks.

**The "only computation leaks" model.** Micali and Reyzin [MR04] introduced a model in which the complexity of leakage functions is unrestricted, the overall amount of leakage is unbounded, but leakage is assumed to only occur on values currently accessed during a computation. Explicitly, in each step of computation, the adversary can adaptively select a polynomial-time leakage function $f$ with bounded output to be applied to current active values. Values are active in a computation step if they are accessed by the cryptographic system; all other values stored in memory but not accessed during the time-step are assumed to be leak-free for the step. This model lends itself to techniques that split a computation into two halves, maintaining only one half as active in any given time-step, as proposed by Dziembowski and Pietrzak [DP08, Pie09] within the construction of leakage-resilient stream ciphers.

Faust et al. [FKP+10] constructed a stateful, tree-based signature scheme in the "only computation leaks" model based on any 3-time secure signature scheme. Their construction can handle $\ell/3$ bits of leakage per signature, where the underlying scheme is resilient to $\ell$ bits of leakage overall. Implementing this with currently known constructions for the underlying scheme yields leakage resilience approaching $L/36$ bits, based on one-way functions.

Goldwasser and Rothblum [GR10] recently presented a way to compile any cryptographic algorithm into one that resists leakage bounded by a constant fraction of the secret key size in the "only computation leaks" model, based on the DDH assumption and the existence of a simple secure hardware component. Using different techniques, Juma and Vahlis [JV10] showed how to encapsulate a secret key and compute on it in a leakage-resilient fashion within the "only computation leaks" model using fully homomorphic encryption and leak-free hardware tokens.

[3]As in [DHL+10a, BTK+10], our schemes are also resilient to leakage of logarithmic length from the random bits used by the refreshing algorithm. We note that this property is directly inherited from the underlying one-way relation, and our approach does not introduce any further limitations. Specifically, instantiating our scheme with any continual leakage-resilient one-way relation that can tolerate leakage of length $\lambda$ bits from the refreshing algorithm results in a signature scheme with the same guarantee.

The "only computation leaks" paradigm is powerful in the sense that it allows any efficiently computable leakage functions, and a large amount of overall leakage. However, it rests on the assumption that values not immediately being used in computation are completely leak-free, which is not always valid. Specifically, this model does not capture known attacks in which inactive values in memory are still vulnerable, such as the cold-boot attacks [HSH$^+$08], or measurements which can detect the physical processes used to maintain values in memory.

**The bounded-leakage model.** In the bounded-leakage model, as discussed above, signature schemes resilient to leakage depending only on the signing key have been constructed in the random-oracle model [ADW09] and in the standard model (i.e., without random oracles) [KV09], handling leakage of up to $(1 - o(1))L$ bits (see also the work of Dodis et al. [DHL$^+$10b] that focuses on efficient instantiations of such schemes).

Many-time signature schemes resistant to leakage from the entire secret state have been constructed only in the random-oracle model. Alwen et al. [ADW09] and Katz and Vaikuntanathan [KV09] presented a scheme using the Fiat-Shamir transform [FS86] with any second-preimage resistant function. Both works concurrently proposed the same scheme, but leakage of randomness was only analyzed in the latter [KV09]. Due to the use of the Fiat-Shamir transform, the construction is only resilient to leakage approaching length $L/2$. A more efficient version, allowing the same amount of leakage, was given by Dodis et al. [DHL$^+$10b].

The only existing constructions of fully leakage-resistant signatures in the standard model are a pair of one-time signature schemes due to Katz and Vaikuntanathan [KV09]. Their first scheme is based on the existence of any one-way function, and is resilient to any leakage of length less than $L/4$ bits. The second is based on specific number-theoretic assumptions, and is resilient to any leakage of length less than $L/2$ bits. Both schemes extend to $t$-time schemes with leakage less than $\Theta(L/t)$. Since these schemes have a deterministic signing algorithm, the secret state of the signer consists only of the secret key, and thus full leakage-resilience simply reduces to the weaker notion of resilience against key leakage.

**The continual-leakage model.** Notions of leakage-resilient signatures were naturally extended to the continual-leakage setting by Dodis et al. [DHL$^+$10a] and Brakerski et al. [BTK$^+$10], as described above. Dodis et al. [DHL$^+$10a] presented two signature schemes in the continual-leakage model by using their abstracted concept of a continual leakage-resilient one-way relation together with known schemes from the bounded-leakage model. The first scheme is a modified version of the scheme of Katz and Vaikuntanathan [KV09], and is resilient to leakage only from the signing key. Their second scheme uses the Fiat-Shamir transform in the random-oracle model, and allows up to $L/2$ bits of leakage in each round from the entire current secret state of the signer.

Brakerski et al. [BTK$^+$10] also presented two signature schemes in this model. The first scheme is based on the one of Katz and Vaikuntanathan [KV09] as in [DHL$^+$10a], and is resilient to leakage only from the signing key. The second scheme is in the random-oracle model, and is secure against leakage from the entire secret state of the signer. The amount of leakage that can be tolerated is $(1/2 - o(1))L$ bits per key update based on the Linear assumption, and $(1 - o(1))L$ bits based on the somewhat non-standard SXDH assumption. In their scheme, each signature leaks information about the secret key, regardless of whether the adversary makes leakage queries. This continual leakage of information through signature queries means the signing key must be refreshed after every few invocation of the signing algorithm. Thus, the construction does not yield a scheme in the bounded-leakage model, where refreshing is not part of the typical functionality of a signature scheme, and is less efficient in the continual-leakage model.

## 1.3 Overview of Our Approach

In this section we present an overview of our approach for constructing fully leakage-resilient signature schemes. We focus here on our construction in the bounded-leakage model, as it already emphasizes the main ideas underlying our approach, and we refer the reader to Section 7 for an overview of our construction in the continual-leakage model. We begin by describing more clearly the notion of a fully leakage-resilient signature scheme in the bounded-leakage model. Then, we briefly describe the leakage-resilient signature scheme of Katz and Vaikuntanathan [KV09], which serves as our starting point, and explain the main challenges in constructing *fully* leakage-resilient signature schemes. The main part of this overview then focuses on our construction.

**Modeling fully leakage-resilient signature schemes.** A signature scheme is fully leakage-resilient in the bounded-leakage model if it is existentially unforgeable against an adversary that can obtain both signatures on any message of her choice, and bounded leakage information on all intermediate values used by the signer throughout the lifetime of the system.

This is formalized by considering an experiment that involves a signer and an adversary. First, the signer invokes the key-generation algorithm and obtains a verification key $vk$ and a signing key $sk$. At this point, a set $\mathsf{state}$ is initialized to contain the random coins that were used by the key-generation algorithm, and the adversary is given the verification key $vk$. Then, the adversary can adaptively submit two types of queries: signing queries, and leakage queries. A signing query consists of a message $m$, and is answered by invoking the signing algorithm with the signing key and the message. Following each such query, the random coins that were used by the signing algorithm are added to the set $\mathsf{state}$. A leakage query consists of a leakage function $f$, and is answered by applying $f$ to the set $\mathsf{state}$. The leakage functions have to be efficiently computable, and the sum of their output lengths has to be upper bounded by a predetermined parameter $\lambda$. The adversary is successful if she outputs a pair $(m^*, \sigma^*)$, where $m^*$ is a message with which she did not issue a signing query, and $\sigma^*$ is a valid signature on $m^*$ with respect to $vk$. We refer the reader to Section 3 for a formal definition.

**The Katz-Vaikuntanathan scheme.** The Katz-Vaikuntanathan signature scheme [KV09] relies on a second-preimage resistant function $\mathsf{F} : \{0,1\}^n \to \{0,1\}^{n^\epsilon}$, a CPA-secure public-key encryption scheme, and an unbounded simulation-sound NIZK proof system[4]. The signing key is a random $x \in \{0,1\}^n$, and the verification key is a triplet $(y = \mathsf{F}(x), pk, \mathsf{crs})$, where $pk$ is a public key for the encryption scheme, and $\mathsf{crs}$ is a common-reference string for the proof system. A signature on a message $m$ consists of a ciphertext $c$ which is an encryption of $m||x$ using $pk$, and a proof that the ciphertext $c$ is indeed an encryption of $m||x'$, for some $x' \in \{0,1\}^n$ such that $\mathsf{F}(x') = y$.[5]

This scheme is leakage resilient in the bounded-leakage model. That is, it satisfies the weaker variant of the above notion of security, where the leakage is allowed to depend on the signing key only. The security of the scheme is based on three main properties:

1. A typical verification key has many possible secret keys. Specifically, the set $\mathsf{F}^{-1}(y)$ is of size roughly $2^{n-n^\epsilon}$.

[4]We note that when $\mathsf{F}$ is only assumed to be a one-way function, the scheme may not always be resilient to leakage, but it is nevertheless existentially unforgeable under an adaptive chosen-message attack. In this case the scheme can be viewed as a variant of the Bellare-Goldwasser signature scheme [BG89].

[5]In fact, Katz and Vaikuntanathan show that it is possible to encrypt only $x$ (instead of $m||x$), and include $m$ as a label in the statement that is proved using the NIZK proof system. However, for making this informal description more intuitive, we consider here an encryption of both $m$ and $x$.

2. Signatures can be produced in a way that reveals no information on the signing key, and is computationally indistinguishable from signatures produced by the actual scheme. This follows from the semantic security of the encryption scheme and from the zero knowledge of the proof system. Specifically, such a signature on a message $m$ can be produced by encrypting $m||0^n$, and then using the simulator of the proof system to generate a proof for a false statement.

3. Given the decryption key corresponding to $pk$, any valid forgery produced by the adversary can be used to extract a preimage $x'$ of $y$. This follows from the unbounded simulation soundness of the proof system, even when signature queries are answered using the simulator of the proof system.

These three properties are used to prove the security of the scheme as follows. The first and second properties imply that the view of the adversary does not completely determine the value $x$. This is because the signing queries of the adversary essentially reveal no information on the particular preimage $x$ of $y$ that is used in the signing key. In addition, any leakage of length $\lambda \leq n - n^{\epsilon} - \omega(\log n)$ bits on $x$ reduces its entropy by at most $\lambda$ (on average), and therefore there is simply not enough information for the adversary to learn $x$. Thus, with a noticeable probability, the value $x'$ extracted from the adversary's forgery (due to the third property) is different from $x$. Therefore, the probability that the adversary produces a valid forgery must be negligible due to the second-preimage resistance of $\mathsf{F}$.

**The main challenges.** The security proof of the Katz-Vaikuntanathan scheme relies on the argument that the output of all signing queries and any leakage of $\lambda$ bits from the signing key $x$ do not reduce its min-entropy by more than $\lambda$ on average[6]. However, when the leakage may depend also on the randomness used by the signing algorithm, this is no longer true, and in fact their scheme is insecure in general. The main problem is that it is no longer possible to rely on the semantic security of the encryption scheme, or on the zero knowledge of the proof system, and these two primitives might actually be *completely* insecure. Consider, for example, an instantiation of their scheme with a CPA-secure encryption scheme defined as $\mathsf{Enc}_{pk}(m||x) = (\mathsf{Enc}'_{pk}(s), \mathsf{PRG}(s) \oplus (m||x))$, where $\mathsf{Enc}'$ is the encryption algorithm of any CPA-secure encryption scheme, and $\mathsf{PRG}$ is a pseudorandom generator that is applied on a random seed $s$. Leaking the seed $s$, whose length may be much shorter than $\lambda$ bits, completely reveals the signing key $x$. A similar instantiation for the proof system can be shown to have a similar effect when the leakage may depend on the randomness used by the prover[7].

A natural observation is that the above problem can be avoided by using an encryption scheme and a proof system with *statistical* guarantees. Specifically, using an encryption scheme which statistically hides the encrypted message, and a proof system (or an argument system) that statistically hides the witness. In such an almost "purely statistical" setting, the output of all signing queries and any leakage of $\lambda$ bits (even leakage that may depend on the randomness) reduce the min-entropy of the signing key by at most $\lambda$. However, this introduces the following conflict: although now the output of all signing queries do not reveal information on the signing key (due

[6]This is done by modifying the signing oracle such that it signs a message $m$ by encrypting $m||0^n$ (instead of $m||x$), and using the simulator of the proof system to generate a proof for a false statement.

[7]Note that even a leakage function with only one output bit can be easily used to distinguish an encryption of $m||x$ from an encryption of $m||0^n$, or to distinguish the prover of the proof system from the simulator of the proof system. Thus, technically speaking, it seems that at no point in time during the various experiments of the security proof it is possible to change the way signing queries are answered (this, in particular, rules out using the simulator of the zero-knowledge proof system).

to the statistical properties), also a valid forgery produced by the adversary does not reveal any information. In particular, it does not provide any information useful for extracting a preimage $x'$ of $y$.

**Our approach.** We show that it is possible to resolve the latter conflict. To describe the mains ideas underlying our approach, we find it useful to first consider a weaker notion of unforgeability that we refer to as *selective unforgeability under a chosen-message attack* (our actual scheme achieves the stronger notion discussed above). This notion differs from the one discussed above by requiring that the adversary specifies the message $m^*$, on which she plans to forge a signature, in advance before receiving the verification key. The adversary can still adaptively interact with the signing oracle and leakage oracle after receiving the verification key.

For satisfying this notion of unforgeability, we introduce the concept of an *all-lossy-but-one (ALBO) public-key encryption scheme*. This is a tag-based public-key encryption scheme where encryption under one specific tag $t^*$ (which is given as input to the key-generation algorithm) allows efficient recovery of the encrypted message using the secret key, while encryption under any other tag $t \neq t^*$ statistically hides the encrypted message. We call $t^*$ the *injective* tag, and any other tag a *lossy* tag[8]. The only computational requirement is that the public key hides the injective tag $t^*$. The first step of our approach is to replace the CPA-secure encryption scheme in the Katz-Vaikuntanathan scheme with an ALBO encryption scheme, where the signing algorithm on input a message $m$, uses $m$ as a tag for encrypting the signing key $x$. Whereas the default injective tag $t^*$ can be set to any fixed value, in the proof of security we set the injective tag to the adversary's specified forgery $m^*$. This guarantees that the ciphertext part of the signatures produced by the signing oracle do not reveal any information on $x$, yet we can still extract $x$ from the forged signature.

The second step of our approach is to replace the unbounded simulation-sound NIZK proof system with a *statistical non-interactive witness-indistinguishable (SNIWI) argument system.* On one hand we relax the unbounded simulation-sound zero knowledge property to witness indistinguishability[9], and on the other hand we require that different witnesses to the same statement result in statistically indistinguishable proofs. The SNIWI argument system is used by our signing algorithm for proving that the ciphertext $c = \mathsf{Enc}_{pk}^{m}(x)$ produced by the ALBO encryption scheme is indeed an encryption under the tag $m$ of some $x' \in \{0,1\}^n$ for which $\mathsf{F}(x') = y$. The statistical property of the argument system guarantees that also the proof part of the signatures produced by the signing oracle do not reveal any information on $x$ (other than the fact $y = \mathsf{F}(x)$, which is already known to the adversary from the verification key). In addition, the soundness of the argument system guarantees that a valid forgery produced by the adversary on $m^*$ indeed contains an encryption under the tag $m^*$ of some preimage $x'$ of $y$. Moreover, since the injective tag of the ALBO encryption scheme is set to $m^*$, this enables us to recover $x'$.

At this point we would like to argue that, with a noticeable probability, it holds that $x' \neq x$. Note that any leakage of length $\lambda \leq n - n^{\epsilon} - \omega(\log n)$ bits can reduce the min-entropy of $x$ by at most $\lambda$ (on average), and this clearly holds even when the leakage may depend on all intermediate values. The main point is that the statistical properties of the ALBO encryption scheme and the SNIWI argument system enable us to argue that signatures produced by the signing oracle do not reveal information on $x$, even conditioned on the leakage, whereas a valid forgery produced by the adversary enables to extract a preimage $x'$ of $y$. The argument itself is rather subtle, and we refer

[8]We note that our notion is the opposite of the notion of an all-but-one lossy trapdoor function, where there is one lossy tag and all the other tags are injective.

[9]As discussed above, it seems that we cannot allow ourselves to rely on a simulator for achieving zero knowledge.

the reader to Section 5 for more details.

**The actual scheme.** So far we described our approach as leading to the rather weak notion of selective unforgeability under a chosen-message attack. Our actual scheme is fully leakage-resilient according to the stronger notion that was discussed in the beginning of this section (i.e., where the adversary is allowed to adaptively choose $m^*$ after making all signing and leakage queries).

In the random oracle model (which we do not consider in this paper), our approach easily extends to satisfying this stronger notion. This is done by modifying the signing algorithm such that it signs $H(m)$ instead of $m$, where $H(\cdot)$ is the random oracle. The proof of security is essentially identical, and the only difference is that the injective tag $t^*$ of the ALBO encryption scheme is now chosen uniformly at random (since $m^*$ is not known in advance), and is dynamically embedded as the answer to a random oracle query. Therefore, with a non-negligible probably it holds that $H(m^*) = t^*$. In this case, it again holds that signatures produced by the signing oracle do not reveal information on $x$, whereas a valid forgery that is produced by the adversary enables to extract a preimage $x'$ of $y$, and the security follows as before.

For our actual scheme (i.e., without a random oracle) we follow the approach of Boneh and Boyen [BB04] for transforming selectively-secure identity-based encryption schemes into fully secure ones using an admissible hash function (see Section 2.6) to replace the random oracle when signing $H(m)$ instead of $m$. The main idea of an admissible hash function is that it allows the reduction in the proof of security to secretly partition the message space into two subsets, which we will label as red and blue, such that there is a noticeable probably that all of the messages in the adversary's signing queries will be in the blue set, but the forgery will be on a message in the red set. In our setting, this translates to the properties that signatures on blue messages do not reveal information on $x$, whereas signatures on red messages enable to extract a preimage $x'$ of $y$. For implementing this approach we introduce the notion of an *$\mathcal{R}$-lossy public-key encryption scheme.* This is a generalization of an ALBO encryption scheme where the set of possible tags is partitioned into injective tags and lossy tags (in particular, there may be more than one injective tag).

