Efficient Public-Key Cryptography in the Presence of Key Leakage

Yevgeniy Dodis[*] Kristiyan Haralambiev [†] Adriana López-Alt [‡] Daniel Wichs[§]

September 8, 2010

Abstract

We study the design of cryptographic primitives resistant to a large class of side-channel attacks, called "memory attacks", where an attacker can repeatedly and adaptively learn information about the secret key, subject *only* to the constraint that the *overall amount* of such information is bounded by some parameter ℓ. Although the study of such primitives was initiated only recently by Akavia et al. [2], subsequent work already produced many such "leakage-resilient" primitives [49, 4, 44], including signature, encryption, identification (ID) and authenticated key agreement (AKA) schemes. Unfortunately, every existing scheme, — for any of the four fundamental primitives above, — fails to satisfy at least one of the following desirable properties:

- **Efficiency.** While the construction may be generic, it should have some *efficient* instantiations, based on standard cryptographic assumptions, and without relying on random oracles.
- **Strong Security**. The construction should satisfy the strongest possible definition of security (even in the presence of leakage). For example, encryption schemes should be secure against chosen *ciphertext* attack (CCA), while signatures should be *existentially* unforgeable.
- **Leakage Flexibility.** It should be possible to set the parameters of the schemes so that the leakage bound ℓ can come arbitrarily close to the size of the secret key sk.

In this work we design the first signature, encryption, ID and AKA schemes which overcome these limitations, and satisfy all the properties above. Moreover, all our constructions are generic, in several cases elegantly simplifying and generalizing the prior constructions (which did not have any efficient instantiations). We also introduce several tools of independent interest, such as the abstraction (and constructions) of *true-simulation extractable* NIZK arguments, and a new *deniable* DH-based AKA protocol based on any CCA-secure encryption.

[*]Computer Science Dept. NYU. Email: `dodis@cs.nyu.edu`.
[†]Computer Science Dept. NYU. Email: `kkh@cs.nyu.edu`.
[‡]Computer Science Dept. NYU. Email: `lopez@cs.nyu.edu`.
[§]Computer Science Dept. NYU. Email: `wichs@cs.nyu.edu`.

1 Introduction

Traditionally, the security of cryptographic schemes has been analyzed in an idealized setting, where an adversary only sees the specified "input/output behavior" of a scheme, but has no other access to its internal secret state. Unfortunately, in the real world, an adversary may often learn some partial information about secret state via various *key leakage* attacks. Such attacks come in a large variety and include *side-channel attacks* [45, 11, 8, 46, 55, 30], where the physical realization of a cryptographic primitive can leak additional information, such as the computation-time, power-consumption, radiation/noise/heat emission etc. The cold-boot attack of Halderman et al. [37] is another example of a key-leakage attack, where an adversary can learn (imperfect) information about memory contents of a machine, even after the machine is powered down. Schemes that are proven secure in an idealized setting, without key leakage, may become completely insecure if the adversary learns even a small amount of information about the secret key. Indeed, even very limited leakage attacks have been shown to have devastating consequences for the security of many natural schemes.

Unfortunately, it is unrealistic to assume that we can foresee, let alone block, all of the possible means through which key leakage can occur in real-world implementations of cryptographic schemes. Therefore, the cryptographic community has recently initiated the investigation of increasingly general (formally modeled) classes of leakage attacks, with the aim of constructing *leakage-resilient* cryptographic schemes that remain provably secure even in the presence of such attacks. Of course, if an adversary can get unrestricted information about the secret key (say, of an encryption scheme), then she can learn the key in its entirety and the security of the system is necessarily compromised. Therefore, we must first place some "upper bound" on the type or amount of information that the adversary can learn. The nature of such bounds varies in the literature, as we survey later. For this work, we only restrict the *amount*, but not the *type*, of information that an adversary can learn through a key-leakage attack. In particular, we will assume that the attacker can learn *any efficiently computable function of the secret key sk*, subject only to the constraint that the total amount of information learned (i.e. the output size of the leakage function) is bounded by ℓ bits, where ℓ is called the "leakage parameter" of the system.[1] Clearly, at this level of generality, the secret-key size s must be strictly greater than the leakage-parameter ℓ.[2] Therefore, the quantity ℓ/s can be thought as the *relative leakage* of the system, with the obvious goal to make it as close to 1 as possible.