Finally, we note that there are generic methods that can be applied, but we do not follow them due to their security loss or inefficiency. First, any scheme that is selectively unforgeable satisfies also the stronger notion with a $2^{\ell(n)}$ security loss by simply guessing the message $m^*$, where $\ell(n)$ is the message length. Second, any scheme that is selectively unforgeable can be transformed to a scheme satisfying the stronger notion by signing each prefix of the message [HW09, BT10] and using a Chameleon hash function [KR00]. Although this transformation applies also for fully leakage-resilient signature schemes, signing each prefix of the message significantly hurts the efficiency.

## 1.4 Paper Organization

In Section 2 we introduce some preliminaries and notation. Section 3 contains a discussion on the models of leakage resilience considered in this paper. In Section 4 we define and construct $\mathcal{R}$-lossy public-key encryption schemes, a tool used in our constructions. Section 5 contains the construction and security proof of our signature scheme in the bounded-leakage model. In Section 6 we present a specific instantiation of our scheme based on the Linear assumption. In Section 7 we extend our scheme to the continual-leakage model. Finally, in Section 8 we discuss several concluding remarks and open problems.

## 2 Preliminaries

In this section we present some basic notions, definitions, and tools that are used in our constructions.

### 2.1 Statistical Distance, Min-Entropy, and Average Min-Entropy

The *statistical distance* between two random variables $X$ and $Y$ over a finite domain $\Omega$ is defined as $\mathrm{SD}(X, Y) = \frac{1}{2}\sum_{\omega\in\Omega} |\Pr[X = \omega] - \Pr[Y = \omega]|$. We say that two variables are *$\epsilon$-close*, and write $X \approx_\epsilon Y$, if their statistical distance is at most $\epsilon$. The *min-entropy* of a random variable $X$ is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X = x])$. Dodis et al. [DOR+08] formalized the notion of *average min-entropy* that captures the remaining unpredictability of a random variable $X$ conditioned on the value of a random variable $Y$, which is defined as follows:

$$\widetilde{\mathrm{H}}_\infty(X|Y) = -\log\left(E_{y\leftarrow Y}\left[2^{-\mathrm{H}_\infty(X|Y=y)}\right]\right) .$$

The following bound on average min-entropy was proved in [DOR+08]:

**Lemma 2.1** ([DOR+08])**.** *For any random variables $X$, $Y$ and $Z$, if $Y$ has at most $2^\lambda$ possible values then*

$$\widetilde{\mathrm{H}}_\infty(X|Y, Z) \geq \widetilde{\mathrm{H}}_\infty(X|Z) - \lambda .$$

### 2.2 The DDH and $d$-Linear Assumptions

Let $\mathsf{GroupGen}$ be a probabilistic polynomial-time algorithm that takes as input a security parameter $1^n$, and outputs a triplet $(\mathbb{G}, p, g)$ where $\mathbb{G}$ is a group of order $p$ that is generated by $g \in \mathbb{G}$, and $p$ is an $n$-bit prime number.

**The decisional Diffie-Hellman assumption.** The decisional Diffie-Hellman (DDH) assumption is that the ensembles $\{(\mathbb{G}, g_1, g_2, g_1^r, g_2^r)\}_{n\in\mathbb{N}}$ and $\{(\mathbb{G}, g_1, g_2, g_1^{r_1}, g_2^{r_2})\}_{n\in\mathbb{N}}$ are computationally indistinguishable, where $(\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^n)$, and the elements $g_1, g_2 \in \mathbb{G}$ and $r, r_1, r_2 \in \mathbb{Z}_p$ are chosen independently and uniformly at random.

**The $d$-Linear assumption.** Boneh, Boyen, and Shacham [BBS04] introduced the Linear assumption, intended to take the place of DDH in groups where DDH is easy (specifically, in bilinear groups). They showed that the hardness of DDH implies that hardness of Linear, but at least in generic groups (see, for example, [JN03, Sho97]), Linear remains hard even if DDH is easy. The DDH and Linear assumptions naturally generalize to the family of $d$-Linear assumptions [Kil07, Sha07], where for every $d \geq 1$ the $d$-Linear assumption is that the ensembles

$$\left\{\left(\mathbb{G}, g_1, \ldots, g_d, g_{d+1}, g_1^{r_1}, \ldots, g_d^{r_d}, g_{d+1}^{\sum_{i=1}^d r_i}\right)\right\}_{n\in\mathbb{N}}$$

$$\left\{\left(\mathbb{G}, g_1, \ldots, g_d, g_{d+1}, g_1^{r_1}, \ldots, g_d^{r_d}, g_{d+1}^{r_{d+1}}\right)\right\}_{n\in\mathbb{N}} ,$$

are computationally indistinguishable, where $(\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^n)$, and the elements $g_1, \ldots, g_{d+1} \in \mathbb{G}$ and $r_1, \ldots, r_{d+1} \in \mathbb{Z}_p$ are chosen independently and uniformly at random.

Note that DDH is the 1-Linear assumption, and that Linear is the 2-Linear assumption. These assumptions are progressively weaker: the hardness of $d$-Linear implies the hardness of $(d + 1)$-Linear, but in generic groups $(d + 1)$-Linear remains hard even if $d$-Linear is easy.

### 2.3 Second-Preimage Resistance

A family of efficiently computable functions is a pair of polynomial-time algorithms $(\mathsf{KeyGen}, \mathsf{F})$, where $\mathsf{KeyGen}$ is a probabilistic algorithm that on input $1^n$ outputs a description $s \in \{0,1\}^*$ of a function $\mathsf{F}(s, \cdot) : \{0,1\}^{m(n)} \to \{0,1\}^{m'(n)}$. Such a family is *second-preimage resistant* (SPR) if given a randomly chosen input $x \in \{0,1\}^{m(n)}$ and a description of a randomly chosen function $s \leftarrow \mathsf{KeyGen}(1^n)$, it is computationally infeasible to find an input $x' \in \{0,1\}^{m(n)}$ such that $x' \neq x$ and $\mathsf{F}(s, x) = \mathsf{F}(s, x')$. This is a weakening of the notion of a family of universal one-way hash functions introduced by Naor and Yung [NY89], in which the input $x$ is allowed to be chosen in an adversarial manner (but still independently of the function description $s$).

**Definition 2.2** (Second-preimage resistance). *A family $\mathcal{F} = (\mathsf{KeyGen}, \mathsf{F})$ of efficiently computable functions is* second-preimage resistant *if for any probabilistic polynomial-time algorithm $\mathcal{A}$ is holds that*

$$\Pr\left[ \mathsf{F}_s(x') = \mathsf{F}_s(x) \wedge x' \neq x \;\middle|\; \begin{array}{c} s \leftarrow \mathsf{KeyGen}(1^n), x \leftarrow \{0,1\}^{m(n)} \\ x' \leftarrow \mathcal{A}(s, x) \end{array} \right] < \nu(n) \ ,$$

*for some negligible function $\nu(n)$, where the probability is taken over the choice of $x \leftarrow \{0,1\}^{m(n)}$ and over the internal randomness of $\mathsf{KeyGen}$ and $\mathcal{A}$.*

In addition, we say that $\mathcal{F} = (\mathsf{KeyGen}, \mathsf{F})$ is a family of *public-coin* second-preimage resistant functions, if it satisfies Definition 2.2 even when the algorithm $\mathcal{A}$ takes as input also the internal randomness that was used by $\mathsf{KeyGen}(1^n)$ for sampling the function. We refer the reader to [HR04] for more details on public-coin hash functions.

For any integer functions $m(n)$ and $m'(n)$ that are polynomially related, the existence of universal one-way hash functions (and therefore also of second-preimage resistant functions) with domain $\{0,1\}^{m(n)}$ and range $\{0,1\}^{m'(n)}$ is known to be equivalent to that of one-way functions [Rom90]. As noted by Katz and Vaikuntanathan [KV09], standard constructions of universal one-way hash functions are public coin. In practice, such public-coin functions can be constructed rather easily from various number-theoretic assumptions. For example, if the discrete log problem is hard in some group $\mathbb{G}$ of prime order $p$, the family of functions $f_{g_1,\ldots,g_k} : \mathbb{Z}_p^k \to \mathbb{G}$ defined as $f_{g_1,\ldots,g_k}(x_1, \ldots, x_k) = \prod_{i=1}^{k} g_i^{x_i}$ is second-preimage resistant (and even collision resistant), where $g_1, \ldots, g_k \in \mathbb{G}$ are chosen uniformly and independently at random by the key-generation algorithm.

We note that for public-coin SPR functions, there is actually no need for an explicit key-generation algorithm. Without loss of generality one can define a single function $\mathsf{F}'_r(x) = (r, \mathsf{F}_s(x))$, where $s = \mathsf{KeyGen}(1^n; r)$, and this is also SPR with the same amount of "lossiness" as the family $\mathcal{F}$.

### 2.4 Statistical Non-Interactive Witness-Indistinguishable Argument Systems

A non-interactive argument system for a language $L$ with witness relation $R_L$ is a triplet of algorithms $(\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$, where $\mathsf{CRSGen}$ is an algorithm generating a common reference string $\mathsf{crs}$, and $\mathsf{P}$ and $\mathsf{V}$ are the prover and verifier algorithms, respectively. The prover takes as input a triplet $(\mathsf{crs}, x, w)$, where $(x, w) \in R_L$, and outputs a proof $\pi$. The verifier takes as input a triplet $(\mathsf{crs}, x, \pi)$ and either accepts or rejects. In this paper we consider a setting where all three algorithms run in probabilistic polynomial time. The two requirements of an argument system are completeness and soundness with respect to efficient cheating provers. Informally, for every $(x, w) \in R_L$ the prover generates proofs that are always accepted by the verifier, and for every $x \notin L$ any efficient cheating prover has only a negligible probability of convincing the verifier to accept. An argument system is called statistical witness indistinguishable if for any $x \in L$ and any two witnesses $w_0 \neq w_1$ such that $(x, w_0), (x, w_1) \in R_L$, the proofs generated by $\mathsf{P}(\mathsf{crs}, x, w_0)$ and $\mathsf{P}(\mathsf{crs}, x, w_1)$ are statistically indistinguishable given the common reference string.

**Definition 2.3** (SNIWI argument system)**.** *A* statistical non-interactive witness-indistinguishable argument system *for a language $L$ with witness relation $R_L$ is a triplet of polynomial-time algorithms* $(\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ *such that the following properties hold:*

1. **Perfect completeness:** *For every $(x, w) \in R_L$ it holds that*
$$\Pr\left[\mathsf{V}(\mathsf{crs}, x, \mathsf{P}(\mathsf{crs}, x, w)) = 1\right] = 1 \ ,$$
*where* $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^n)$*, and the probability is taken over the internal randomness of* $\mathsf{CRSGen}$*,* $\mathsf{P}$*, and* $\mathsf{V}$*.*

2. **Soundness:** *For every $x \notin L$ and for every probabilistic polynomial-time prover* $\mathsf{P}^*$ *it holds that*
$$\Pr\left[\mathsf{V}(\mathsf{crs}, x, \mathsf{P}^*(\mathsf{crs}, x)) = 1\right] < \nu(n) \ ,$$
*for some negligible function $\nu(n)$, where* $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^n)$*, and the probability is taken over the internal randomness of* $\mathsf{CRSGen}$*,* $\mathsf{P}^*$*, and* $\mathsf{V}$*.*

3. **Statistical witness indistinguishability:** *For every $x \in L$ and two witnesses $w_0$ and $w_1$ such that $(x, w_0), (x, w_1) \in R_L$, the distributions* $\{\mathsf{crs}, \mathsf{P}(\mathsf{crs}, x, w_0)\}$ *and* $\{\mathsf{crs}, \mathsf{P}(\mathsf{crs}, x, w_1)\}$ *are statistically indistinguishable, where* $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^n)$*.*

For our construction we are interested in SNIWI argument systems for NP. Such an argument system is implied by the construction of Groth, Ostrovsky and Sahai [GOS06] that satisfies the stronger notion of a perfect non-interactive zero-knowledge argument system. Their construction can be based on the hardness of either the Decisional Subgroup problem [BGN05] or the Decisional Linear problem [BBS04]. As pointed out by Groth et al. we note that in their Linear-based construction the algorithm $\mathsf{CRSGen}$ admits oblivious sampling (specifically, the distribution of the common reference string is statistically-close to the uniform distribution), which is a technical property that is required for our construction in the bounded leakage model.

## 2.5 Lossy Public-Key Encryption

A lossy public-key encryption scheme is a public-key encryption scheme in which public keys can be generated in two modes that are computationally indistinguishable: an *injective* mode in which ciphertexts can be decrypted using a corresponding secret key, and a *lossy* mode in which ciphertexts statistically hide the encrypted messages. Lossy public-key encryption was recently found useful for various applications (see, for example, [BHY09, KN08, PVW08]), and its existence was shown to be equivalent to 2-move statistical semi-honest oblivious transfer via rather simple and efficient black-box reductions [HLO$^+$09]. In particular, this implies that it can be realized based also on any other primitive that is known to imply such oblivious transfer protocols, including in particular homomorphic encryption, 2-move private information retrieval, and lossy trapdoor functions. Thus, lossy public-key encryption schemes can be constructed based on the hardness of various number-theoretic problems, such as Decisional Diffie-Hellman (and, more generally, Decisional $d$-Linear), Quadratic Residuosity, Composite Residuosity, Learning With Errors, and more.

**Definition 2.4** (Lossy PKE)**.** *A* lossy public-key encryption scheme *is a 4-tuple of probabilistic polynomial-time algorithms* $(\mathsf{KeyGen}_0, \mathsf{KeyGen}_1, \mathsf{Enc}, \mathsf{Dec})$ *such that:*

1. **Lossy key generation:** $\mathsf{KeyGen}_0(1^n)$ *outputs a public key pk.*
2. **Injective key generation:** $\mathsf{KeyGen}_1(1^n)$ *outputs a secret key sk and a public key pk.*
3. **Lossiness under lossy keys:** *For every public key pk produced by* $\mathsf{KeyGen}_0(1^n)$*, and for every two messages* $m_0, m_1 \in \{0,1\}^{\ell(n)}$*, the statistical distance between the distributions* $\mathsf{Enc}_{pk}(m_0)$ *and* $\mathsf{Enc}_{pk}(m_1)$ *is negligible in* $n$*.*
4. **Decryption under injective keys:** *For every message* $m \in \{0,1\}^{\ell(n)}$ *it holds that*

   $$\Pr\left[\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m)) = m\right] > 1 - \nu(n) \ ,$$

   *for some negligible function* $\nu(n)$*, where* $(sk, pk) \leftarrow \mathsf{KeyGen}_1(1^n)$*, and the probability is taken over the internal randomness of* $\mathsf{KeyGen}_1$*,* $\mathsf{Enc}$ *and* $\mathsf{Dec}$*.*
5. **Indistinguishability of injective and lossy public keys:** *The two ensembles* $\{pk : pk \leftarrow \mathsf{KeyGen}_0(1^n)\}_{n\in\mathbb{N}}$ *and* $\{pk : (sk, pk) \leftarrow \mathsf{KeyGen}_1(1^n)\}_{n\in\mathbb{N}}$ *are computationally indistinguishable.*

For our application we need to be able to obliviously sample public descriptions that are computationally indistinguishable from those produced by $\mathsf{KeyGen}_0$ and $\mathsf{KeyGen}_1$. Specifically, we require that the public descriptions that are produced by $\mathsf{KeyGen}_0$ and $\mathsf{KeyGen}_1$ are computationally indistinguishable from the uniform distribution. This holds, for example, in the construction of Peikert et al. [PVW08] (based on the DDH assumption – see also [BHY09]), and in the constructions resulting from the lossy trapdoor functions of Peikert and Waters [PW08] (based on the DDH and LWE assumptions) and of Freeman et al. [FGK+10] (based on the $d$-Linear assumption).

## 2.6 Admissible Hash Functions

The concept of an *admissible hash function* was first defined by Boneh and Boyen [BB04] to convert a natural selectively-secure identity-based encryption scheme into a fully-secure one. In this paper we use such hash functions in a similar manner to convert a selectively-secure signature scheme (where the adversary declares the message to be forged ahead of time, before receiving the verification key) into a fully secure one. The main idea of an admissible hash function is that it allows the reduction in the proof of security to secretly partition the message space into two subsets, which we will label as red ($\mathtt{R}$) and blue ($\mathtt{B}$), such that there is a noticeable probably that all of the messages in the adversary's signing queries will be in the blue set, but the forgery will be on a message in the red set. This is useful if the simulator can efficiently answer signing queries in the blue set, yet break some hard problem given a valid forgery on a message from the red set. Our exposition and definition of admissible hash function follow that of Cash, Hofheinz, Kiltz, and Peikert [CHK+10].

For $K \in \{0,1,\perp\}^{\tau(n)}$, we define the function $F_K : \{0,1\}^{\tau(n)} \to \{\mathtt{R},\mathtt{B}\}$ which "colors" the space $\{0,1\}^{\tau(n)}$ in the following way:

$$F_K(y) := \begin{cases} \mathtt{R} & \text{if } \forall\, i \in \{1,\ldots,\tau(n)\} \ : \ K_i \neq y_i \\ \mathtt{B} & \text{if } \exists\, i \in \{1,\ldots,\tau(n)\} \ : \ K_i = y_i \end{cases}$$

For any $u = u(n) < \tau(n)$, we let $\mathcal{K}_{u,n}$ denote the uniform distribution over $\{0,1,\perp\}^{\tau(n)}$ *conditioned on* exactly $u$ positions having $\perp$ values. (Note, if $K$ is chosen from $\mathcal{K}_{u,n}$, then the map $F_K(\cdot)$ colors exactly $2^u$ values red.)

Let $\mathcal{H} = \{\mathcal{H}_n\}_{n\in\mathbb{N}}$ be a hash-function ensemble, where each $H \in \mathcal{H}_n$ is a polynomial-time computable function $H : \{0,1\}^* \to \{0,1\}^{\tau(n)}$. For each $H \in \mathcal{H}_n$, we define the function $F_{K,H} : \{0,1\}^* \to \{\mathtt{R},\mathtt{B}\}$, which "colors" the space $\{0,1\}^*$ according to $F_{K,H}(x) = F_K(H(x))$.

**Definition 2.5** (Admissible hash function [BB04, CHK$^+$10])**.** *We say that $\mathcal{H}$ is an* admissible hash-function ensemble *if for every $H \in \mathcal{H}$ there exists a set $\mathbf{bad}_H$ of string-tuples such that the following two properties hold:*

- *For every probabilistic polynomial-time algorithm $\mathcal{A}$ there exists a negligible function $\nu(n)$ satisfying*

$$\Pr[(x_0, \dots, x_q) \in \mathbf{bad}_H \mid H \leftarrow \mathcal{H}_n, (x_0, \dots, x_q) \leftarrow \mathcal{A}(1^n, H)] \leq \nu(n) \ .$$

- *For every polynomial $q = q(n)$ there is a polynomial $p = p(n)$ and an efficiently computable $u = u(n)$ such that, for every $H \in \mathcal{H}_n$ and $(x_0, \dots, x_q) \notin \mathbf{bad}_H$ with $x_0 \notin \{x_1, \dots, x_q\}$, we have:*

$$\Pr_{K \leftarrow \mathcal{K}_{u,n}} [F_{K,H}(x_0) = \mathtt{R} \wedge F_{K,H}(x_1) = \cdots = F_{K,H}(x_q) = \mathtt{B}\ ] \geq \frac{1}{p(n)} \ .$$

We note that for the application to identity-based encryption [BB04, CHK$^+$10] the bad sets $\mathbf{bad}_H$ are required to be efficiently recognizable, but this is not required for our application. In addition, we say that $\mathcal{H}$ is a *public-coin* admissible hash-function ensemble, if it satisfies Definition 2.5 even when the algorithm $\mathcal{A}$ takes as input also the internal randomness that was used by $\mathsf{KeyGen}(1^n)$ for sampling the function.

The work of Boneh and Boyen [BB04] shows how to construct admissible hash functions from collision-resistant hash functions. Moreover, if the underlying collision-resistant hash functions are public coin, then so are the resulting admissible hash functions. As already mentioned in Section 2.3, public-coin collision-resistant hash functions can be constructed rather easily from various number-theoretic assumptions.

## 3 Modeling Leakage-Resilient Signature Schemes

A signature scheme is a triplet $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ of probabilistic polynomial-time algorithms. The key-generation algorithm $\mathsf{KeyGen}$ receives as input a security parameter $1^n$, and outputs a verification key $vk$ and a signing key $sk$. The signing algorithm $\mathsf{Sign}$ receives as input a signing key $sk$ and a message $m$, and outputs a signature $\sigma$. The verification algorithm $\mathsf{Verify}$ receives as input a verification key $vk$, a message $m$, and a signature $\sigma$, and outputs a bit $b$. In terms of functionality, it is required that for any message $m$, with overwhelming probability it holds that $\mathsf{Verify}_{vk}(m, \mathsf{Sign}_{sk}(m)) = 1$, where the probability is taken over the internal coin tosses of $\mathsf{KeyGen}$, $\mathsf{Sign}$ and $\mathsf{Verify}$. In terms of security, the standard notion of security for signature schemes is *existential unforgeability under a chosen-message attack* [GMR88]: any efficient adversary that is given a randomly sampled verification key and adaptive access to a signing oracle (that sign messages using the corresponding signing key), should have only a negligible probability of producing a valid signature on a message on which she did not query the signing oracle.

In this section we define the various notions of security for fully leakage-resilient signature schemes that are considered in this paper. We consider two different models: the bounded-leakage model (see Section 3.1) and the continual-leakage model (see Section 3.2).

### 3.1 The Bounded-Leakage Model

A signature scheme is fully leakage-resilient (FLR) in the bounded-leakage model if it is existentially unforgeable against an adversary that can obtain both signatures on any message of her choice, and bounded leakage information on all intermediate values used by the signer throughout the lifetime of the system. These consist of all intermediate values used by the key-generation algorithm, as well as all intermediate values used by the signing algorithm.

As discussed in Section 1.3, this is formalized by considering an experiment that involves a signer and an adversary. First, the signer invokes the key-generation algorithm and obtains a verification key $vk$ and a signing key $sk$. At this point, a set $\mathsf{state}$ is initialized to contain the random coins that were used by the key-generation algorithm, and the adversary is given the verification key $vk$. Then, the adversary can adaptively submit two types of queries: signing queries, and leakage queries. A signing query consists of a message $m$, and is answered by invoking the signing algorithm with the signing key and the message. Following each such query, the random coins that were used by the signing algorithm are added to the set $\mathsf{state}$. A leakage query consists of a leakage function $f$, and is answered by applying $f$ to the set $\mathsf{state}$. The leakage functions have to be efficiently computable, and the sum of their output lengths has to be upper bounded by a predetermined parameter $\lambda$. As in the standard notion of security for signature schemes, the adversary is successful if she outputs a pair $(m^*, \sigma^*)$, where $m^*$ is a message with which the adversary did not issue a signing query, and $\sigma^*$ is a valid signature on $m^*$ with respect to $vk$. More formally, the following definition is due to Katz and Vaikuntanathan [KV09]:

**Definition 3.1** (FLR security — bounded leakage)**.** *A signature scheme* $\Pi = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is* $\lambda$-fully-leakage-resilient in the bounded-leakage model *if for any probabilistic polynomial-time adversary* $\mathcal{A}$ *it holds that* $\Pr\left[\mathsf{Success}^{\lambda\text{-}\mathsf{FLR}}_{\Pi,\mathcal{A}}(n)\right]$ *is negligible in* $n$*, where the event* $\mathsf{Success}^{\lambda\text{-}\mathsf{FLR}}_{\Pi,\mathcal{A}}(n)$ *is defined via the following experiment:*

1. *Sample* $r \leftarrow \{0,1\}^*$*, compute* $(vk, sk) = \mathsf{KeyGen}(1^n; r)$*, and set* $\mathsf{state} = \{r\}$*.*

2. *The adversary* $\mathcal{A}$ *receives as input the pair* $(1^n, vk)$*, and can adaptively query a signing oracle and a leakage oracle that are defined as follows:*

   - **Signing queries.** *The signing oracle receives as input a message* $m_i$*, samples* $r_i \leftarrow \{0,1\}^*$*, and then computes* $\sigma_i \leftarrow \mathsf{Sign}_{sk}(m_i; r_i)$*. It updates* $\mathsf{state} := \mathsf{state} \cup \{r_i\}$ *and outputs* $\sigma_i$*.*
   - **Leakage queries.** *The leakage oracle receives as input a description of an efficiently computable function* $f_j : \{0,1\}^* \to \{0,1\}^{\lambda_j}$*, and outputs* $f_j(\mathsf{state})$*. We call* $\lambda_j$ *the* output length *of the* $j$*-th leakage function.*

3. *The adversary* $\mathcal{A}$ *outputs a pair* $(m^*, \sigma^*)$*.*

4. $\mathsf{Success}^{\lambda\text{-}\mathsf{FLR}}_{\Pi,\mathcal{A}}(n)$ *denotes the event in which:*

   - $\mathsf{Verify}_{vk}(m^*, \sigma^*) = 1$.
   - $m^*$ *was not queried to the signing oracle.*
   - *The sum of output lengths of all leakage functions is at most* $\lambda(n)$*.*

### 3.2 The Continual-Leakage Model

In the continual-leakage model, a signature scheme also includes an additional "key-refresh algorithm" $sk' \leftarrow \mathsf{Refresh}_{pk}(sk)$, which the signer can use at any point in time to refresh his signing key. Each new signing key $sk'$ produced by the key-refresh algorithm is functionally equivalent to the original key. We imagine a setting where the signer periodically updates her signing key, while the adversary is continuously leaking information about the state of the system. We model this with as an attack which consists of arbitrarily many leakage *epochs*, during each of which the adversary can learn an additional $\lambda$ bits of information about the current state of the system. In the beginning of the first epoch, the set $\mathsf{state}$ consists of just the signing key $sk$ produced by key-generation algorithm. During each epoch, the adversary can adaptively issue signing queries, where the randomness of signing algorithm is appended to the set $\mathsf{state}$, and leakage queries for up to $\lambda$ bits of information about the $\mathsf{state}$. At any point in time the adversary can move to the next epoch by issuing a key-refresh query, which results in the set $\mathsf{state}$ being reset[10] to $sk \leftarrow \mathsf{Refresh}_{pk}(sk)$. Notice that, since there is no bound on the number of epochs, there is also no bound on the overall amount of leakage the adversary can learn during the attack game.

**Definition 3.2** (CFLR security — continual leakage)**.** *A signature scheme* $\Pi = (\mathsf{KeyGen}, \mathsf{Refresh}, \mathsf{Sign}, \mathsf{Verify})$ *is* $\lambda$-fully-leakage-resilient in the continual-leakage model (CFLR) *if for any probabilistic polynomial-time adversary* $\mathcal{A}$ *it holds that* $\Pr\left[\mathsf{Success}^{\lambda\text{-}\mathsf{CFLR}}_{\Pi,\mathcal{A}}(n)\right]$ *is negligible in* $n$*, where the event* $\mathsf{Success}^{\lambda\text{-}\mathsf{CFLR}}_{\Pi,\mathcal{A}}(n)$ *is defined via the following experiment:*

1. *Sample* $(vk, sk) \leftarrow \mathsf{KeyGen}(1^n)$*, and set* $\mathsf{state} := sk$ *and* $L = 0$*.*
2. *The adversary* $\mathcal{A}$ *receives as input the pair* $(1^n, vk)$*, and can adaptively issue the following types of queries:*
    - **Signing queries.** *The signing oracle receives as input a message* $m_i$*, samples* $r_i \leftarrow \{0,1\}^*$*, and then computes* $\sigma_i \leftarrow \mathsf{Sign}_{sk}(m_i; r_i)$*. It updates* $\mathsf{state} := \mathsf{state} \cup \{r_i\}$ *and outputs* $\sigma_i$*.*
    - **Leakage queries.** *The leakage oracle receives as input a description of an efficiently computable function* $f_j : \{0,1\}^* \to \{0,1\}^{\lambda_j}$*. If* $L + \lambda_j \leq \lambda(n)$ *then it outputs* $f_j(\mathsf{state})$*, and updates* $L := L + \lambda_j$*. Otherwise it outputs* $\perp$*.*
    - **Key-refresh queries.** *On a key-refresh query, the signing key is refreshed by sampling* $sk' \leftarrow \mathsf{Refresh}_{pk}(sk)$ *and setting* $sk := sk'$*. In addition, we reset* $\mathsf{state} := sk$ *and* $L := 0$*.*
3. *The adversary* $\mathcal{A}$ *outputs a pair* $(m^*, \sigma^*)$*.*
4. $\mathsf{Success}^{\lambda\text{-}\mathsf{FLR}}_{\Pi,\mathcal{A}}(n)$ *denotes the event in which:*
    - $\mathsf{Verify}_{vk}(m^*, \sigma^*) = 1$*.*
    - $m^*$ *was never queried to the signing oracle.*

Note that our definition for the continual-leakage model does not consider leakage of the randomness used by the key-generation algorithm[11] or the key-refresh algorithm. We could define a stronger definition which also allows some leakage from these algorithms. As shown in [BTK+10] and [DHL+10a], any scheme that satisfies the basic definition (as stated) with some super-logarithmic $\lambda(n)$, is also generically secure if an additional $O(\log(n))$ bits are leaked about the randomness of key-generation and *each of* the key-refresh executions. However, it remains an important open problem to come up with a specific scheme that beats the above generic bounds and allows for super-logarithmic leakage. Therefore, to keep the definition simple, we do not consider these forms of leakage in Definition 3.2.

[10] The necessary requirement that the state is reset models the ability of the honest parties to *erase/overwrite* their prior signing key and the randomness used in prior executions during a key refresh.

[11] This is in contrast to our bounded-leakage definition, which does consider such leakage.

## 4 $\mathcal{R}$-Lossy Public-Key Encryption

In this section we introduce the notion of an $\mathcal{R}$-lossy public-key encryption scheme. Informally, such a scheme is a tag-based public-key encryption scheme where the set of possible tags is partitioned into two subsets: *injective* tags, and *lossy* tags. When a message is encrypted under an injective tag, the resulting ciphertext can be correctly decrypted using the secret key. On the other hand, when encrypted under a lossy tag, the ciphertext statistically hides the message. The partitioning of the tags in defined by a binary relation $\mathcal{R}$: the key-generation algorithm receives as input an initialization value $K$, and this specifies that a tag $t$ is injective if and only if $(K, t) \in \mathcal{R}$. The only computational requirement is that the public key of the encryption scheme hides the initialization value $K$. That is, public keys produced by different initialization values are computationally indistinguishable.

For our constructions, we are mainly interested in a relation $\mathcal{R}^{\mathsf{AD}}$ that partitions the set of tags according to the same red/blue coloring that we used when defining admissible hash functions (see Section 2.6). In the notation of Section 2.6, any initialization value $K \in \{0, 1, \perp\}^{\tau(n)}$ partitions the set of tags according to the function $F_K$ into red tags and blue tags. The red tags are the injective tags, and the blue tags are the lossy tags. Specifically, a tag $t \in \{0,1\}^{\tau(n)}$ is injective if and only if for every $i \in \{1, \dots, \tau(n)\}$ it holds that $K_i \neq t_i$. We discuss this relation below in more detail.

Formally, an efficiently computable binary relation $\mathcal{R} \subseteq \mathcal{K} \times \mathcal{T}$ is a sequence of efficiently recognizable sets $\mathcal{R}_n \subseteq \mathcal{K}_n \times \mathcal{T}_n$. We refer to $\mathcal{K}_n$ as the set of initialization values, and to $\mathcal{T}_n$ as the set of tags for a given security parameter $n$. An $\mathcal{R}$-lossy public-key encryption scheme is defined as follows.

**Definition 4.1** ($\mathcal{R}$-lossy PKE). *Let $\mathcal{R} \subseteq \mathcal{K} \times \mathcal{T}$ be an efficiently computable binary relation. An $\mathcal{R}$-*lossy public-key encryption scheme *is a triplet of probabilistic polynomial-time algorithms* ($\mathsf{KeyGen}$, $\mathsf{Enc}, \mathsf{Dec}$) *such that:*

1. **Key generation:** *For any initialization value $K \in \mathcal{K}_n$, the algorithm $\mathsf{KeyGen}(1^n, K)$ outputs a secret key $sk$ and a public key $pk$.*

2. **Decryption under injective tags:** *For any initialization value $K \in \mathcal{K}_n$ and tag $t \in \mathcal{T}_n$ such that $(K, t) \in \mathcal{R}_n$, and for any message $m \in \{0,1\}^{\ell(n)}$, it holds that*

$$\Pr\left[\mathsf{Dec}^t_{sk}(\mathsf{Enc}^t_{pk}(m)) = m\right] > 1 - \nu(n) \ ,$$

*for some negligible function $\nu(n)$, where $(sk, pk) \leftarrow \mathsf{KeyGen}(1^n, K)$, and the probability is taken over the internal randomness of* $\mathsf{KeyGen}$, $\mathsf{Enc}$ *and* $\mathsf{Dec}$.

3. **Lossiness under lossy tags:** *For any initialization value $K \in \mathcal{K}_n$ and tag $t \in \mathcal{T}_n$ such that $(K, t) \notin \mathcal{R}_n$, for every pair $(sk, pk)$ of keys produced by $\mathsf{KeyGen}(1^n, K)$, and for every two messages $m_0, m_1 \in \{0,1\}^{\ell(n)}$, the distributions $\mathsf{Enc}^t_{pk}(m_0)$ and $\mathsf{Enc}^t_{pk}(m_1)$ are statistically indistinguishable.*

4. **Indistinguishability of initialization values:** *For every sequence $\{(K_n, K'_n)\}_{n \in \mathbb{N}}$ such that $K_n, K'_n \in \mathcal{K}_n$, the two ensembles $\{pk : (sk, pk) \leftarrow \mathsf{KeyGen}(1^n, K_n)\}_{n \in \mathbb{N}}$ and $\{pk : (sk, pk) \leftarrow \mathsf{KeyGen}(1^n, K'_n)\}_{n \in \mathbb{N}}$ are computationally indistinguishable.*

As with the other primitives that are used on our construction, we need to be able to obliviously sample public keys in a way that is computationally indistinguishable from those produced by $\mathsf{KeyGen}(1^n, \cdot)$. Specifically, we require that there exists a sequence of initialization values $\{K_n\}_{n \in \mathbb{N}}$ such that the ensemble $\{pk : (sk, pk) \leftarrow \mathsf{KeyGen}(1^n, K_n)\}_{n \in \mathbb{N}}$ is computationally indistinguishable from the uniform distribution over $\{0,1\}^*$. Note that by the indistinguishability of initialization values property defined above, this in fact holds for every sequence $\{K_n\}_{n\in\mathbb{N}}$.

For our constructions of fully leakage-resilient signature schemes we consider two relations: the equality relation $\mathcal{R}^{\mathsf{EQ}}$, and the relation $\mathcal{R}^{\mathsf{AD}}$ that was mentioned above.

**The relation $\mathcal{R}^{\mathsf{EQ}}$.** An $\mathcal{R}^{\mathsf{EQ}}$-lossy public-key encryption scheme, where $\mathcal{R}^{\mathsf{EQ}}$ is the equality relation for binary tags of length $\tau(n)$ bits[12] is an *all-but-one-lossy* (ALBO) public-key encryption scheme, a primitive discussed in Section 1.3. In this case there is one injective tag, and all the other tags are lossy. Using an $\mathcal{R}^{\mathsf{EQ}}$-lossy public-key encryption scheme in our construction results in fully leakage-resilient signature schemes (both in the bounded-leakage and continual-leakage models) that are selectively unforgeable under a chosen-message attack (SU-CMA). This is a seemingly weak notion of unforgeability that is obtained from Definition 3.1 by requiring the adversary to specify the message $m^*$ in advance before receiving the verification key. The adversary can still adaptively interact with the signing oracle and leakage oracle after receiving the verification key.