Our model of leakage-resilience was recently introduced recently by Akavia et al. [2], but already attracted a lot of attention from the cryptographic community [49, 4, 44, 3]. In particular, as we survey later, we already know many "leakage-resilient" primitives, including such fundamental primitives as signature schemes, encryption schemes, identification (ID) schemes and authenticated key agreement (AKA) protocols. Unfortunately, we observe that every existing scheme, — for any of the four fundamental primitives above, — fails to satisfy at least one of the following desirable properties:

- **Efficiency.** While the proposed construction may be based on some generic cryptographic primitives, — which is in fact preferable for modular design, — it should have some *efficient* instantiations, based on standard cryptographic assumptions, and without relying on random oracles. We view this property as the main property we will strive to achieve.
- **Strong Security**. The construction should satisfy the strongest possible definition of security (even in the presence of leakage). For example, encryption schemes should be secure against chosen *ciphertext* attack (CCA), while signatures should be *existentially* unforgeable, etc.
- **Leakage Flexibility.** It should be possible to set the parameters of the schemes so that the relative leakage ℓ/s is arbitrarily close to 1. We call such schemes *leakage-flexible*.

[1]More formally, we allow adaptive measurements, as long as the sum of leaked outputs is bounded by ℓ.

[2]In fact, our actual constructions easily extend to the more general "noisy leakage" model of Naor and Segev [49], where the outputs can be longer than s, as long as the "average min-entropy" of sk drops by at most ℓ bits. However, we do not pursue this generalization, in order to keep our notation simple.

1.1 Our Results

In this work we design the first signature, encryption, ID and AKA schemes which simultaneously satisfy the efficiency, strong security and leakage flexibility properties mentioned above. Moreover, all our constructions are generic. This means that the actual construction is modularly defined and explained using natural simpler blocks, and its security against key leakage is also proven no matter how these simpler blocks are (securely) implemented. However, unlike the prior generic constructions, which did not have any known efficient instantiations (at least, with the desired security and flexibility we seek), ours are yet more general, which will allow us to obtain several efficient instantiations. Given this fact, it is not surprising that our contributions can be roughly split into two categories: "conceptual" contributions, allowing us to obtain more general (and, yet, conceptually simpler) leakage-resilient constructions, and "concrete" contributions, allowing us to actually instantiate our general schemes efficiently.

CONCEPTUAL CONTRIBUTIONS. As we will see, existing schemes (e.g., signature and CCA-encryption) could be largely divided into two categories: potentially efficient schemes, with some *inherent* limitation not allowing them to achieve relative leakage approaching 1 (which also prevents us from using these ideas for our purposes), and more theoretical schemes [49, 44], achieving good relative leakage, but relying on the notion of *simulation-sound* non-interactive zero-knowledge (ss-NIZK) [56]. Informally, ss-NIZK proofs remain sound even if the attacker can see simulated proofs of arbitrary (even false) statements. Unfortunately, it appears that the existing cryptographic machinery does not allow us to instantiate non-trivial ss-NIZK proofs efficiently.[3] On the other hand, a recent breakthrough result of Groth-Sahai [36] showed that one can obtain efficient *non-simulation-sound* NIZK proofs for a non-trivial class of languages. While the techniques of [34] could be applied to Groth-Sahai proofs to achiehve ss-NIZKs, it is a non-trivial "exercise" and the resulting proofs are *significantly* less efficient, as the construction involves OR-proofs for Groth-Sahai languages. Therefore, our first idea was to try to generalize the existing constructions sufficiently, making them rely only on regular NIZKs, in the hope that such regular NIZKs can then be instantiated using the powerful Groth-Sahai techniques.

In the end, this is indeed what we realized. However, in the process we also abstracted away an elegant notion of independent interest: *true-simulation extractable* (tSE) NIZKs. While quite similar to the notion of simulation-sound extractable NIZKs [34], it involves a subtle but rather important difference: whether the adversary has oracle access to simulated proofs for arbitrary (even false) statements or only true ones. Intuitively, both the Naor-Segev's leakage-resilient CCA encryption [49] and Katz-Vaikuntanathan's leakage-resilient signature scheme [44] used the technique of encrypting a witness x for some relation R, and then providing a ss-NIZK proof φ that the ciphertext c indeed contains the encryption of a valid witness x. The main reason for using this technique is to allow the reduction to extract a valid witness from any "new" valid pair (c^*, φ^*) produced by the attacker $\mathcal{A}$ (who saw many such valid pairs earlier). In this paper, we will abstract this property into the tSE notion mentioned above (of which the above mentioned technique is a specific example, where the pair (c, φ) together makes up a single tSE-NIZK proof). Moreover, we show that true-simulation extractability, as we abstract it, is *precisely* the right notion for generalizing and proving the security of the previous constructions. This has two positive effects. First, it makes the generic constructions of CCA-encryption and signatures somewhat more intuitive, both for proving and understanding. For example, the traditional "double-encryption" paradigm of Naor-Yung [50] for designing CCA-secure schemes from chosen-plaintext secure (CPA-secure) schemes, also used by [49] in the context of key leakage, can be stated as "CPA-encrypting message m under two keys and proving plaintext equality". Using our more general "simulation-extractability view", it is now stated as "CPA-encrypting m and proving that one knows the plaintext". We believe that the latter view is not only more general, but also more intuitive as a way of explaining "CPA-to-CCA" transformation. A similar discussion is true for our signature constructions.