**The relation $\mathcal{R}^{\mathsf{AD}}$.** Building on the approach of Boneh and Boyen [BB04] for transforming selectively-secure identity-based encryption schemes into fully secure ones using admissible hash functions (see Section 2.6), we consider a bit more subtle relation $\mathcal{R}^{\mathsf{AD}}$. This relation enables us to directly achieve the stronger notion of *existential unforgeability under a chosen-message attack* (see Definition 3.1). The relation $\mathcal{R}^{\mathsf{AD}}$ is defined as follows. For every $n \in \mathbb{N}$ we let $\mathcal{K}_n = \{0,1,\perp\}^{\tau(n)}$ and $\mathcal{T}_n = \{0,1\}^{\tau(n)}$, where $\tau(n)$ is the length of the tags, and define

$$\mathcal{R}^{\mathsf{AD}}_n = \left\{ (K,t) \in \{0,1,\perp\}^{\tau(n)} \times \{0,1\}^{\tau(n)} : K_i \neq t_i \ \forall i \in \{1,\ldots,\tau(n)\} \right\} .$$

That is, in the notation introduced in Section 2.6, any initialization value $K \in \{0,1,\perp\}^{\tau(n)}$ partitions the set of tags according to the function $F_K$ into red tags and blue tags. The red tags are the injective tags, and the blue tags are the lossy tags.

**Constructions.** We propose two constructions of $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption schemes[13]. Our first construction is rather generic and is based on any lossy public-key encryption scheme (recall Section 2.5). In turn, this implies $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption schemes can be based on a variety of number-theoretic assumptions. Our second construction is based on a specific number-theoretic assumption (the DDH assumption[14]) and is significantly more efficient than our generic construction.

## 4.1 A Generic Construction of $\mathcal{R}^{\mathsf{AD}}$-Lossy PKE from Lossy PKE

Let $\Pi = (\mathsf{KeyGen}_0, \mathsf{KeyGen}_1, \mathsf{Enc}, \mathsf{Dec})$ be any lossy public-key encryption scheme. The key-generation algorithm of our $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption scheme samples $\tau(n)$ pairs of public keys of the scheme $\Pi$. Each such pair is of one out of three possible types according to the symbols of the initialization value $K \in \{0,1,\perp\}^{\tau(n)}$. For every $1 \in \{1,\ldots,\tau(n)\}$, if $K_i = 0$ then the $i$-th pair consists of a lossy key and an injective key, if $K_i = 1$ then the $i$-th pair consists of an injective and a lossy key (i.e., the order is reversed), and if $K_i = \perp$ then $i$-th pair consists of two injective keys[15]. A

[12]That is, $\mathcal{K}_n = \mathcal{T}_n = \{0,1\}^{\tau(n)}$, and $(K,t) \in \mathcal{R}^{\mathsf{EQ}}_n$ if and only if $K = t$.

[13]We note that rather straightforward variants of these constructions yield $\mathcal{R}^{\mathsf{EQ}}$-lossy public-key encryption schemes.

[14]Our construction easily generalizes to rely on the $d$-Linear assumption for any $d \geq 1$.

[15]Observe that if the underlying lossy encryption scheme allows oblivious sampling of public keys, then so does our construction.

message is encrypted under a tag $t \in \{0,1\}^{\tau(n)}$ by using a $\tau(n)$-out-of-$\tau(n)$ (information-theoretic) secret-sharing scheme to share the message, and then encrypting the $i$-th share using one of the keys from the $i$-th pair of keys according to the $i$-th bit of $t$. More formally, consider the following encryption scheme $\Pi' = (\mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$:

- **Key generation:** On input $1^n$ and an initialization value $K = K_1 \cdots K_{\tau(n)} \in \{0,1,\perp\}^{\tau(n)}$, for every $1 \le i \le \tau(n)$ the algorithm $\mathsf{KeyGen}'$ produces a pair $((sk_{i,0}, pk_{i,0}), (sk_{i,1}, pk_{i,1}))$ as follows:
  - If $K_i = 0$ then it samples $pk_{i,0} \leftarrow \mathsf{KeyGen}_0(1^n)$, $(sk_{i,1}, pk_{i,1}) \leftarrow \mathsf{KeyGen}_1(1^n)$, and sets $sk_{i,0} = \perp$.
  - If $K_i = 1$ then it samples $(sk_{i,0}, pk_{i,0}) \leftarrow \mathsf{KeyGen}_1(1^n)$, and $pk_{i,1} \leftarrow \mathsf{KeyGen}_0(1^n)$, and sets $sk_{i,1} = \perp$.
  - If $K_i = \perp$ then it samples $(sk_{i,0}, pk_{i,0}) \leftarrow \mathsf{KeyGen}_1(1^n)$ and $(sk_{i,1}, pk_{i,1}) \leftarrow \mathsf{KeyGen}_1(1^n)$.

  It then outputs the pair $(sk, pk)$ defined as

  $$sk = \left(K, \{(sk_{i,0}, sk_{i,1})\}_{i=1}^{\tau(n)}\right)$$

  $$pk = \left(\{(pk_{i,0}, pk_{i,1})\}_{i=1}^{\tau(n)}\right)$$

- **Encryption:** On input a public key $pk$ of the above form, a tag $t = t_1 \cdots t_{\tau(n)} \in \{0,1\}^{\tau(n)}$ and a message $m \in \{0,1\}^{\ell(n)}$, the algorithm $\mathsf{Enc}'$ uses a $\tau(n)$-out-of-$\tau(n)$ (information-theoretic) secret-sharing scheme to compute shares $(m_1, \dots, m_{\tau(n)})$ of $m$, and outputs a ciphertext $c$ defined as
  $$c = \left(\mathsf{Enc}_{pk_{1,t_1}}(m_1), \dots, \mathsf{Enc}_{pk_{\tau(n),t_{\tau(n)}}}(m_{\tau(n)})\right)$$

- **Decryption:** On input a secret key $sk$ of the above form, a tag $t = t_1 \cdots t_{\tau(n)} \in \{0,1\}^{\tau(n)}$ and a ciphertext $c = (c_1, \dots, c_{\tau(n)})$, the algorithm $\mathsf{Dec}'$ proceeds as follows. If $(K, t) \notin \mathcal{R}^{\mathsf{AD}}$ (i.e., $t$ is a lossy tag) then it outputs $\perp$. Otherwise (i.e., $t$ is an injective tag), for every $1 \le i \le \tau(n)$ it computes $m_i = \mathsf{Dec}_{sk_{i,t_i}}(c_i)$, and uses the reconstruction procedure of the secret sharing scheme to output the message $m$ corresponding to the shares $(m_1, \dots, m_{\tau(n)})$.

**Theorem 4.2.** *If* $\Pi = (\mathsf{KeyGen}_0, \mathsf{KeyGen}_1, \mathsf{Enc}, \mathsf{Dec})$ *is a lossy public-key encryption scheme, then* $\Pi' = (\mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$ *is an* $\mathcal{R}^{\mathsf{AD}}$*-lossy public-key encryption scheme.*

**Proof.** Indistinguishability of initialization values follows directly from the indistinguishability of lossy and injective public keys of the underlying lossy encryption scheme via a straightforward hybrid argument. The correctness of the decryption algorithm under injective tags follows from the fact that when encrypting a message under an injective tag $t$ (i.e., for every $i \in \{1, \dots, \tau(n)\}$ it holds that $K_i \neq t_i$), each share of the message is encrypted under an injective public key of the underlying lossy encryption scheme. Lossiness of encryption under lossy tags follows from the fact that when encrypting a message under any lossy tag $t$ (i.e., there exists some $i \in \{1, \dots, \tau(n)\}$ for which $K_i = t_i$), at least one of the shares is encrypted using a lossy public key. This guarantees that the ciphertext corresponding to this share is statistically indistinguishable from an encryption of any other share under the same lossy public key. Thus, the $\tau(n)$-out-of-$\tau(n)$ (information-theoretic) secret-sharing scheme implies that encryptions of any two messages under any lossy tag are statistically indistinguishable. ■

## 4.2 A More Efficient Construction of $\mathcal{R}^{\mathsf{AD}}$-Lossy PKE Based on DDH

We now present a specific DDH-based scheme with better efficiency than the generic construction from Section 4.1. In this scheme, the public key still consists of $\tau(n)$ pairs, where $\tau(n)$ is the length of the tags, but each ciphertext consists of only two group elements. This scheme satisfies the requirements of Definition 4.1 with overwhelming probability over the internal randomness of the key-generation algorithm (oblivious sampling of public keys is always guaranteed), which will be sufficient for the security of our constructions in this paper. We refer to a scheme that satisfies this slightly weaker guarantee as an *almost-always* $\mathcal{R}$-lossy public-key encryption scheme.

Let $\mathsf{GroupGen}$ be a probabilistic polynomial-time algorithm that takes a security parameter $1^n$ as input and outputs a triplet $(\mathbb{G}, p, g)$, where $\mathbb{G}$ is a group of order $p$ generated by $g \in \mathbb{G}$, and $p$ is an $n$-bit prime number. Consider the following encryption scheme $\Pi_{\mathsf{DDH}} = (\mathsf{KeyGen}_{\mathsf{DDH}}, \mathsf{Enc}_{\mathsf{DDH}}, \mathsf{Dec}_{\mathsf{DDH}})$:

- **Key generation:** On input $1^n$ and an initialization value $K = K_1 \cdots K_{\tau(n)} \in \{0, 1, \perp\}^{\tau(n)}$, the algorithm $\mathsf{KeyGen}_{\mathsf{DDH}}$ samples $(\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^n)$, together with a uniformly distributed element $h \leftarrow \mathbb{G}$. Then, for every $1 \leq i \leq \tau(n)$ it samples $\alpha_{i,0}, \beta_{i,0}, \alpha_{i,1}, \beta_{i,1} \leftarrow \mathbb{Z}_p$ uniformly and independently at random, and continues as follows:
  - If $K_i = 0$ then it sets $\alpha_{i,1} = \beta_{i,1}$.
  - If $K_i = 1$ then it sets $\alpha_{i,0} = \beta_{i,0}$.
  - If $K_i = \perp$ then it sets $\alpha_{i,0} = \beta_{i,0}$ and $\alpha_{i,1} = \beta_{i,1}$.

  Finally, for every $1 \leq i \leq \tau(n)$ and $b \in \{0, 1\}$ it sets $(u_{i,b}, v_{i,b}) = \left(g^{\alpha_{i,b}}, h^{\beta_{i,b}}\right)$, and outputs the pair $(sk, pk)$ defined as

  $$sk = \left(K, \{(\alpha_{i,0}, \beta_{i,0}), (\alpha_{i,1}, \beta_{i,1})\}_{i=1}^{\tau(n)}\right)$$

  $$pk = \left(g, h, \{(u_{i,0}, v_{i,0}), (u_{i,1}, v_{i,1})\}_{i=1}^{\tau(n)}\right)$$

- **Encryption:** On input a public key $pk$ of the above form, a tag $t = t_1 \cdots t_{\tau(n)} \in \{0,1\}^{\tau(n)}$ and a message $m \in \mathbb{G}$, the algorithm $\mathsf{Enc}_{\mathsf{DDH}}$ chooses $r, r' \in \mathbb{Z}_p$ uniformly and independently at random, computes

  $$u_t = \prod_{i=1}^{\tau(n)} u_{i,t_i} \ , \quad v_t = \prod_{i=1}^{\tau(n)} v_{i,t_i} \ , \quad c_1 = g^r h^{r'} \ , \quad c_2 = (u_t)^r (v_t)^{r'} \cdot m \ ,$$

  and outputs the ciphertext $(c_1, c_2)$.

- **Decryption:** On input a secret key $sk$ of the above form, a tag $t = t_1 \cdots t_{\tau(n)} \in \{0,1\}^{\tau(n)}$ and a ciphertext $c = (c_1, c_2)$, the algorithm $\mathsf{Dec}_{\mathsf{DDH}}$ proceeds as follows. If $(K, t) \notin \mathcal{R}^{\mathsf{AD}}$ (i.e., $t$ is a lossy tag) then it outputs $\perp$. Otherwise (i.e., $t$ is an injective tag), it outputs the message $m$ defined as

  $$m = c_2 \cdot \left(c_1^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}}\right)^{-1} \ .$$

**Theorem 4.3.** *Assuming the hardness of the DDH problem, then* $\Pi_{\mathsf{DDH}} = (\mathsf{KeyGen}_{\mathsf{DDH}}, \mathsf{Enc}_{\mathsf{DDH}}, \mathsf{Dec}_{\mathsf{DDH}})$ *is an almost-always* $\mathcal{R}^{\mathsf{AD}}$*-lossy public-key encryption scheme for tags of length* $\tau(n) \leq \log p - \omega(\log n)$.

**Proof.** Indistinguishability of initialization values follows directly from the hardness of the DDH problem and a standard hybrid argument. In addition, (perfect) correctness of the decryption algorithm under injective tags follows from the fact that for any injective tag $t = t_1 \cdots t_{\tau(n)}$ (i.e., for every $i \in \{1, \dots, \tau(n)\}$ it holds that $K_i \neq t_i$) it holds that $\alpha_{i,t_i} = \beta_{i,t_i}$ for every $i \in \{1, \dots, \tau(n)\}$. Therefore, for any ciphertext $(c_1, c_2)$ that is produced by encrypting a message $m$ under an injective tag $t$ it holds that

$$
\begin{aligned}
c_2 \cdot \left( c_1^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \right)^{-1} &= (u_t)^r (v_t)^{r'} \cdot m \cdot \left( \left( g^r h^{r'} \right)^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \right)^{-1} \\
&= \left( \prod_{i=1}^{\tau(n)} u_{i,t_i} \right)^r \left( \prod_{i=1}^{\tau(n)} v_{i,t_i} \right)^{r'} \cdot m \cdot \left( \left( g^r h^{r'} \right)^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \right)^{-1} \\
&= \left( \prod_{i=1}^{\tau(n)} g^{\alpha_{i,t_i}} \right)^r \left( \prod_{i=1}^{\tau(n)} h^{\beta_{i,t_i}} \right)^{r'} \cdot m \cdot \left( \left( g^r h^{r'} \right)^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \right)^{-1} \\
&= \left( \prod_{i=1}^{\tau(n)} g^{\alpha_{i,t_i}} \right)^r \left( \prod_{i=1}^{\tau(n)} h^{\alpha_{i,t_i}} \right)^{r'} \cdot m \cdot \left( \left( g^r h^{r'} \right)^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \right)^{-1} \\
&= \left( g^r h^{r'} \right)^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \cdot m \cdot \left( \left( g^r h^{r'} \right)^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \right)^{-1} \\
&= m \ .
\end{aligned}
$$

Finally, we prove that lossiness under lossy tags holds with probability at least $1 - 2^{\tau(n)}/p$ over the internal randomness of the key-generation algorithm. Fix an initialization value $K \in \{0, 1, \perp\}^{\tau(n)}$, and a lossy tag $t \in \{0, 1\}^{\tau(n)}$ (i.e., there exists some $i \in \{1, \dots, \tau(n)\}$ for which $K_i = t_i$). Then, there exists some $i \in \{1, \dots, \tau(n)\}$ for which the values $\alpha_{i,t_i}$ and $\beta_{i,t_i}$ are uniformly and *independently* chosen, and therefore the values $u_t$ and $v_t$ as defined by the encryption algorithm are independently and uniformly distributed in $\mathbb{G}$. Thus, with probability $1 - 1/p$, it holds that $(g, h, u_t, v_t)$ is not a DDH tuple (i.e., $\log_g(u_t) \neq \log_h(v_t)$). In this case, the elements $g^r h^{r'}$ and $(u_t)^r (v_t)^{r'}$ are also independently and uniformly distributed in $\mathbb{G}$ over the choice of $r$ and $r'$ (see, for example, [PVW08, Lemma 4]). This implies that a ciphertext $(c_1, c_2)$ encrypted under a lossy tag $t$ carries no information on the message. This holds for any specific lossy tag $t$, and therefore a union bound guarantees that this holds for all lossy tags with probability at least $1 - 2^{\tau(n)}/p$. ■

## 5 A Signature Scheme in the Bounded-Leakage Model

In this section we present a fully leakage-resilient signature scheme in the bounded-leakage model (see Definition 3.1). Let $\mathcal{F} = (\mathsf{KeyGen}_{\mathsf{SPR}}, \mathsf{F})$ be a family of public-coin second-preimage resistant functions $\mathsf{F}_s(\cdot) : \{0,1\}^n \to \{0,1\}^{n^\epsilon}$ for some constant $\epsilon < 1$, let $\mathcal{H}$ be a public-coin admissible hash function ensemble, let $(\mathsf{KeyGen}_{\mathcal{R}^{\mathsf{AD}}}, \mathsf{Enc}, \mathsf{Dec})$ be an $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption scheme, and let $(\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ be a SNIWI argument system for the language

$$L = \{(s, y, pk, t, C) : \exists x, \omega \text{ st } C = \mathsf{Enc}_{pk}^t(x; \omega) \text{ and } \mathsf{F}_s(x) = y\}.$$

We assume that the distribution of public keys and common-reference strings produced by the algorithms $\mathsf{KeyGen}_{\mathcal{R}^{\mathsf{AD}}}$ and $\mathsf{CRSGen}$, respectively, are computationally indistinguishable from the uniform distribution over $\{0,1\}^*$. We refer the reader to Section 2 for the definitions of these primitives and a discussion on oblivious sampling of public descriptions. Consider the following signature scheme $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$:

- **Key generation:** On input $1^n$, the algorithm $\mathsf{KeyGen}$ samples a uniformly distributed $x \leftarrow \{0,1\}^n$, a function description $s \leftarrow \mathsf{KeyGen}_{\mathsf{SPR}}(1^n)$ from the SPR family, and computes $y = \mathsf{F}_s(x)$. Then, it samples a description of an admissible hash function $H \leftarrow \mathcal{H}_n$, and samples $pk \leftarrow \{0,1\}^*$ and $\mathsf{crs} \leftarrow \{0,1\}^*$ to be used as a public key for the $\mathcal{R}^{\mathsf{AD}}$-lossy encryption scheme and a common-reference string for the SNIWI argument system, respectively. It outputs the signing key $sk = x$ and the verification key $vk = (s, y, H, pk, \mathsf{crs})$.