Second, we show a generic way to build tSE-NIZKs which *avoids using (expensive) ss-NIZKs*. Instead,

[3] The work of [34] constructs ss-NIZK proofs for practical languages and uses them to construct group signatures, but the resulting scheme has signature size of "thousands or perhaps even millions of group elements" [35] despite being constant.

Reference	**Unforgeability**	**Model**	**Leakage**	**Efficient?**
[4]	Existential	*Random Oracle*	*1/2*	Yes
[4]	*Entropic*	*Random Oracle*	1	Yes
[44]	Existential	Standard	1	*No*
This Work	Existential	Standard	1	Yes

Table 1: Previous work on leakage-resilient signatures and results of this work

Reference	**Attack**	**Model**	**Leakage**	**Efficient?**
[2, 49]	*CPA*	Standard	1	Yes
[49]	CCA	Standard	*1/6*	Yes
[49]	CCA	Standard	1	*No*
This Work	CCA	Standard	1	Yes

Table 2: Previous work on leakage-resilient encryption and results of this work

our method uses *regular* NIZKs and *any* CCA-secure encryption scheme.[4] Perhaps surprisingly, given the current state-of-the-art NIZK and CCA schemes, the combination "CCA + NIZK" appears to be much more efficient in practice than the combination "CPA + ss-NIZK".[5] As a result, we were able to provide a general framework for building leakage-flexible signature and CCA-encryption schemes, eventually allowing us to efficiently instantiate our schemes (by avoiding using ss-NIZKs). We summarize our results for signature and CCA-encryption schemes in Tables 1 and 2, also comparing them to the best prior constructions. In all the tables, the "sub-optimal" entries (for efficiency, security, model or relative leakage of prior constructions) are written in italics, and most prior rows are also explained in the related work Section 1.2. For signatures, we stress that no efficient construction in the standard model was known prior to our work, for any non-trivial relative leakage fraction (let alone 1).

Once we have efficient leakage-flexible signature schemes, we observe that the standard signature-based ID scheme, where the verifier asks the prover to sign a random message, easily extends to the leakage setting. Moreover, the resulting actively secure ID scheme inherits its relative leakage from the corresponding signature scheme, and satisfies the strongest notion of "anytime-leakage" [4] (see Section 6.1), where the leakage can occur even during the impersonation attack. We summarize our results for ID schemes in Table 3. Although our method is pretty simple, we notice that the other two popular methods of building ID schemes — the use of Σ-protocols for hard relations analyzed in [4] (see first two rows of Tables 3), and the use of CCA-secure encryption (where the prover decrypts a random challenge ciphertext) — inherently do not allow us to obtain optimal results, even when instantiated with leakage-flexible hard relations or CCA-encryption schemes. See Section 6.1 for more details.

Finally, we summarize our results for AKA protocols in Table 4. We actually obtain two such protocols. First, similarly to the case of ID schemes, we can obtain leakage-resilient AKA schemes from any leakage-resilient signature scheme, as formally explained in [4]. The idea is to essentially sign every flow of a standard Diffie-Hellman-based protocol, but with a leakage-resilient signature scheme. We notice, though, that the resulting protocol is not *deniable*. Namely, the transcript of the protocol leaves irrefutable evidence that the protocol took place. Motivated by this deficiency, we design another general AKA protocol based on CCA-encryption. The details are given in Section 6.2, but, intuitively, the parties encrypt the flows of the standard Diffie-Hellman-based protocol, effectively proving their identities by successfully re-encrypting the appropriate flows. Although we do not formalize this, this protocols is "deniable", because the transcript of the protocol

[4]This is OK for the signature application, but might appear strange for our CCA-encryption application, as we need "CCA to get CCA". However, as a building block for tSE-NIZKs, we only need *standard* CCA schemes (which are known), and as a result obtain *leakage-resilient* CCA schemes.

[5]Indirectly, the same realization was made by Groth [35] and Camenisch et al. [13] in different concrete contexts.

Reference	**Security**	**Model**	**Leakage**	**Efficient?**
[4]	*Pre-Impersonation*	Standard	1	Yes
[4]	Anytime	Standard	*1/2*	Yes
[44] (implicit)	Anytime	Standard	1	*No*
This Work	Anytime	Standard	1	Yes

Table 3: Previous work on leakage-resilient identification schemes and results of this work

Reference	**Model**	**Leakage**	**Deniable?**	**Efficient?**
[4]	*Random Oracle*	1	*No*	Yes
[4, 44]	Standard	1	*No*	*No*
This Work	Standard	1	*No*/Yes*	Yes

* Our first AKA protocol is not deniable; our second — is.

Table 4: Previous work on leakage-resilient AKA and results of this work.

can be simulated without the knowledge of parties' secret keys. To the best of our knowledge, this protocol was not suggested and analyzed even in the leakage-free setting, where it appears interesting already. Here we actually show that our (new) deniable AKA protocol works even in the presence of leakage.

CONCRETE CONTRIBUTIONS. As we explained above, we generically reduce the question of building efficient leakage-flexible ID schemes and AKA protocol to the question of efficiently instantiating our leakage-flexible signature and/or encryption schemes. Such instantiations are given in Section 5 (with most details in Appendix C). We also explained how the latter instantiations became possible in our work, since we gave generic constructions of both primitives based on the new notion of tSE-NIZK, and then showed that satisfying this notion may be possible using *ordinary* NIZKs for appropriate languages, without relying on the expensive simulation-sound NIZKs. Unfortunately, efficient construction of (even ordinary) NIZKs, due to Groth and Sahai [36], are only known for a pretty restrictive class or languages in bilinear groups. Thus, obtaining a *concrete* efficient instantiation still requires quite a substantial effort.

Specifically, all the building blocks have to be instantiated efficiently, and expressed in a form such that the resulting NP relation satisfies the severe limitations imposed by the Groth-Sahai NIZKs. For example, to build leakage-resilient CCA-encryption, we need to have an efficient leakage-flexible CPA scheme, a CCA scheme supporting labels and a one-time signature scheme, all connected together by an efficient NIZK for a complicated "plaintext equality" relation. Similarly, for leakage-resilient signature schemes, we need an efficient second-preimage resistant (SPR; see Definition 2.1) relation and a CCA scheme supporting labels, once again connected by an efficient NIZK for a complex relation. Not surprisingly, such tasks cannot typically be done by simply combining "off-the-shelf" schemes from the literature. At best, it requires very careful selection of parameters to make everything "match", followed by a round of further efficiency optimizations. Usually, though, it requires the design of new primitives, which work well with other known primitives, to enable efficient NIZK. For example, in this work, we designed two new SPR relations (see Claims C.1 and C.2), since prior SPR relations did not appear to mesh well with our CCA encryption scheme. To emphasize the importance of the new SPR relations, we point out that combining previous constructions with Groth-Sahai proofs would require committing to the witness bit-by-bit in order to achieve full extractability.

Overall, we get two different efficient instantiations of both leakage-resilient signature and CCA encryption schemes in the standard model, based on standard (static and "fixed-length") assumptions in bilinear groups, called external Diffie-Hellman (SXDH) and Decision-Linear (DLIN). Ignoring many technicalities, the high-level idea of all these schemes, as well as the efficiency they achieve, is described in Section 5. The actual low-level details of how to put "everything together", in the most efficient manner, is described Appendix C.

1.2 Related Work

LEAKAGE-RESILIENCE AND MEMORY ATTACKS. Our model of leakage, sometimes called memory-attacks, was first proposed by Akavia, Goldwasser and Vaikuntanathan [2], who also constructed CPA secure PKE and IBE schemes in this model under the *learning with errors (LWE)* assumption. Later Naor and Segev [49] generalized the main ideas behind these constructions to show that all schemes based on *hash proof systems* (see [18]) are leakage-resilient. In particular, this resulted in efficient constructions based on the DDH and K-Linear assumptions, where the relative leakage on the secret key could be made to approach 1. Moreover, [49] showed how to also achieve CCA security in this model by either: (1) relying on the generic (and inefficient) Naor-Yung paradigm where the leakage-rate can be made to approach 1 or (2) using efficient hash proof systems with leakage-rate only approaching $1/6$. Unfortunately, it seems that the hash proof system approach to building CCA encryption is inherently limited to leakage-rates below $1/2$: this is because the secret-key consists of two components (one for verifying that the ciphertext is well-formed and one for decrypting it) and the proofs break down if either of the components is individually leaked in its entirety.

The work of [3] generalizes [49] still further by showing how to construct leakage-resilient IBE schemes generically based on *identity-based hash proof systems*, with several instantiations.

Leakage-resilient signature schemes in the model of memory attacks were constructed in the random-oracle model by [4, 44], and in the standard model by [44]. The random-oracle schemes are highly-efficient but suffer from two limitations. Firstly they rely on the Fiat-Shamir [28] transform which is only known to be secure in the Random Oracle model and is not sound in general [32]. Secondly, the schemes can only tolerate leakage which approaches $1/2$ of the secret key. On the other hand, the standard-model schemes allow for relative-leakage approaching 1, but are based on generic simulation-sound NIZKs and do not come with an efficient instantiation.