- **Signing:** On input message $m$, the algorithm $\mathsf{Sign}$ computes an encryption $C = \mathsf{Enc}_{pk}^{H(m)}(x; \omega)$ of $x$ under the tag $H(m)$ using fresh randomness $\omega$. Then, it invokes the prover of the argument system to obtain a proof $\pi \leftarrow \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C), (x, \omega))$, and outputs the signature $(C, \pi)$.

- **Verifying:** On input message $m$ and signature $\sigma = (C, \pi)$, the algorithm $\mathsf{Verify}$ invokes the verifier of the argument system and outputs 1 if and only if $\mathsf{V}(\mathsf{crs}, (s, y, pk, H(m), C), \pi) = 1$.

**Theorem 5.1.** *The scheme* $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is $\lambda$-fully-leakage-resilient in the bounded-leakage model for any $\lambda \leq n - n^{\epsilon} - \omega(\log n)$.*

**Proof.** Fix a probabilistic polynomial-time adversary $\mathcal{A}$, and let $q = q(n)$ be a polynomial upper bound on the number of signing queries made by $\mathcal{A}$ in any execution. Without loss of generality we assume that $\mathcal{A}$ always submits $q$ signing queries, and we denote them by $m_1, \ldots, m_q$. Let $\mathcal{K}_{u,n}$ be the distribution from Definition 2.5 for the appropriate setting of $u = u(n)$ corresponding to $q$. The proof consists of a sequence of experiments.

**Experiment 0:** This is the original experiment in the definition of $\lambda$-fully-leakage-resilience, as described in Section 3. Note that the initial state is $\mathsf{state} = \{x, r_s, r_H, pk, \mathsf{crs}\}$, where $r_s$ and $r_H$ denote the randomness used to generate $s$ and $H$, respectively. In this experiment we also sample $K \leftarrow \mathcal{K}_{u,n}$, but this value is not used by the challenger in any way.

Let $\mathsf{Forge}_0$ be the event that the signature $(C^*, \pi^*)$ produced by $\mathcal{A}$ at the conclusion of Experiment 0 is valid, and that $m^* \neq m_i$ for every $i \in \{1, \ldots, q\}$. In addition, let $\mathsf{CorrectHash}_0$ be the event that all signature queries $m_1, \ldots, m_q$ made by $\mathcal{A}$ during the course of Experiment 0 fall into the set of "Blue" tags defined by $K$, while the forged message $m^*$ falls into the set of "Red" tags. That is, in the notation of Section 2.6, it holds that $F_{K,H}(m_1) = \cdots = F_{K,H}(m_q) = \mathtt{B}$ and $F_{K,H}(m^*) = \mathtt{R}$.

**Claim 5.2.** *There exists a polynomial $p(n)$ such that*

$$\Pr[\mathsf{Forge}_0 \wedge \mathsf{CorrectHash}_0] \geq \Pr[\mathsf{Forge}_0]/p(n) - \mathsf{negl}(n) \ .$$

**Proof.** Using the notation of Definition 2.5, let $\mathsf{Bad}$ be the event $(m^*, m_1, \ldots, m_q) \in \mathbf{bad}_H$. Then Definition 2.5 clearly implies that $\Pr[\mathsf{Bad}] \leq \mathsf{negl}(n)$. Let $p(\cdot)$ be as in Definition 2.5. Then

$$\begin{aligned} \Pr[\mathsf{Forge}_0 \wedge \mathsf{CorrectHash}_0] &\geq \Pr[\mathsf{Forge}_0 \wedge \mathsf{CorrectHash}_0 \wedge \neg\mathsf{Bad}] \\ &\geq \Pr[\mathsf{Forge}_0 \wedge \neg\mathsf{Bad}] \Pr[\mathsf{CorrectHash}_0 \mid \mathsf{Forge}_0 \wedge \neg\mathsf{Bad}] \\ &\geq (\Pr[\mathsf{Forge}_0] - \mathsf{negl}(n))/p(n) \qquad (5.1) \end{aligned}$$

Where equation 5.1 follows by the definition of admissible hash functions, and since the choice of $K$ in this experiment is independent of the events $\mathsf{Forge}_0$ and $\mathsf{Bad}$. ■

**Experiment 1:** Modify Experiment 0 as follows. The key-generation algorithm does not sample $pk$ for encryption obliviously. Instead, it uses the value $K$ and samples $(pk, sk_E) \leftarrow \mathsf{KeyGen}_{\mathcal{R}^{\mathsf{AD}}}(1^n, K)$. Note that from the adversary's point of view the initial state remains $\mathsf{state} = \{x, r_s, r_H, pk, \mathsf{crs}\}$. Define the events $\mathsf{Forge}_1$ and $\mathsf{CorrectHash}_1$ analogously to experiment 0. (In the sequel, events with the same name but different subscripts are defined identically, except that they occur in the context of the experiment corresponding to the subscript).

**Claim 5.3.** $\Pr[\mathsf{Forge}_1 \wedge \mathsf{CorrectHash}_1] \geq \Pr[\mathsf{Forge}_0 \wedge \mathsf{CorrectHash}_0] - \mathsf{negl}(n)$.

**Proof.** This follows by the "indistinguishability of initialization values" property of the $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption scheme. That is, even if revealed our choice of $K$ to the adversary, it is impossible to distinguish between the case where $pk$ is chosen obliviously (experiment 0) and where it is chosen using the initialization value $K$ (experiment 1). ■

Define the event $\mathsf{Extract}_1$ to be the event that $\mathsf{Forge}_1$ and $\mathsf{CorrectHash}_1$ hold and, in addition, the signature $\sigma^* = (C^*, \pi^*)$ on the message $m^*$ produced by $\mathcal{A}$ at the conclusion of Experiment 1 is such that $x' = \mathsf{Dec}_{sk_E}^{H(m^*)}(C^*)$ satisfies $\mathsf{F}_s(x') = y$.

**Claim 5.4.** $\Pr[\mathsf{Extract}_1] \geq \Pr[\mathsf{Forge}_1 \wedge \mathsf{CorrectHash}_1] - \mathsf{negl}(n)$.

**Proof.** Follows by the soundness of the SNIWI argument. That is, when $\mathsf{Forge}_1 \wedge \mathsf{CorrectHash}_1$ occurs but $\mathsf{Extract}_1$ does not, the proof $\pi^*$ is that of a false statement. ■

Define the event $\mathsf{SameExtract}_1$ to be the event that $\mathsf{Extract}_1$ occurs and that $x' = x$, where $x' = \mathsf{Dec}_{sk_E}^{H(m^*)}(C^*)$ is the extracted value and $x$ is the secret-key used by the challenger.

**Claim 5.5.** $\Pr[\mathsf{SameExtract}_1] \geq \Pr[\mathsf{Extract}_1] - \mathsf{negl}(n)$.

**Proof.** Follows by the second pre-image resistance (SPR) of $(\mathsf{KeyGen}, \mathsf{F})$. That is, if the probability of $\mathsf{Extract}_1 \wedge \neg\mathsf{SameExtract}_1$ occurring is noticeable, than we can break the SPR-security by using the SPR-challenge $(s, x)$ to run Experiment 1 and recovering $x' \neq x$ such that $\mathsf{F}_s(x) = \mathsf{F}_s(x')$ from the adversary's forgery (with noticeable probability) . ■

We now want to show that, in Experiment 1, the only information that the adversary learns about the secret-key $x$ is from the leakage-queries, while the signature queries do not reveal additional information. To do so, we introduce Experiments 2-5. Since our argument, from now on, is solely information-theoretic the following experiments will no longer be efficient.

**Experiment 2:** Modify Experiment 1 by changing the response of the signing oracle. For any ciphertext-plaintext pair $(C, z)$, define $\mathsf{Rand}_{pk}^{H(m)}(C, z)$ to be the weighted distribution $\{r : C = \mathsf{Enc}_{pk}^{H(m)}(z; r)\}$ of random values which give $C$ as an encryption of $z$. When responding to a signing query $m$, the oracle first generates an encryption $C = \mathsf{Enc}_{pk}^{H(m)}(x; r_E)$ as in the previous experiments. It then samples a new value for the randomness $r'_E \leftarrow \mathsf{Rand}_{pk}^{H(m)}(C, x)$ and uses this value in the place of $r_E$ in the state update process and as the witness for the proof $\pi$. Explicitly, the output signature is $(C, \pi)$, where

$$C = \mathsf{Enc}_{pk}^{H(m)}(x; r_E), \quad \pi = \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C), (x, r'_E); r_\pi),$$

and the state is updated as $\mathsf{state} \leftarrow \mathsf{state} \cup \{r'_E, r_\pi\}$.

Note that, in the description of Experiment 2, the challenger is *no longer efficient*. However, the distribution of this modified experiment is identical to the original; this step merely introduces the randomness $r'_E$ as a function of $C$ and $x$, as opposed to viewing the ciphertext $C$ as a function of $x$ and $r'_E$. Therefore we get the following claim.

**Claim 5.6.** $\Pr[\mathsf{SameExtract}_2] = \Pr[\mathsf{SameExtract}_1]$.

**Experiment 3:** Again modify the response of the signing oracle, this time replacing the encryption $C$ of $x$ in each signature with a new encryption $C'$ of a uniformly chosen preimage $x'$ of $y$ under $\mathsf{F}_s(\cdot)$. Explicitly, for each signature query $m$,

1. Choose $x'$ uniformly at random subject to $\mathsf{F}_s(x') = y$.
2. Sample $r_E \leftarrow \{0,1\}^*$ and compute $C' \leftarrow \mathsf{Enc}_{pk}^{H(m)}(x'; r_E)$.
3. Sample $r'_E \leftarrow \mathsf{Rand}_{pk}^{H(m)}(C', x)$. Note this is with respect to the original $x$, and if $\mathsf{Rand}_{pk}^{H(m)}(C', x) = \emptyset$ then the experiment terminates.
4. Sample $r_\pi \leftarrow \{0,1\}^*$ and compute $\pi' = \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C'), (x, r'_E); r_\pi)$.
5. Output the signature $(C', \pi')$, where

$$C' = \mathsf{Enc}_{pk}^{m}(x'; r_E), \quad \pi' = \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C'), (x, r'_E); r_\pi),$$

and update $\mathsf{state} \leftarrow \mathsf{state} \cup \{r'_E, r_\pi\}$.

**Claim 5.7.** $\Pr[\mathsf{SameExtract}_3] \geq \Pr[\mathsf{SameExtract}_2] - \mathsf{negl}(n)$.

**Proof.** Define $\mathsf{View}_i$ be the view of $\mathcal{A}$ in Experiment $i$, consisting of its random coins and observed values of the verification key, queried signatures, and leakage values.

Let $C \leftarrow \mathsf{Enc}_{pk}^{H(m_i)}(x)$ and $C' \leftarrow \mathsf{Enc}_{pk}^{H(m_i)}(x')$. Recall that $\mathsf{CorrectHash}_2$ implies the adversary's signature queries $m_i$ correspond to lossy tags for the $\mathcal{R}^{\mathsf{AD}}$-lossy scheme. Thus, for each $i$ the distributions $(pk, C)$ and $(pk, C')$ are statistically indistinguishable, say $\delta$ close. In particular, this also implies $\mathsf{Rand}_{pk}^{H(m_i)}(C, x) \neq \emptyset$ with overwhelming probability. This indistinguishability remains true even if the value of $x$ is known.

The remainder of the adversary's view is composed of the verification key, the proofs $\pi_i$ from the queried signatures, and the leakage function evaluations $\mathsf{Leakage} = \bigcup_j f_j(\mathsf{state}_j)$. The verification key $vk = (s, y, H, pk, \mathsf{crs})$ can be computed purely as a function of $x$ and randomness. And, in Experiments 2 and 3, the proofs $\pi_i$ and leakage values are computed as a function of $vk, x$, independent randomness $r_\pi$, and $r'_E$, which in turn is selected as a function of $x$ and the corresponding ciphertext $C$ or $C'$. For each $i$ the joint distribution $(x, C, vk, \pi_i, \mathsf{Leakage})$ must then be $\delta$-close to that of $(x, C', vk, \pi_i, \mathsf{Leakage})$. If the adversary makes $q = \mathsf{poly}(n)$ signature queries, we will then have $(x, \mathsf{View}_2) \approx_{q\delta} (x, \mathsf{View}_3)$ Hence, $\Pr[\mathsf{SameExtract}_2] \leq \Pr[\mathsf{SameExtract}_3] + q\delta$. ∎

**Experiment 4:** Modify the response of the signing oracle by performing steps 1-4 as in Experiment 3, then continuing as follows. Analogous to the distribution that was defined above for sampling $r'_E$ for the $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption scheme, let $\mathsf{Rand}_{H(m)}(\pi, x, r_E)$ be the weighted distribution $\{r : \pi = \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C), (x, r_E); r)\}$.

5. Sample $r'_\pi \leftarrow \mathsf{Rand}_{H(m)}(\pi', x, r'_E)$.

6. Output the signature $(C', \pi')$, where

$$C' = \mathsf{Enc}^m_{pk}(x'; r_E), \quad \pi' = \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C'), (x, r'_E); r_\pi)$$

as before, but update $\mathsf{state} \leftarrow \mathsf{state} \cup \{r'_E, r'_\pi\}$ using $r'_\pi$.

Again, the distribution of experiments 3 and 4 are identical and hence we get the following:

**Claim 5.8.** $\Pr[\mathsf{SameExtract}_4] = \Pr[\mathsf{SameExtract}_3]$.

**Experiment 5:** Modify the response of the signing oracle by replacing the proof with one using witness $(x', r_E)$ instead of $(x, r'_E)$. Explicitly, perform steps 1-3 as in Experiment 3, then continue as follows.

4. Sample $r_\pi \leftarrow \{0,1\}^*$ and compute $\pi'' = \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C'), (x', r_E); r_\pi)$.
5. Sample $r'_\pi \leftarrow \mathsf{Rand}_{H(m)}(\pi'', x, r'_E)$. If $\mathsf{Rand}_{H(m)}(\pi'', x, r'_E) = \emptyset$, the experiment terminates.
6. Output the signature $(C', \pi'')$, where

$$C' = \mathsf{Enc}^m_{pk}(x'; r_E), \quad \pi'' = \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C'), (x', r_E); r_\pi),$$

and update $\mathsf{state} \leftarrow \mathsf{state} \cup \{r'_E, r'_\pi\}$.

**Claim 5.9.** $\Pr[\mathsf{SameExtract}_5] \geq \Pr[\mathsf{SameExtract}_4] - \mathsf{negl}(n)$.

**Proof.** The proof is analogous to that of Claim 5.7. Specifically, let

$$\pi' \leftarrow \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C'), (x, r'_E); r_\pi)$$

$$\pi'' \leftarrow \mathsf{P}(\mathsf{crs}, (s, y, pk, H(m), C'), (x', r_E); r_\pi)\ ,$$

then by the statistical witness indistinguishability of the argument system, the distributions of $\pi'$ and $\pi''$ are $\delta$-close for some negligible $\delta$, even if $x$ is known. In particular, this implies $\mathsf{Rand}_{H(m)}(\pi'', x, r'_E) \neq \emptyset$ with overwhelming probability. Since the leakage is computed on $r'_\pi$, which is selected as a function of the proof and $x$, the joint distributions of

$$(x, vk, C', \pi', \mathsf{Leakage}(x, \pi')) \approx_\delta (x, vk, C', \pi'', \mathsf{Leakage}(x, \pi''))$$

must be $\delta$-close. Thus, if the adversary makes $q$ signature queries, we will have $(x, \mathsf{View}_4) \approx_{q\delta} (x, \mathsf{View}_5)$, and so $\mathsf{View}_4 \approx_{q\delta} \mathsf{View}_5$. ■

As a final step in the proof, we show that $x$ still possesses high average min-entropy conditioned on the view of $\mathcal{A}$ within Experiment 5.

**Claim 5.10.** $\widetilde{\mathrm{H}}_\infty(x|\mathsf{View}_5) \geq n - n^\epsilon - \lambda$.

**Proof.** We consider how the average min-entropy of $x$ decreases as the experiment progresses. At the beginning of the experiment (i.e., before the adversary submits any queries), the view of the adversary consists of the verification key $vk$ and its own random coins $\mathsf{RandCoins}$. The experiment proceeds with a series of queries made by $\mathcal{A}$ to the leakage and signing oracles. Let $\mathsf{View}^{(j)}_5$ denote the view of the adversary within Experiment 5 after $j$ such queries (thus $\mathsf{View}^{(0)}_5 = vk||\mathsf{RandCoins}$, where $||$ denotes concatenation).