The work of [4] also constructs identification (ID) schemes and authenticated-key agreement (AKA) protocols. For ID schemes, two notions of security (we describe these in detail in Section 6.1) were considered: a weaker notion called pre-impersonation leakage-resilience and a stronger notion called anytime leakage-resilience. Although efficient schemes in the standard model were given for both notions, the leakage resilience could be made to approach 1 only for pre-impersonation leakage while, for anytime leakage, the given schemes can only tolerate a leakage-rate below $1/2$. For AKA schemes, a construction was given based on leakage-resilient signatures (only requiring a weakened notion of security called entropic-unforgeability). Using the appropriate signature schemes, this yielded two types of constructions: efficient constructions in the random-oracle model and generic but inefficient constructions in the standard model (both of which have leakage-rates approaching 1).

OTHER MODELS OF LEAKAGE-RESILIENCE. Several other models of leakage-resilience have appeared in the literature. They differ from the model we described in that they restrict the *type*, as well as *amount*, of information that the adversary can learn. For example, the work on *exposure resilient cryptography* [14, 23, 43] studies the case where an adversary can only learn some small *subset of the physical bits of the secret key*. Similarly, [41] studies how to implement arbitrary computation in the setting where an adversary can observe a small *subset of the physical wires of a circuit*. Most recently, [27] study a similar problem, where the adversary can observe a low-complexity (e.g. AC^0) function of the wires. Unfortunately, these models fail to capture many meaningful side-channel attacks, such as learning the hamming-weight of the bits or their parity.

In their seminal work, Micali and Reyzin [48] initiated the formal modeling of side-channel attacks under the axiom that *"only computation leaks information"* (OCLI), where each invocation of a cryptographic primitive leaks a function of *only* the bits accessed during that invocation. Several primitives have been constructed in this setting including stream ciphers [25, 54] and signatures [26]. More recently, [42] construct a general compiler that can secure *all primitives* in this setting assuming the use of some limited leak-free components and the existence of fully homomorphic encryption. On the positive side, the OCLI model only imposes a bound on the amount of information learned during each invocation of a primitive, but not on the overall amount of information that the attacker can get throughout the lifetime of the system. On the negative side, this model fails to capture many leakage-attacks, such as the cold-boot attack of [37], where *all* memory contents leak information, even if they were never accessed.

Lastly, we mention several models of leakage-resilience which are strictly stronger than the memory-attacks model. Firstly, the Bounded-Retrieval Model [19, 24, 4, 3] imposes an additional requirement on leakage-resilient schemes, by insisting that they provide a way to "grow" the secret-key (possibly to many Gigabytes) so as to proportionally increase the amount of tolerated leakage, but without increasing the size of the public-key, the computational-efficiency of the scheme, or the ciphertext/signature/communication lengths. The work of [4] constructs "entropic" signatures, ID schemes and AKA protocols in this setting, while the work of [3] constructs PKE and IBE schemes in this model. A different strengthening is the auxiliary input model [21, 20] where the leakage is not necessarily bounded in length, but it is (only) assumed to be computationally hard to recover the secret-key from the leakage. The work of [21] constructs symmetric-key encryption in this model, under a strengthening of the learning parity with noise (LPN) assumption, while [20] constructs public-key encryption under the DDH and LWE assumptions. Yet another strengthening of the memory-attacks model, proposed by [31], is to require that there is a single scheme (parameterized only by the security parameter) which can tolerate essentially any amount of relative-leakage where the exact-security of the scheme degrades smoothly as the relative-leakage increases. In this model, [31] construct a symmetric-key encryption scheme.

2 Definitions of Leakage-Resilient Primitives

We model leakage attacks by giving the adversary access to a *leakage oracle*, which he can adaptively access to learn leakage on the secret key. A leakage oracle $\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)$ is parametrized by a secret key sk, a leakage parameter ℓ, and a security parameter λ. A query to the leakage oracle consists of a function $h_i : \{0,1\}^* \to \{0,1\}^{\alpha_i}$, to which the oracle answers with $y_i = h_i(sk)$. We only require that the functions h_i be efficiently computable, and the total number of bits leaked is $\sum_i \alpha_i \leq \ell$.