For each leakage query $f_j$ with a $\lambda_j$-bit output, it holds that $\mathsf{View}_5^{(j)} = \mathsf{View}_5^{(j-1)} || f_j(\mathsf{state})$, and therefore Lemma 2.1 guarantees that $\widetilde{\mathrm{H}}_\infty\left(x \middle| \mathsf{View}_5^{(j)}\right) \geq \widetilde{\mathrm{H}}_\infty\left(x \middle| \mathsf{View}_5^{(j-1)}\right) - \lambda_j$. For each signature query with a message $m_j$, it holds that $\mathsf{View}_5^{(j)} = \mathsf{View}_5^{(j-1)} || m_j || \sigma_j$. First, note that the message $m_j$ is chosen by the adversary as a function of $\mathsf{View}_5^{(j-1)}$. Second, note that in Experiment 5, each signature $\sigma_j$ is computed as a function of $x'$ which is independent of $x$ given $y$. Therefore, the pair $(m_j, \sigma_j)$ does not reduce the average min-entropy of $x$ given $\mathsf{View}_5^{(j-1)}$ That is, $\widetilde{\mathrm{H}}_\infty\left(x \middle| \mathsf{View}_5^{(j)}\right) = \widetilde{\mathrm{H}}_\infty\left(x \middle| \mathsf{View}_5^{(j-1)}\right)$. Thus, at the conclusion of Experiment 5 it holds that

$$\widetilde{\mathrm{H}}_\infty\left(x | \mathsf{View}_5\right) \geq \widetilde{\mathrm{H}}_\infty\left(\mathsf{View}_5^{(0)}\right) - \sum \lambda_j \geq \widetilde{\mathrm{H}}_\infty\left(x | vk, \mathsf{RandCoins}\right) - \lambda \ .$$

Now, the random coins $\mathsf{RandCoins}$ of the adversary are independent of $x$, and the same holds for all components of the verification key $vk$ except $y = \mathsf{F}_s(x)$. This means $\widetilde{\mathrm{H}}_\infty\left(x | vk, \mathsf{RandCoins}\right) = \widetilde{\mathrm{H}}_\infty\left(x | \mathsf{F}_s(x)\right)$. Finally, since $\mathsf{F}_s(\cdot) : \{0,1\}^n \to \{0,1\}^{n^\epsilon}$, we have $\widetilde{\mathrm{H}}_\infty\left(x | \mathsf{F}_s(x)\right) \geq n - n^\epsilon$ by Lemma 2.1. Hence, it follows that $\widetilde{\mathrm{H}}_\infty\left(x | \mathsf{View}_5\right) \geq \widetilde{\mathrm{H}}_\infty\left(x | \mathsf{F}_s(x)\right) - \lambda \geq n - n^\epsilon - \lambda$. ■

Combining Claims 5.2-5.9 above, if $\Pr[\mathsf{Forge}_0]$ is non-negligible, it must be that $\Pr[\mathsf{SameExtract}_5]$ is also non-negligible. That is, in Experiment 5, $\mathcal{A}$ is able to produce a valid signature $(C^*, \pi^*)$ for which $x = \mathsf{Dec}_{sk_E}^{H(m^*)}(C^*)$ with non-negligible probability. But, such a signature uniquely determines the value of $x$. By Claim 5.10, if $\lambda \leq n - n^\epsilon - \omega(\log n)$, then $x$ still has super-logarithmic average min-entropy given the view of $\mathcal{A}$ in Experiment 5, which means this is not possible even for a computationally unbounded $\mathcal{A}$. Explicitly, from the definition of average min-entropy, $\Pr[\mathsf{SameExtract}_5] \leq 2^{-(n-n^\epsilon-\lambda)} \leq 2^{-\omega(\log n)} = \mathsf{negl}(n)$. Therefore, the event $\mathsf{Forge}_0$ occurs with only a negligible probability, and this concludes the proof of Theorem 5.1. ■

## 6 An Efficient Instantiation based on the Linear Assumption

In this section we show that our construction, which is based on generic cryptographic primitives, can be instantiated based on specific number-theoretic assumptions to yield a rather efficient scheme. That is, we demonstrate that our approach is not only of theoretical interest (due to the argument system for general NP languages), but may also be of practical interest. We follow the approach of Dodis et al. [DHL+10b] who presented rather efficient instantiations of the leakage-resilient signature scheme of Katz and Vaikuntanathan [KV09] using the proof system of Groth and Sahai [GS08]. For our scheme, this means that all of its building blocks have to be instantiated efficiently, and expressed in a form such that the resulting NP language fits the proof system of Groth and Sahai. Here we present an instantiation based on the Linear assumption, and we note that an additional instantiation can be based on the seemingly less standard SXDH assumption as in [DHL+10b]. In what follows we present Linear-based instantiations of a family of SPR functions, and of an $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption scheme. We then briefly describe the proof system of Groth and Sahai [GS08] that we use as a SNIWI argument system.

We note that our construction can be instantiated with any public-coin admissible hash function ensemble. As discussed in Section 2.6, such functions can be constructed based on public-coin collision-resistant hash functions, which in turn can be constructed based on the discrete log assumption in some group $\mathbb{G}$ of prime order $p$ (which follows from the Linear assumption).

### 6.1 A Linear-based Family of SPR Functions

Let $\mathsf{GroupGen}$ be a probabilistic polynomial-time algorithm that takes a security parameter $1^n$ as input and outputs a 5-tuple $(\mathbb{G}, \mathbb{G}_T, p, g, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order $p$, the group $\mathbb{G}$ is generated by $g$, and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear maps. The following family $\mathcal{F} = (\mathsf{KeyGen}, \mathsf{F})$ of functions is based on the SPR relation of Dodis et al. [DHL+10b, Appendix C.2.1].

- **Key generation:** On input $1^n$ the algorithm $\mathsf{KeyGen}$ begins by sampling $(\mathbb{G}, \mathbb{G}_T, p, g, e) \leftarrow \mathsf{GroupGen}(1^n)$. Then, it samples $s = (h_1, \dots, h_{k(n)}, h'_1, \dots, h'_{k(n)}) \leftarrow \mathbb{G}^{2k(n)}$ uniformly at random to be used as a public description of a function $\mathsf{F}(s, \cdot) : \mathbb{G}^{k(n)} \to \mathbb{G}_T^2$.
- **Evaluation:** On input a public description $s$ of the above form and an input $(g_1, \dots, g_{k(n)}) \in \mathbb{G}^{k(n)}$, the algorithm $\mathsf{F}$ outputs

$$\mathsf{F}\left(s, \left(g_1, \dots, g_{k(n)}\right)\right) = \left(\prod_{i=1}^{k(n)} e(h_i, g_i), \prod_{i=1}^{k(n)} e(h'_i, g_i)\right) .$$

The following theorem establishes the SPR security of these functions. The proof is essentially identical to the security proof of the SPR relation of Dodis et al. [DHL+10b, Claim C.2], and is therefore omitted.

**Theorem 6.1.** *Assuming the hardness of the decisional Linear problem, then $\mathcal{F} = (\mathsf{KeyGen}, \mathsf{F})$ is a family of SPR functions.*

## 6.2 A Linear-based $\mathcal{R}^{\mathsf{AD}}$-Lossy Public-Key Encryption Scheme

We present a natural generalization of the scheme described in Section 4.2. Let $\mathsf{GroupGen}$ be a probabilistic polynomial-time algorithm that takes a security parameter $1^n$ as input and outputs a triplet $(\mathbb{G}, p, g)$, where $\mathbb{G}$ is a group of order $p$ generated by $g \in \mathbb{G}$, and $p$ is an $n$-bit prime number. Consider the following encryption scheme $\Pi_{\mathsf{Lin}} = (\mathsf{KeyGen}_{\mathsf{Lin}}, \mathsf{Enc}_{\mathsf{Lin}}, \mathsf{Dec}_{\mathsf{Lin}})$:

- **Key generation:** On input $1^n$ and an initialization value $K = K_1 \cdots K_{\tau(n)} \in \{0, 1, \perp\}^{\tau(n)}$, the algorithm $\mathsf{KeyGen}_{\mathsf{Lin}}$ samples $(\mathbb{G}, p, g) \leftarrow \mathsf{GroupGen}(1^n)$, together with three independently and uniformly chosen elements $g_1, g_2, g_3 \leftarrow \mathbb{G}$. Then, for every $1 \le i \le \tau(n)$ it samples $\alpha_{i,0}, \beta_{i,0}, \gamma_{i,0}, \alpha_{i,1}, \beta_{i,1}, \gamma_{i,1} \leftarrow \mathbb{Z}_p$ uniformly and independently at random, and continues as follows:
  - If $K_i = 0$ then it sets $\gamma_{i,1} = \alpha_{i,1} + \beta_{i,1}$.
  - If $K_i = 1$ then it sets $\gamma_{i,0} = \alpha_{i,0} + \beta_{i,0}$.
  - If $K_i = \perp$ then it sets $\gamma_{i,0} = \alpha_{i,0} + \beta_{i,0}$ and $\gamma_{i,1} = \alpha_{i,1} + \beta_{i,1}$.

  Finally, for every $1 \le i \le \tau(n)$ and $b \in \{0, 1\}$ it sets $(u_{i,b}, v_{i,b}, w_{i,b}) = \left(g_1^{\alpha_{i,b}}, g_2^{\beta_{i,b}}, g_3^{\gamma_{i,b}}\right)$, and outputs the pair $(sk, pk)$ defined as

$$sk = \left(K, \left\{\left(\alpha_{i,0}, \beta_{i,0}\right), \left(\alpha_{i,1}, \beta_{i,1}\right)\right\}_{i=1}^{\tau(n)}\right)$$

$$pk = \left(g_1, g_2, g_3, \left\{\left(u_{i,0}, v_{i,0}, w_{i,0}\right), \left(u_{i,1}, v_{i,1}, w_{i,1}\right)\right\}_{i=1}^{\tau(n)}\right)$$

- **Encryption:** On input a public key $pk$ of the above form, a tag $t = t_1 \cdots t_{\tau(n)} \in \{0, 1\}^{\tau(n)}$ and a message $m \in \mathbb{G}$, the algorithm $\mathsf{Enc}_{\mathsf{Lin}}$ chooses $r, r', r'' \in \mathbb{Z}_p$ uniformly and independently at random, computes

$$u_t = \prod_{i=1}^{\tau(n)} u_{i,t_i} \ , \quad v_t = \prod_{i=1}^{\tau(n)} v_{i,t_i} \ , \quad w_t = \prod_{i=1}^{\tau(n)} w_{i,t_i} \ ,$$

$$c_1 = g_1^r g_3^{r''} \quad , \quad c_2 = g_2^{r'} g_3^{r''} \quad , \quad c_3 = (u_t)^r (v_t)^{r'} (w_t)^{r''} \cdot m \enspace ,$$

and outputs the ciphertext $(c_1, c_2, c_3)$.

- **Decryption:** On input a secret key $sk$ of the above form, a tag $t = t_1 \cdots t_{\tau(n)} \in \{0,1\}^{\tau(n)}$ and a ciphertext $c = (c_1, c_2, c_3)$, the algorithm $\mathsf{Dec}_{\mathsf{Lin}}$ proceeds as follows. If $(K, t) \notin \mathcal{R}^{\mathsf{AD}}$ (i.e., $t$ is a lossy tag) then it outputs $\perp$. Otherwise (i.e., $t$ is an injective tag), it outputs the message $m$ defined as
$$m = c_3 \cdot \left( c_1^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} c_2^{\sum_{i=1}^{\tau(n)} \beta_{i,t_i}} \right)^{-1} \enspace .$$

**Theorem 6.2.** *Assuming the hardness of the decisional Linear problem, then* $\Pi_{\mathsf{Lin}} = (\mathsf{KeyGen}_{\mathsf{Lin}}, \mathsf{Enc}_{\mathsf{Lin}}, \mathsf{Dec}_{\mathsf{Lin}})$ *is an almost-always $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption scheme for tags of length $\tau(n) \leq \log p - \omega(\log n)$.*

**Proof.** Indistinguishability of initialization values follows directly from the hardness of the decisional Linear problem and a standard hybrid argument. In addition, (perfect) correctness of the decryption algorithm under injective tags follows from the fact that for any injective tag $t = t_1 \cdots t_{\tau(n)}$ (i.e., for every $i \in \{1, \ldots, \tau(n)\}$ it holds that $K_i \neq t_i$) it holds that $\gamma_{i,t_i} = \alpha_{i,t_i} + \beta_{i,t_i}$ for every $i \in \{1, \ldots, \tau(n)\}$. Therefore, for any ciphertext $(c_1, c_2, c_3)$ that is produced by encrypting a message $m$ under an injective tag $t$ it holds that

$$\begin{aligned}
& c_3 \cdot \left( c_1^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} c_2^{\sum_{i=1}^{\tau(n)} \beta_{i,t_i}} \right)^{-1} \\
&= (u_t)^r (v_t)^{r'} (w_t)^{r''} \cdot m \cdot \left( \left( g_1^r g_3^{r''} \right)^{\sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} \left( g_2^{r'} g_3^{r''} \right)^{\sum_{i=1}^{\tau(n)} \beta_{i,t_i}} \right)^{-1} \\
&= \left( \prod_{i=1}^{\tau(n)} u_{i,t_i} \right)^r \left( \prod_{i=1}^{\tau(n)} v_{i,t_i} \right)^{r'} \left( \prod_{i=1}^{\tau(n)} w_{i,t_i} \right)^{r''} \cdot m \cdot \left( g_1^{r \sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} g_2^{r' \sum_{i=1}^{\tau(n)} \beta_{i,t_i}} g_3^{r'' \sum_{i=1}^{\tau(n)} (\alpha_{i,t_i} + \beta_{i,t_i})} \right)^{-1} \\
&= \left( g_1^{r \sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} g_2^{r' \sum_{i=1}^{\tau(n)} \beta_{i,t_i}} g_3^{r'' \sum_{i=1}^{\tau(n)} \gamma_{i,t_i}} \right) \cdot m \cdot \left( g_1^{r \sum_{i=1}^{\tau(n)} \alpha_{i,t_i}} g_2^{r' \sum_{i=1}^{\tau(n)} \beta_{i,t_i}} g_3^{r'' \sum_{i=1}^{\tau(n)} (\alpha_{i,t_i} + \beta_{i,t_i})} \right)^{-1} \\
&= m \enspace .
\end{aligned}$$

Finally, we prove that lossiness under lossy tags holds with probability at least $1 - 2^{\tau(n)}/p$ over the internal randomness of the key-generation algorithm. Fix an initialization value $K \in \{0, 1, \perp\}^{\tau(n)}$, and a lossy tag $t \in \{0,1\}^{\tau(n)}$ (i.e., there exists some $i \in \{1, \ldots, \tau(n)\}$ for which $K_i = t_i$). Then, there exists some $i \in \{1, \ldots, \tau(n)\}$ for which the values $\alpha_{i,t_i}$, $\beta_{i,t_i}$, and $\gamma_{i,t_i}$ are uniformly and *independently* chosen, and therefore the values $u_t$, $v_t$, and $w_t$ as defined by the encryption algorithm are independently and uniformly distributed in $\mathbb{G}$. Thus, with probability $1 - 1/p$, it holds that $(g_1, g_2, g_3, u_t, v_t, w_t)$ is not an instance of the Linear problem (i.e., $\log_{g_3}(w_t) \neq \log_{g_1}(u_t) + \log_{g_2}(w_t)$). In this case, the triplet $(g_1^r g_3^{r''}, g_2^{r'} g_3^{r''}, (u_t)^r (v_t)^{r'} (w_t)^{r''})$ is uniformly distributed in $\mathbb{G}^3$ over the choice of $r$, $r'$, and $r''$ (this is a natural generalization of [PVW08, Lemma 4]). This implies that a ciphertext $(c_1, c_2, c_3)$ encrypted under a lossy tag $t$ carries no information on the message. This holds for any specific lossy tag $t$, and therefore a union bound guarantees that this holds for all lossy tags with probability at least $1 - 2^{\tau(n)}/p$. ∎

## 6.3 A Linear-based SNIWI Argument System

In this section we briefly review the proof system of Groth and Sahai [GS08] for proving that a system of equations is satisfiable. We mainly follow the exposition of Dodis et al. [DHL$^+$10b], and refer the reader to [GS08] for more details. We note that in the Groth-Sahai proof system, there are two computationally indistinguishable methods for generating the common reference string: one (called real) that yields perfect soundness, and another (called simulated) that yields perfect witness indistinguishability. By using the simulated common reference string we can thus use their system as a SNIWI argument system. We consider here the cases of one-sided multi-exponentiation equations and one-sided pairing-product equations, as these are the cases that arise from our Linear-based constructions in Sections 6.1 and 6.2.

**The CRS-generation algorithm.** On input $1^n$ the algorithm $\mathsf{CRSGen}$ samples $(\mathbb{G}, \mathbb{G}_T, p, g, e) \leftarrow \mathsf{GroupGen}(1^n)$, together with 3 independently and uniformly distributed elements $u_0, u_1, u_2 \leftarrow \mathbb{G}$, and sets

$$\begin{aligned} \vec{u}_1 &= (u_0, u_1, 1) \\ \vec{u}_2 &= (u_0, 1, u_2) \ . \end{aligned}$$

Then, it samples $\vec{u}_0 \leftarrow \mathbb{G}^3 \backslash \mathfrak{U}$ and $\vec{u} \leftarrow \mathfrak{U}$ independently and uniformly at random, where $\mathfrak{U} = \left\{ \left( u_0^{\alpha+\beta}, u_1^{\alpha}, u_2^{\beta} \right) : \alpha, \beta \in \mathbb{Z}_p \right\}$. It outputs $\mathsf{crs} = (\vec{u}_0, \vec{u}_1, \vec{u}_2, \vec{u})$.

**Dealing with one-sided multi-exponentiation equations.** For every equation of the form

$$\prod_{i=1}^{k} g_i^{\chi_i} = g_0 \ ,$$

where $g_0, g_1, \ldots, g_k \in \mathbb{G}$ are constants, and $\chi_1, \ldots, \chi_k \in \mathbb{Z}_p$ are variables (i.e., the $\chi_i$'s are the satisfying assignment), the prover begins by committing to each of the $\chi_i$'s. The commitment to each $\chi_i \in \mathbb{Z}_p$ is defined as $\vec{\gamma}_i = \vec{u}^{\chi_i} \prod_{j=1}^{2} \vec{u}_j^{\,t_i^{(j)}}$, where $\vec{t_i} = \left( t_i^{(1)}, t_i^{(2)} \right) \leftarrow \mathbb{Z}_p^2$ is sampled uniformly at random. Then, the prover computes the group elements $p_1$ and $p_2$ that are defined as

$$p_j = \prod_{i=1}^{k} g_i^{t_i^{(j)}} \quad , \quad j = 1, 2.$$

In turn, the verifier accepts if and only if for every such equation it holds that

$$\prod_{i=1}^{k} E(\vec{\gamma}_i, g_i) = E(\vec{u}, g_0) \prod_{j=1}^{2} E(\vec{u}_j, p_j) \ ,$$

where $E : \mathbb{G}^3 \times \mathbb{G} \to \mathbb{G}_T^3$ is defined as $E((\alpha_0, \alpha_1, \alpha_2), \beta) = (e(\alpha_0, \beta), e(\alpha_1, \beta), e(\alpha_2, \beta))$. Note that for a set of $r$ such equations with a witness of size $k$, the proof consists of $3k + 2r$ group elements.

**Dealing with one-sided pairing-product equations.** For every equation of the form

$$\prod_{i=1}^{k} e(h_i, x_i) = T \ ,$$

where $h_1, \ldots, h_k \in \mathbb{G}$ and $T \in \mathbb{G}_T$ are constants, and $x_1, \ldots, x_k \in \mathbb{G}$ are variables (i.e., the $x_i$'s are the satisfying assignment), the prover begins by committing to each of the $x_i$'s. The commitment to each $x_i \in \mathbb{G}$ is defined as $\vec{\delta}_i = (x_i, 1, 1) \prod_{j=0}^{2} \vec{u}_j^{\,s_i^{(j)}}$, where $\vec{s_i} = \left(s_i^{(0)}, s_i^{(1)}, s_i^{(2)}\right) \leftarrow \mathbb{Z}_p^3$ is sampled uniformly at random, and vector multiplication is defined component-wise. Then, the prover computes the group elements $p_0$, $p_1$, and $p_2$ that are defined as

$$p_j = \prod_{i=1}^{k} h_i^{s_i^{(j)}} \quad , \quad j = 0, 1, 2.$$

In turn, the verifier accepts if and only if for every such equation it holds that

$$\prod_{i=1}^{k} E(h_i, \vec{\delta}_i) = (T, 1, 1) \prod_{j=0}^{2} E(p_j, \vec{u}_j) \ ,$$

where $E : \mathbb{G} \times \mathbb{G}^3 \to \mathbb{G}_T^3$ is defined as $E(\alpha, (\beta_0, \beta_1, \beta_2)) = (e(\alpha, \beta_0), e(\alpha, \beta_1), e(\alpha, \beta_2))$. Note that for a set of $r$ such equations with a witness of size $k$, the proof consists of $3(k + r)$ group elements.

## 7 A Signature Scheme in the Continual-Leakage Model

In this section, we extend our approach to the continual-leakage model. In order to do this, in Section 7.1 we first introduce an alternate and more general measure of leakage called "entropy leakage." Instead of measuring leakage in terms of the output length of a leakage function, we look at the entropy loss that such a function causes to a random input. In Section 7.2, we offer a generalized explanation of our scheme in the bounded-leakage model as a construction of leakage-resilient signatures from *leakage-resilient one-way functions* (LR-OWF). This explanation is only meant to build intuition for our construction in the continual setting, and hence the exposition will be rather informal. Finally, in Section 7.3, we show that our construction generalizes to constructing fully leakage-resilient signatures in the continual-leakage model from *continuous-leakage-resilient one-way relations* (in the entropy leakage sense), whose instantiations were given in [DHL$^+$10a, BTK$^+$10].

### 7.1 Entropy Leakage

So far, we have measured the *amount* of leakage that the adversary learns from a function $f$ via the output length of $f$. We call this *length-bounded leakage.* However, this is not the most general way of measuring leakage. As an alternative, we could consider measuring the amount of leakage via the *entropy loss* to the input of $f$, given the output of $f$. In particular, we want our definitions to allow long leakage as long as it does not reveal too much *useful* information. The idea of measuring the entropy loss of a leakage function is due to Naor and Segev [NS09], but our definition is closer to that of Dodis et al. [DHL$^+$10a]. In particular, we measure the amount of leakage learned by a function $f : \{0,1\}^* \to \{0,1\}^*$ in terms of the amount of entropy that reduces from the *uniform distribution.* As shown in [DHL$^+$10a], this definition has some nice composability properties.

**Definition 7.1** ($\lambda$-*entropy leaky* functions)**.** *A (possibly randomized) efficiently computable function $f : \{0,1\}^* \to \{0,1\}^*$ is* $\lambda$-entropy leaky *if there exists some (possibly inefficiently computable) function $f'$ such that:*

- *For all $x \in \{0,1\}^*$, $f(x) \approx_s f'(x)$ (over the randomness of $f$ and $f'$).*

- *For all integers $n \geq 1$, $\widetilde{\mathrm{H}}_\infty(U_n \mid f'(U_n)) \geq n - \lambda$ where $U_n$ is the uniform distribution over $\{0,1\}^n$.*

Notice that any function $f : \{0,1\}^* \to \{0,1\}^\lambda$ is $\lambda$-entropy leaky. However, there are clearly functions which are $\lambda$-entropy leaky, but whose output lengths can be arbitrarily long. Therefore, resilience to $\lambda$ bits of entropy leakage is a seemingly stronger notion of security than resilience to $\lambda$ bits of length-bounded leakage (see also the discussion in Section 8). Moreover, this definition turns out to be more robust in allowing us to reduce the leakage-resilient security of a signature scheme to that of a leakage-resilient one-way function, which we do next.

## 7.2 A Generalized Explanation: FLR Signatures from LR-OWFs

Recall that our original signature scheme is based on a second-preimage resistant (SPR) function $(\mathsf{KeyGen}, \mathsf{F})$. The verification key contains an image $y = \mathsf{F}_s(x)$ of the secret key, and each signature is essentially a proof that the signer knows a preimage of $y$ under $\mathsf{F}_s$. The important property of the SPR family is that one can easily evaluate $F_s(x)$ given $x$ (to generate the verification key), but an adversary cannot extract a preimage of $y$, even given "leakage" information on $x$ learned via leakage and signature queries during the security experiment. It turns out this is the only property we need. In what follows, we abstract out the role of the SPR family in our construction to something satisfying the more general notion of a *leakage-resilient one-way function*, for a class of leakage that captures these oracle responses. As this section is only meant to provide intuition, the discussion will be somewhat informal.

**LR one-way functions.** We define the notion of a *leakage-resilient one-way function* (LR-OWF) $\mathsf{F}$ in the following way. Given the image $y = \mathsf{F}(x)$ of a random value $x$ in the domain, together with $\lambda$ bits of leakage on $x$, it is computationally hard to produce any preimage $x'$ of $y$ under $\mathsf{F}$. We can define this notion with respect to either length-bounded leakage or entropy-bounded leakage (which is more general)

**Definition 7.2** (LR-OWF). *A collection* $(\mathsf{KeyGen}, \mathsf{F})$ *is a* $\lambda$-leakage-resilient one-way function (LR-OWF) *with respect to* entropy-bounded leakage *(respectively,* length-bounded leakage*) if for any probabilistic polynomial-time algorithm $\mathcal{A}$ and efficiently computable function $g$ which is $\lambda$-entropy-leaky (respectively, has $\lambda$-bit outputs) there exists a negligible function $\nu(\cdot)$ such that*

$$\Pr[\mathsf{F}_s(x') = y \mid s \leftarrow \mathsf{KeyGen}(1^n), x \leftarrow \{0,1\}^n, y = \mathsf{F}_s(x), x' \leftarrow \mathcal{A}(s, y, g(x))] \leq \nu(n).$$

It is easy to show (as is implicity in [ADW09, KV09]) that a second-preimage resistant (SPR) function with $m(n)$-bit inputs and $m'(n)$-bit outputs is also a $\lambda$-leakage-resilient one-way function for $\lambda(n) = m(n) - m'(n) - \omega(\log n)$. Essentially, this is because a random $x$ has entropy even given $y = \mathsf{F}_s(x)$ and some $\lambda$ bits of leakage on $x$. Therefore, if there exists an algorithm that is able to produce an $x'$ such that $\mathsf{F}_s(x') = y$ given only this information, then with overwhelming probability $x' \neq x$, and we can use this adversary to break the second-preimage resistance. For the above argument, it does not matter if we consider entropy-bounded or leakage length-bounded leakage. Therefore, we get the following observation.

**Observation 7.3.** *An second-preimage resistant function* $(\mathsf{KeyGen}, \mathsf{F}$*) with $m(n)$-bit inputs and $m'(n)$-bit outputs is a LR-OWF with respect to $\lambda(n)$-entropy-bounded leakage, where $\lambda(n) = m(n) - m'(n) - \omega(\log n)$.*

**FLR signatures from LR-OWFs.** We now informally explain our signature scheme from the bounded-leakage model (Section 5) in a slightly more general manner. Recall that our construction was based on an SPR function $(\mathsf{KeyGen}, \mathsf{F})$, where $y = \mathsf{F}_s(x)$ was in the verification key and $x$ was the secret key. To sign a message $m$, we used a $\mathcal{R}^{\mathsf{AD}}$-lossy encryption scheme to encrypt $x$ under label $m$ and proved that the ciphertext is formed correctly using a SNIWI argument system. In general, we do not need $(\mathsf{KeyGen}, \mathsf{F})$ to be an SPR function, but rather any LR-OWF with respect to $\lambda$-entropy-bounded leakage.

**Observation 7.4.** *The construction from Section 5 is a $\lambda$-FLR signature scheme in the bounded-leakage model, when instantiated with* any *LR-OWF with respect to $\lambda$-entropy-bounded leakage.*

The original proof of security (Theorem 5.1) extends naturally to this more general case. To give the main intuition, let us look at the proof. Once we move from Experiment 0 to Experiment 1, there is a good chance of all the signing queries being "lossy" and the forgery still decrypting to some $x^*$ with $\mathsf{F}_s(x^*) = \mathsf{F}_s(x) = y$, where $y$ is in the verification key and $x$ is in the secret key. The main idea of the rest of the proof is that, if this is the case, then the entire view of the adversary (the leakage-queries and the signing queries) does not reduce the entropy of $x$ significantly. Therefore, we can think of the view of the adversary as being "entropy-bounded" leakage (but not "length-bounded" leakage, since we do not know how to efficiently compress it). A successful forgery in this setting means the adversary only receives entropy-bounded leakage on $x$ but still manages to produce (an encryption of) some pre-image $x^*$, thus breaking one-way security.

Although the above generalization does not seem significant at first, this view of our basic construction will make it easier to extend to the setting of continuous leakage, which we will do in the next section.

## 7.3 Extension to Continuous Leakage

We begin by generalizing the concept of a leakage-resilient one-way function to the continual leakage setting, following [DHL$^+$10a].

First, we relax the requirement that $x$ is uniformly random and $y = \mathsf{F}(x)$ is a deterministic function of $x$. Instead, we define a one-way *relation* $(\mathsf{KeyGen}, \mathsf{R})$, where the $\mathsf{KeyGen}$ algorithm generates pairs $(y, x) \in \mathsf{R}$ simultaneously using internal randomness. The security property is similar to that of a one-way function: given a randomly generated $y$ and leakage on $x$, it should be hard to find $x'$ such that $(y, x') \in \mathsf{R}$.[16] One can keep in mind an example where $x$ and $y$ are a secret key and public key for a cryptographic system, and $(y, x) \in \mathsf{R}$ when $x$ is a proper secret key corresponding to $y$.

Second, we add an algorithm $\mathsf{Refresh}$ that allows us to refresh the secret $x$. That is, if $(y, x) \in \mathsf{R}$ and $x' \leftarrow \mathsf{Refresh}(x)$ then $(y, x') \in \mathsf{R}$. The main goal of the refresh algorithm is to allow for continual leakage on the secret key. The user starts off with a key $x_0$ produced by the key-generation algorithm and periodically updates it by running $x_i \leftarrow \mathsf{Refresh}(x_{i-1})$. The adversary can continually receive partial leakage on *every* version $x_i$ of the key that the user ever creates (so that the total amount of leakage learned is unbounded). Nevertheless, at no point in this process should the adversary be able to come up with some $x'$ such that $(y, x') \in \mathsf{R}$.

Third, we consider "entropy-bounded" rather than "length-bounded" leakage. The formal definition appears below.

[16] In the leak-free setting, any one-way relation $(\mathsf{KeyGen}, \mathsf{R})$ can easily be turned into a one-way function $y = \mathsf{KeyGen}(r)$ mapping the random-coins of $\mathsf{KeyGen}$ to the value $y$ it produces. In the setting of leakage, this transformation no longer holds since leakage on $r$ gives more information than leakage on $x$.

**Definition 7.5** (CLR-OWR : Similar to [DHL$^+$10a]). *A continuous-leakage-resilient one-way relation (CLR-OWR) consists of three poly-time procedures* $(\mathsf{KeyGen}, \mathsf{Refresh}, \mathsf{R})$ *with syntax:*

- **Key Generation:** $\mathsf{KeyGen}(1^n)$ *outputs a public key* $y$ *and a secret key* $x$.
- **Key Refreshing:** $\mathsf{Refresh}_y(x)$ *outputs a refreshed secret key* $x'$.
- **Relation Testing:** $\mathsf{R}(y, x)$ *outputs 1 if and only if the pair* $(y, x)$ *is "valid" and satisfies the relation.*

*We say that the relation is* continuous leakage resilient with respect to $\lambda$-entropy-bounded leakage *if it satisfies the following properties:*

**Correctness:** *For any polynomial* $q = q(n)$, *if we sample* $(y, x) \leftarrow \mathsf{KeyGen}(1^n)$,
$x_1 \leftarrow \mathsf{Refresh}_y(x), \ldots, x_q \leftarrow \mathsf{Refresh}_y(x_{q-1})$, *then, with overwhelming probability,*
$\mathsf{R}(y, x) = \mathsf{R}(y, x_1) = \ldots = \mathsf{R}(y, x_q) = 1$.

**Security:** *For any PPT adversary* $\mathcal{A}$, *we have* $\Pr[\mathcal{A} \text{ wins }] \leq \mathsf{negl}(n)$ *in the following game:*

- *The challenger chooses* $(y, x) \leftarrow \mathsf{KeyGen}(1^n)$ *and gives* $y$ *to* $\mathcal{A}$.
- *The adversary* $\mathcal{A}$ *runs for arbitrarily many* leakage rounds $i = 1, 2, \ldots$ *In each round, the adversary chooses a leakage-function* $f_i : \{0,1\}^* \rightarrow \{0,1\}^*$ *and learns* $f_i(x_i)$. *The next-round secret key is sampled as* $x_{i+1} \leftarrow \mathsf{Refresh}_y(x_i)$.
- *The adversary* wins *if at some point it produces a value* $x^*$ *such that* $\mathsf{R}(y, x^*) = 1$ and *each of the leakage-functions* $f_i$ *are* $\lambda$-*entropy-leaky.*

The results of [DHL$^+$10a, BTK$^+$10] show how to construct a CLR-OWR under the linear assumption in bilinear groups.[17] We state the results of those works as follows:[18]

**Claim 7.6** ([DHL$^+$10a, BTK$^+$10]). *For any* $\epsilon > 0$, *there exist* $\lambda$-*CLR-OWR schemes with relative leakage (ratio of leakage to secret key size) given by* $\lambda/|sk| = \frac{1}{2} - \epsilon$ *under the linear assumption in bilinear groups, and* $\lambda/|sk| = 1 - \epsilon$ *under the SXDH assumption.*

Given a CLR-OWR $(\mathsf{KeyGen}_{\mathsf{OWR}}, \mathsf{Refresh}, \mathsf{R})$, we can naturally generalize our construction of signatures from Section 5. Namely, the CLR-OWR will take the place of the second-preimage resistant function in the original construction, analogous to the use of the bounded-leakage-resilient one-way function in the scheme described in the previous subsection. Explicitly, let $(\mathsf{KeyGen}_{\mathcal{R}^{\mathsf{AD}}}, \mathsf{Enc}, \mathsf{Dec})$ be an $\mathcal{R}^{\mathsf{AD}}$-lossy public-key encryption scheme, and let $(\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ be a SNIWI argument system for the language

$$L = \{(s, y, pk, t, C) : \exists x, \omega \text{ st } C = \mathsf{Enc}^t_{pk}(x; \omega) \text{ and } \mathsf{R}(y, x) = 1\}.$$

Consider the following signature scheme $(\mathsf{KeyGen}, \mathsf{Refresh}, \mathsf{Sign}, \mathsf{Verify})$:

- **Key Generation:** Sample $(pk, \cdot) \leftarrow \mathsf{KeyGen}_{\mathcal{R}^{\mathsf{AD}}}(1^n)$, $(y, x) \leftarrow \mathsf{KeyGen}_{\mathsf{OWR}}(1^n)$ and $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^n)$. Output $vk = (\mathsf{crs}, pk, y)$ and $sk = x$.
- **Key Refreshing:** Use the $\mathsf{Refresh}$ procedure of the CLR-OWR to compute $x' \leftarrow \mathsf{Refresh}_y(x)$.

[17] Although the main definitions of those works are with respect to length-bounded leakage, they easily extend to entropy-leakage.

[18] The stated optimized parameters were obtained by [BTK$^+$10], improving those of [DHL$^+$10a].

- **Signing:** On input message $m$, the algorithm $\mathsf{Sign}$ computes an encryption $C = \mathsf{Enc}^m_{pk}(x; \omega)$ of $x$ under the tag $m$ using fresh randomness $\omega$. Then, it invokes the prover of the SNIWI argument system to obtain a proof $\pi \leftarrow \mathsf{P}(\mathsf{crs}, (y, pk, m, C), (x, \omega))$, and outputs the signature $(C, \pi)$.

- **Verifying:** On input message $m$ and signature $\sigma = (C, \pi)$, the algorithm $\mathsf{Verify}$ invokes the verifier of the SNIWI argument system and outputs 1 if and only if $\mathsf{V}(\mathsf{crs}, (y, pk, m, C), \pi) = 1$.

**Theorem 7.7.** *Assume that, in the above construction, the relation is a $(\lambda(n)+1)$-CLR-OWR, the encryption is $\mathcal{R}^{\mathsf{AD}}$-lossy, and the argument system is a SNIWI. Then the above signature scheme is $\lambda(n)$-leakage-resilient in the continual leakage model.*

**Proof.** The structure of the proof is essentially identical to the proof of Theorem 5.1. We will refer to the prior proof liberally, with a focus on the main differences.

Assume that an adversary $\mathcal{A}$ breaks the signature security with a noticeable probability.

We define Experiment 0 to be the continual fully leakage-resilient security game for signatures (see Definition 3.2) and Experiment 1 to be the modified game where the $\mathcal{R}^{\mathsf{AD}}$-lossy encryption scheme is initialized with an initialization value $K \leftarrow \mathcal{K}_{u,n}$ for an appropriate $u$ corresponding to the number of signature queries $q$ that the attacker makes. This mirrors Experiments 0 and 1 in the proof of Theorem 5.1, and the same proof shows that these experiments are indistinguishable. We can define the event $\mathsf{Extract}_1$ to occur when:

- All of the signing queries fall into the "blue"/lossy set and the forgery message falls into the "red"/injective set as defined by the initialization value $K$. (This corresponds to the event $\mathsf{CorrectHash}_1$ defined in the proof of Theorem 5.1.)