Definition 2.1 (Leakage Resilient Hard Relation)**.** *A relation R with a randomized PPT sampling algorithm* `KeyGen` *is an ℓ-*leakage resilient hard relation *if:*

- *For any $(sk, pk) \leftarrow \texttt{KeyGen}(1^\lambda)$, we have $(sk, pk) \in R$.*
- *There is a poly-time algorithm that decides if $(sk, pk) \in R$.*
- *For all PPT adversaries $\mathcal{A}^{\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)}$ with access to the leakage oracle $\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)$, we have that*

$$\Pr\left[R(sk^*, pk) = 1 \mid (pk, sk) \leftarrow \texttt{KeyGen}(1^\lambda)\ ,\ sk^* \leftarrow \mathcal{A}^{\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)}(pk)\right] \leq negl(\lambda)$$

Notice that without loss of generality, we can assume that $\mathcal{A}$ queries $\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)$ only once with a function h whose output is ℓ bits.

Definition 2.2 (Leakage Resilient Signatures)**.** *A signature scheme $\mathcal{S} = (\texttt{KeyGen}, \texttt{Sign}, \texttt{SigVer})$ is ℓ-*leakage resilient *if $\forall$ PPT $\mathcal{A}$ we have $\Pr[\mathcal{A}$ wins$] \leq negl(\lambda)$ in the following game:*

1. **Key Generation:** *The challenger runs $(vk, sk) \leftarrow \texttt{KeyGen}(1^\lambda)$ and gives vk to $\mathcal{A}$.*
2. **Signing and leakage queries:** *$\mathcal{A}^{\mathcal{O}_{sk}^{\lambda,\ell}(\cdot), \mathcal{S}_{sk}(\cdot)}$ is given access to the leakage oracle $\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)$ and the signing oracle $\mathcal{S}_{sk}(\cdot)$. A query to the signing oracle $\mathcal{S}_{sk}(\cdot)$ consists of a message m, to which the oracle responds with $\sigma = \texttt{Sign}_{sk}(m)$.*
3. *$\mathcal{A}$ outputs (m^*, σ^*) and wins if $\texttt{SigVer}_{vk}(m^*, \sigma^*) = 1$ and m^* was not given to $\mathcal{S}_{sk}(\cdot)$ as a signing query.*

Definition 2.3 (Leakage Resilient CCA-Secure Encryption)**.** *We say that an encryption scheme* $\mathcal{E} = (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$ *is* ℓ-leakage resilient CCA-secure *if* $\forall$ *PPT* $\mathcal{A}$ *we have* $\Pr[\mathcal{A}\ wins] \leq \frac{1}{2} + negl(\lambda)$ *in the following game:*

1. **Key Generation:** *The challenger runs* $(pk, sk) \leftarrow \texttt{KeyGen}(1^\lambda)$ *and gives* pk *to* $\mathcal{A}$.
2. **Decryption and leakage queries:** $\mathcal{A}^{\mathcal{O}_{sk}^{\lambda,\ell}(\cdot),\mathcal{D}_{sk}(\cdot)}$ *is given access to the leakage oracle* $\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)$ *and the decryption oracle* $\mathcal{D}_{sk}(\cdot)$*. A query to the decryption oracle* $\mathcal{D}_{sk}(\cdot)$ *consists of a ciphertext* c*, to which the oracle responds with* $m = \texttt{Dec}_{sk}(c)$.
3. **Challenge generation:** $\mathcal{A}$ *sends plaintexts* m_0, m_1 *to the challenger. The challenger chooses* $b \xleftarrow{\$} \{0,1\}$*, and sends* $c^* \leftarrow \texttt{Enc}_{pk}(m_b)$ *to* $\mathcal{A}$.
4. **Decryption queries:** $\mathcal{A}^{\mathcal{D}_{sk}(\cdot)}$ *is given access to the decryption oracle* $\mathcal{D}_{sk}(\cdot)$ *with the restriction that* $\mathcal{A}$ *cannot send* c^* *as a decryption query. Notice also that* $\mathcal{A}^{\mathcal{D}_{sk}(\cdot)}$ *is* not *given access to the leakage oracle* $\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)$.
5. $\mathcal{A}$ *outputs* b'*, and wins if* $b = b'$.

If an encryption scheme is 0-leakage-resilient CCA-secure *we simply refer to it as being* CCA secure.

Recall that we can define labeled CCA encryption in which a message is encrypted and decrypted according to a public label L. If an encryption scheme $\mathcal{E} = (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$ supports labels, we use the syntax $\texttt{Enc}^L(m)$ to denote the encryption of message m under label L. Similarly, we use $\texttt{Dec}^L(c)$ to denote the decryption of ciphertext c under the label L. In this case, we extend the correctness of encryption/decryption to requiring that $\texttt{Dec}^L(\texttt{Enc}^L(m)) = m$. The security definition described in Definition 2.3 can also be easily modified as follows. A query to the decryption oracle now consists of a ciphertext c and a label L, to which the oracle responds with $m = \texttt{Dec}_{sk}^L(c)$. In the challenge generation stage, $\mathcal{A}$ submits a label L^* as well as messages m_0, m_1 and the challenger computes $c^* \leftarrow \texttt{Enc}_{pk}^{L^*}(m_b)$ for $b \xleftarrow{\$} \{0,1\}$. Finally, in the second stage of decryption queries we require that the adversary is allowed to ask for decryptions of any ciphertext c under label L only subject to $(L, c) \neq (L^*, c^*)$.

Definition 2.4 (Leakage Resilient CPA-Secure Encryption)**.** *We say that an encryption scheme* $\mathcal{E} = (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$ *is* ℓ-leakage resilient CPA-secure *if* $\forall$ *PPT* $\mathcal{A}$ *we have* $\Pr[\mathcal{A}\ wins] \leq \frac{1}{2} + negl(\lambda)$ *in the game described above with the modification that* $\mathcal{A}$ *does not have access to the decryption oracle* $\mathcal{D}_{sk}(\cdot)$*. If an encryption scheme is* 0-leakage-resilient CPA-secure *we simply refer to it as being* CPA secure.

3 Simulation Extractability

We start by briefly recalling the notion of *non-interactive zero-knowledge (NIZK)* [9]. For our purposes, it will be slightly more convenient to use the notion of *(same-string) NIZK argument* from [57]. Note, however, that the definitions and constructions given in this section can be extended to the case of NIZK proofs.

Let R be an NP relation on pairs (x, y) with corresponding language $L_R = \{y \mid \exists x \text{ s.t. } (x, y) \in R\}$. A *non-interactive zero-knowledge (NIZK) argument* for a relation R consists of three algorithms $(\texttt{Setup}, \texttt{Prove}, \texttt{Verify})$ with syntax:

- $(\texttt{CRS}, \texttt{TK}) \leftarrow \texttt{Setup}(1^\lambda)$: Creates a common reference string (CRS) and a trapdoor key to the CRS.
- $\pi \leftarrow \texttt{Prove}_{\texttt{CRS}}(x, y)$: Creates an argument that $R(x, y) = 1$.
- $0/1 \leftarrow \texttt{Verify}_{\texttt{CRS}}(y, \pi)$: Verifies whether or not the argument π is correct.

For the sake of clarity, we write $\texttt{Prove}$ and $\texttt{Verify}$ without the $\texttt{CRS}$ in the subscript when the $\texttt{CRS}$ can be inferred from the context. We require that the following three properties hold:

Completeness: For any $(x, y) \in R$, if $(\texttt{CRS}, \texttt{TK}) \leftarrow \texttt{Setup}(1^\lambda)$, $\pi \leftarrow \texttt{Prove}(x, y)$, then $\texttt{Verify}(y, \pi) = 1$.

Soundness: For any PPT adversary $\mathcal{A}$,

$$\Pr\left[\begin{array}{c} \texttt{Verify}(y, \pi^*) = 1 \\ y \notin L_R \end{array} \middle| \begin{array}{c} (\texttt{CRS}, \texttt{TK}) \leftarrow \texttt{Setup}(1^\lambda) \\ (y, \pi^*) \leftarrow \mathcal{A}(\texttt{CRS}) \end{array}\right] \le negl(\lambda).$$

Composable Zero-Knowledge: There exists PPT simulator $\texttt{Sim}$ such that, for any PPT adversary $\mathcal{A}$ we have $\left|\Pr[\mathcal{A} \text{ wins }] - \frac{1}{2}\right| \le negl(\lambda)$ in the following game:

- The challenger samples $(\texttt{CRS}, \texttt{TK}) \leftarrow \texttt{Setup}(1^\lambda)$ and gives $(\texttt{CRS}, \texttt{TK})$ to $\mathcal{A}$.
- The adv. $\mathcal{A}$ chooses $(x, y) \in R$ and gives these to the challenger.
- The challenger samples $\pi_0 \leftarrow \texttt{Prove}(x, y)$, $\pi_1 \leftarrow \texttt{Sim}(y, \texttt{TK})$, $b \leftarrow \{0, 1\}$ and gives π_b to $\mathcal{A}$.
- The adv. $\mathcal{A}$ outputs a bit $\tilde{b}$ and wins if $\tilde{b} = b$.