- The ciphertext portion of the forgery decrypts to a valid $x^*$ for which $\mathsf{R}(y, x^*) = 1$.

As in the proof of Theorem 5.1 (specifically, Claim 5.4), the event $\mathsf{Extract}_1$ occurs with a noticeable probability. We now show how to use an adversary $\mathcal{A}$ from Experiment 1 to break the security of the CLR-OWR. Our reduction $\mathcal{B}$ samples the $\mathsf{crs}$ of the NIZK and the public/secret key $(pk, sk)$ of the encryption scheme, as in Experiment 1. Recall that the adversary $\mathcal{A}$ expects to run in many epochs (periods between issuing a key refresh query). The view of $\mathcal{A}$ during each epoch $i$ consists of his random coins together with the signing queries and leakage queries issued during that epoch. The main idea is that the reduction $\mathcal{B}$ can simulate this view for $\mathcal{A}$ by learning a single leakage-function $g_i$ on the secret key $x_i$ of the CLR-OWR in each epoch. The selection of $g_i$ (described below) will ensure that:

1. The simulation perfectly matches Experiment 1. In particular, the event $\mathsf{Extract}_1$ occurs with polynomial probability.

2. If the event $\mathsf{Extract}_1$ occurs, then every function $g_i$ queried by $\mathcal{B}$ is at most $(\lambda + 1)$-entropy-leaky.

When the event $\mathsf{Extract}_1$ occurs, then $\mathcal{B}$ can decrypt the ciphertext portion of $\mathcal{A}$'s forgery to some correct $x^*$ such that $\mathsf{R}(y, x^*) = 1$, and win the CLR-OWR security game. Therefore the above two requirements ensure that this occurs with polynomial probability, which leads to a contradiction.

We are left to describe how $\mathcal{B}$ chooses the leakage functions so as to satisfy conditions (1) and (2). In epoch $i$, the function $g_i : \{0,1\}^* \rightarrow \{0,1\}^*$ includes, in its description, the entire view (including the random coins) of the adversary $\mathcal{A}$ up to the start of epoch $i$, along with the verification key $vk = (\mathsf{crs}, pk, y)$ of the signature scheme. The function $g_i(x_i)$ first checks whether

$\mathsf{R}(y, x_i) = 1$ and, if not, returns a 0. Otherwise, it internally runs the code of $\mathcal{A}$ for that epoch, and uses the current secret key $x_i$ (and internal random coins) to answer the leakage queries and the signing queries. The output of $g_i$ consists of all the answers to the various queries asked by $\mathcal{A}$ during the epoch.

It is easy to see that this leakage can be used by $\mathcal{B}$ to (perfectly) simulate the epoch to $\mathcal{A}$, so we satisfy requirement (1).

For requirement (2), note that the *output length* of $g_i$ is long (possibly much longer than the secret key), since it includes the queried signatures. However, when the event $\mathsf{Extract}_1$ occurs, all the signing queries correspond to lossy tags of the encryption scheme, and hence do not reveal information about $x$. In particular, we can define an (inefficient) leakage function $g_i'$ so that (for fixed $x$), $g_i(x) \approx_s g_i'(x)$ are statistically close, and the signature portion of $g_i'(x)$ perfectly hides $x$ given $y$. This function $g_i'$ precisely corresponds to the (inefficiently) generated responses to signature and leakage queries within Experiment 5 of the proof of Theorem 5.1. As shown in the proof of Theorem 5.1, the only entropy loss induced by this $g_i$ given $y$ is due to the output corresponding to $\mathcal{A}$'s leakage queries, and *not* to the signature queries. Our $g_i$ reveals 1 extra bit on a uniform string, corresponding to whether the string is of a form $x$ s.t. $\mathsf{R}(y, x) = 1$ for the fixed, hard-coded $y$. Therefore, since $\mathcal{A}$'s leakage queries were limited to being $\lambda$-entropy leaky, when $\mathsf{Extract}_1$ occurs, the function $g_i$ is $(\lambda + 1)$-entropy leaky, proving (2).

■

## 7.4 Comparison to Scheme of Brakerski et al. [BTK+10]

Our construction in the continual-leakage model shares some similarities with the scheme presented by Brakerski et al. [BTK+10] in the random-oracle model. But, as we now discuss, the two are conceptually quite different. A signature in the Brakerski et al. scheme is composed of two parts: a form of "encryption" of the signing key using lossy trapdoor functions (LTDFs), and a short non-interactive argument that the encryption is formed correctly. The "encryption" portion of their signature is formed by applying message-dependent branches of a LTDF to secret shares of the signing key. Brakerski et al. prove that if the signing key is updated every few signatures, then this signature scheme satisfies a weak notion of unforgeability, where the adversary is required to specify the target forgery message prior to learning the verification key. They then use two known transformations (see [KR00, HW09]) to convert the scheme into one that is unforgeable in the more standard sense.

In the Brakerski et al. scheme, each signature leaks information about the signing key regardless of whether the adversary makes leakage queries. Each LTDF encryption as presented reveals an amount of information equal to the lossy parameter of the LTDF. In addition, each non-interactive proof is treated entirely as leakage—i.e., revealing information on the signing key equal to its full length. This framework of leaking information in each signature has some unfortunate consequences. In order to keep the amount of leakage per signature small enough to maintain security, their construction requires short-length proofs, which are only known to exist within the random-oracle model [Kil92, Mic00, BG08]. In addition, since each signature reveals new information on the current signing key, the signer *must* update the signing key after every couple signatures, even if no side-channel leakage occurs. This means their construction does not yield a scheme in the bounded-leakage model where key refreshing is not a standard part of the model, and that execution within the continual-leakage model is rather inefficient. Requiring frequent key refreshes thus puts a strong restriction on the model, as the key-refreshing operation must be performed in a secure, nearly leakage-free environment.

Using the generic signature transformations to go from weak unforgeability to standard unforgeability also induces a drop in the efficiency of their scheme. For instance, one transformation requires signing each prefix of the original message, thus growing the overall signature size by a factor of the message length [HW09].

Our construction avoids these practical issues through two main technical differences. First, our combination of $\mathcal{R}$-lossy encryption with a SNIWI argument system (in the standard model) yields signatures which statistically reveals *no information* about the signing key (see the related discussion in Section 8). Second, we are able to bypass the generic signature transformations in an efficient fashion by extending the technique of admissible hash functions [BB04]. Essentially, the two transformations used by Brakerski et al. are replaced by simply hashing the message with a special hash function before signing. Using these new techniques, we are able to construct a scheme that is more efficient, no longer relies on the random-oracle model, and can withstand a greater fraction of leakage ($(1 - o(1))L$ as opposed to $(1/2 - o(1))L$ based on the Linear assumption).

## 8 Concluding Remarks and Open Problems

**Deterministic leakage-resilient signatures.** An alternative approach for constructing fully leakage-resilient signature schemes is constructing a signature scheme that is resilient to leakage from the signing key, and has a deterministic signing algorithm (this is indeed the idea underlying the fully leakage-resilient *one-time* signature schemes of Katz and Vaikuntanathan [KV09]). In general, the signing algorithm of any signature scheme can be made deterministic by using as its random coins the output of a pseudorandom function applied to the message. This requires, however, that the signing key will include also the key of the pseudorandom function, and therefore it is not clear that such a transformation can preserve leakage resilience.

**Bounded leakage vs. noisy leakage.** In some scenarios it is not always possible to assume that the total amount of leakage is upper bounded by $\lambda$ bits, where $\lambda$ is less than the length of the secret key. This motivated the approach of Naor and Segev [NS09] (later refined by Dodis et al. [DHL$^+$10a, Definition 7.2]) who considered the more general notion of *noisy leakage*, in which the leakage is not necessarily of bounded length, but is guaranteed to reduce the average min-entropy of the secret key by at most $\lambda$. Although our schemes are secure with respect to bounded leakage, they are in fact insecure with respect to noisy leakage. This seems to be the first separation between bounded leakage and noisy leakage, and this settles an open problem posed by Naor and Segev.

Specifically, in our schemes the public key for the $\mathcal{R}^{\mathsf{AD}}$-lossy encryption scheme is sampled obliviously as a uniformly random string $pk \in \{0,1\}^*$. For our specific constructions based on the DDH or Linear assumptions (see Sections 4.2 and 6.2), this can be easily seen to imply that with an overwhelming probably all possible tags for the $\mathcal{R}^{\mathsf{AD}}$-lossy scheme are lossy. An analysis almost identical to that presented in the security proofs of our schemes then shows that a leakage function that simply outputs a signature on any message $m^*$ is a valid leakage function with respect to noisy leakage (yet clearly invalid with respect to bounded leakage).

**Better relative leakage in the continual-leakage model.** Our construction in the continual-leakage model relies on any continual leakage-resilient one-way relation, and the amount of leakage that can be tolerated between any two successive refreshes of the signing key depends on the leakage resilience of the underlying one-way relation. In turn, instantiating our scheme with existing constructions of such one-way relations yields schemes that are resilient to any leakage of length $(1/2 - o(1))L$ bits based on the Linear assumption [DHL$^+$10a, BTK$^+$10], and of length $(1 - o(1))L$ bits based on the Symmetric External Diffie-Hellman assumption [BTK$^+$10]. An interesting open problem is to construct continual leakage-resilient one-way relations based on other assumptions, or with better leakage resilience based on the Linear assumption, and these will immediately improve our schemes.

**Modeling hard-to-invert leakage for signature schemes.** So far signature schemes were considered with respect to leakage with an information-theoretic guarantee: even after seeing the leakage, the signing key still has a certain amount of min-entropy. In the setting of public-key encryption a more general model was formalized by only assuming that the decryption key cannot be efficiently recovered given the leakage (see [DTL09, DGT$^+$10, GTP$^+$10, BG10] and the references therein). For signature schemes, however, due to the interaction between the adversary and the signer, it is not clear how to meaningfully formalize such an attack model. It would be interesting to formalize hard-to-invert leakage for signature schemes (especially when any intermediate value may leak, and not only the signing key), and to construct schemes that are leakage resilient in such a model.

## References

[ADN$^+$10] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs. Public-key encryption in the bounded-retrieval model. In *Advances in Cryptology – EUROCRYPT '10*, pages 113–134, 2010.

[ADW09] J. Alwen, Y. Dodis, and D. Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In *Advances in Cryptology – CRYPTO '09*, pages 36–54, 2009.

[AGV09] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *Proceedings of the 6th Theory of Cryptography Conference*, pages 474–495, 2009.

[BB04] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *Advances in Cryptology – CRYPTO '04*, pages 443–459, 2004.

[BBS04] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Advances in Cryptology – CRYPTO '04*, pages 41–55, 2004.

[BDL97] D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of checking cryptographic protocols for faults. In *Advances in Cryptology – EUROCRYPT '97*, pages 37–51, 1997.

[BG89] M. Bellare and S. Goldwasser. New paradigms for digital signatures and message authentication based on non-interative zero knowledge proofs. In *Advances in Cryptology – CRYPTO '89*, pages 194–211, 1989.

[BG08] B. Barak and O. Goldreich. Universal arguments and their applications. *SIAM Journal on Computing*, pages 1661–1694, 2008.

[BG10] Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability (or: Quadratic Residuosity strikes back). In *Advances in Cryptology – CRYPTO '10*, pages 1–20, 2010.

[BGN05] D. Boneh, E.-J. Goh, and K. Nissim. Evaluating 2-DNF formulas on ciphertexts. In *Proceedings of the 2nd Theory of Cryptography Conference*, pages 325–341, 2005.

[BHY09] M. Bellare, D. Hofheinz, and S. Yilek. Possibility and impossibility results for encryption and commitment secure under selective opening. In *Advances in Cryptology – EUROCRYPT '09*, pages 1–35, 2009.

[BS97] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In *Advances in Cryptology – CRYPTO '97*, pages 513–525, 1997.

[BT10] Z. Brakerski and Y. Tauman Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. Cryptology ePrint Archive, Report 2010/086, 2010.

[BTK$^+$10] Z. Brakerski, Y. Tauman Kalai, J. Katz, and V. Vaikuntanathan. Cryptography resilient to continual memory leakage. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, pages ???–???, 2010.

[CHK$^+$10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *Advances in Cryptology – EUROCRYPT '10*, pages 523–552, 2010.

[DGT$^+$10] Y. Dodis, S. Goldwasser, Y. Tauman Kalai, C. Peikert, and V. Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *Proceedings of the 7th Theory of Cryptography Conference*, pages 361–381, 2010.

[DHL$^+$10a] Y. Dodis, K. Haralambiev, A. Lopez-Alt, and D. Wichs. Cryptography against continuous memory attacks. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, pages ???–???, 2010.

[DHL$^+$10b] Y. Dodis, K. Haralambiev, A. Lopez-Alt, and D. Wichs. Efficient public-key cryptography in the presence of key leakage. Cryptology ePrint Archive, Report 2010/154, 2010.

[DOR$^+$08] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[DP08] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 293–302, 2008.

[DTL09] Y. Dodis, Y. Tauman Kalai, and S. Lovett. On cryptography with auxiliary input. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, pages 621–630, 2009.

[FGK$^+$10] D. M. Freeman, O. Goldreich, E. Kiltz, A. Rosen, and G. Segev. More constructions of lossy and correlation-secure trapdoor functions. In *Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography*, pages 279–295, 2010.

[FKP$^+$10] S. Faust, E. Kiltz, K. Pietrzak, and G. N. Rothblum. Leakage-resilient signatures. In *Proceedings of the 7th Theory of Cryptography Conference*, pages 343–360, 2010.

[FRR$^+$10] S. Faust, T. Rabin, L. Reyzin, E. Tromer, and V. Vaikuntanathan. Protecting circuits from leakage: The computationally-bounded and noisy cases. In *Advances in Cryptology – EUROCRYPT '10*, pages 135–156, 2010.

[FS86] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology – CRYPTO '86*, pages 186–194, 1986.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[GMR88] S. Goldwasser, S. Micali, and R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.

[GOS06] J. Groth, R. Ostrovsky, and A. Sahai. Perfect non-interactive zero knowledge for NP. In *Advance in Cryptology – EUROCRYPT '06*, pages 339–358, 2006.

[GR10] S. Goldwasser and G. Rothblum. How to play mental solitaire under continuous side-channels: A completeness theorem using secure hardware. In *Advances in Cryptology – CRYPTO '10*, pages 59–79, 2010.

[GS08] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Advances in Cryptology – EUROCRYPT '08*, pages 415–432, 2008.

[GTP$^+$10] S. Goldwasser, Y. Tauman Kalai, C. Peikert, and V. Vaikuntanathan. Robustness of the learning with errors assumption. In *Proceedings of the 1st Symposium on Innovations in Computer Science*, pages 230–240, 2010.

[HLO$^+$09] B. Hemenway, B. Libert, R. Ostrovsky, and D. Vergnaud. Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. Cryptology ePrint Archive, Report 2009/088, 2009.

[HR04] C.-Y. Hsiao and L. Reyzin. Finding collisions on a public road, or do secure hash functions need secret coins? In *Advances in Cryptology – CRYPTO '04*, pages 92–105, 2004.

[HSH$^+$08] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. Lest we remember: Cold boot attacks on encryption keys. In *Proceedings of the 17th USENIX Security Symposium*, pages 45–60, 2008.

[HW09] S. Hohenberger and B. Waters. Short and stateless signatures from the RSA assumption. In *Advances in Cryptology – CRYPTO '09*, pages 654–670, 2009.

[ISW03] Y. Ishai, A. Sahai, and D. Wagner. Private circuits: Securing hardware against probing attacks. In *Advances in Cryptology – CRYPTO '03*, pages 463–481, 2003.

[JN03] A. Joux and K. Nguyen. Separating decision Diffie-Hellman from computational Diffie-Hellman in cryptographic groups. *Journal of Cryptology*, 16(4):239–247, 2003.

[JV10] A. Juma and Y. Vahlis. On protecting cryptographic keys against side-channel attacks. In *Advances in Cryptology – CRYPTO '10*, pages 41–58, 2010.

[Kil92] J. Kilian. A note on efficient zero-knowledge proofs and arguments (extended abstract). In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 723–732, 1992.

[Kil07] E. Kiltz. Chosen-ciphertext secure key-encapsulation based on gap hashed Diffie-Hellman. In *Proceedings of the 10th International Conference on Practice and Theory in Public-Key Cryptography*, pages 282–297, 2007.

[KJJ99] P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Advances in Cryptology – CRYPTO '99*, pages 388–397, 1999.

[KN08] G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In *Proceedings of the 5th Theory of Cryptography Conference*, pages 320–339, 2008.

[Koc96] P. C. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Advances in Cryptology – CRYPTO '96*, pages 104–113, 1996.

[KR00] H. Krawczyk and T. Rabin. Chameleon signatures. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2000.

[KV09] J. Katz and V. Vaikuntanathan. Signature schemes with bounded leakage resilience. In *Advances in Cryptology – ASIACRYPT '09*, pages 703–720, 2009.

[LPS10] V. Lyubashevsky, A. Palacio, and G. Segev. Public-key cryptographic primitives provably as secure as subset sum. In *Proceedings of the 7th Theory of Cryptography Conference*, pages 382–400, 2010.

[Mic00] S. Micali. Computationally sound proofs. *SIAM Journal on Computing*, pages 1253–1298, 2000.

[MR04] S. Micali and L. Reyzin. Physically observable cryptography. In *Proceedings of the 1st Theory of Cryptography Conference*, pages 278–296, 2004.

[NS09] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. In *Advances in Cryptology – CRYPTO '09*, pages 18–35, 2009.

[NY89] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 33–43, 1989.

[Pie09] K. Pietrzak. A leakage-resilient mode of operation. In *Advances in Cryptology – EUROCRYPT '09*, pages 462–482, 2009.

[PVW08] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *Advances in Cryptology – CRYPTO '08*, pages 554–571, 2008.

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 187–196, 2008.

[Rom90] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, 1990.

[Sha07] H. Shacham. A Cramer-Shoup encryption scheme from the Linear assumption and from progressively weaker Linear variants. Cryptology ePrint Archive, Report 2007/074, 2007.

[Sho97] V. Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology – EUROCRYPT '97*, pages 256–266, 1997.