We revisit the notion of simulation extractable NIZK arguments [58, 16, 52, 53, 34], and define a new primitive called *true-simulation extractable* NIZK arguments. Apart from satisfying the three properties described above, an NIZK argument is simulation extractable if there exists a PPT *extractor* $\texttt{Ext}$ which (when given an additional extraction trapdoor to the $\texttt{CRS}$) extracts a witness x' from any proof π produced by a malicious prover P^*, *even* if P^* has previously seen some *simulated proofs* for other statements. We make an important distinction between our new definition of *true*-simulation extractability, where all simulated proofs seen by P^* are only of *true* statements, and the stronger notion of *any*-simulation extractability, where P^* can also see proofs of *false* statements. As we will see, the former notion is often simpler to construct and sufficient in our applications.

We extend our definition to *f-extractability*, where $\texttt{Ext}$ only needs to output some function $f(x')$ of a valid witness x'. We further extend this definition to support *labels*, so that the $\texttt{Prove}$, $\texttt{Verify}$, $\texttt{Sim}$, and $\texttt{Ext}$ algorithms now also take a public label L as input, and the correctness, soundness, and zero-knowlegde properties are updated accordingly. If $\Pi = (\texttt{Setup}, \texttt{Prove}, \texttt{Verify})$ is an NIZK argument with simulator $\texttt{Sim}$ and extractor $\texttt{Ext}$, we write $\texttt{Prove}^L, \texttt{Verify}^L, \texttt{Sim}^L, \texttt{Ext}^L$ to denote proof, verification, simulation, and extraction under label L, respectively.

We start by defining a simulation oracle $\mathcal{SIM}_{\text{TK}}(\cdot)$. A query to the simulation oracle consists of a pair (x, y) and a label L. The oracle checks if $(x, y) \in R$. If true, it ignores x and outputs a simulated argument $\texttt{Sim}^L(\texttt{TK}, y)$, and otherwise outputs $\perp$. We now give a formal definition of true-simulation extractability.

Definition 3.1 (True-Simulation f-Extractability)**.** *Let f be a fixed efficiently computable function and let* $\Pi = (\texttt{Setup}, \texttt{Prove}, \texttt{Verify})$ *be an NIZK argument for a relation R, satisfying the completeness, soundness and zero-knowledge properties above. We say that* Π *is* true-simulation f-extractable *(f-tSE) with labels if:*

- *Apart from outputting a CRS and a trapdoor key,* $\texttt{Setup}$ *also outputs an extraction key:* $(\texttt{CRS}, \texttt{TK}, \texttt{EK}) \leftarrow \texttt{Setup}(1^\lambda)$.
- *There exists a PPT algorithm* $\texttt{Ext}(y, \varphi, \texttt{EK})$ *such that for all P^* we have* $\Pr[P^*\ wins] \le negl(\lambda)$ *in the following game:*
  1. **Key Generation:** *The challenger runs* $(\texttt{CRS}, \texttt{TK}, \texttt{EK}) \leftarrow \texttt{Setup}(1^\lambda)$ *and gives* $\texttt{CRS}$ *to P^*.*
  2. **Simulation queries:** ${P^*}^{\mathcal{SIM}_{\text{TK}}(\cdot)}$ *is given access to the simulation oracle* $\mathcal{SIM}_{\text{TK}}(\cdot)$*, which it can adaptively access.*

3. **Adversary Output:** *P^* outputs a tuple (y^*, L^*, φ^*).*
4. **Extraction:** *The challenger runs $z^* \leftarrow \mathtt{Ext}^{L^*}(y^*, \varphi^*, \mathtt{EK})$.*
5. *P^* wins if (a) the pair (y^*, L^*) was not part of a simulator query, (b) $\mathtt{Verify}^{L^*}(y^*, \varphi^*) = 1$, and (c) for all x' such that $f(x') = z^*$ we have $R(x', y^*) = 0$.*[6]

In the case when f is the identity function, we simply say that Π is true-simulation extractable (tSE).

We give several variations of this new primitive. First, we define *one-time* simulation extractability, in which the adversary P^* is only given *a single* query to the simulation oracle $\mathcal{SIM}_{\mathtt{TK}}(\cdot)$. Second, we define the notion of *strong* simulation extractability by changing the winning condition so that P^* is now required to output a new statement/argument pair instead of a new statement. More formally, condition 5a becomes: the tuple (y^*, L^*, φ^*) is new, that is, either (y^*, L^*) was not part of a simulator query, or if it was, the argument φ^* is different from the one(s) given to P^* by $\mathcal{SIM}_{\mathtt{TK}}(\cdot)$. We observe that we can generically construct strong f-tSE NIZK arguments from (standard) f-tSE NIZK arguments if we additionally use a strongly-secure one-time signature. In particular, the prover now computes the standard f-tSE argument, signs it, and attaches the verification key vk to the public label. To verify, we first check that the signature is valid and then verify the f-tSE argument.

Finally, we say that an NIZK argument Π is *any-simultation f-extractable (f-aSE)* (similar to the notion of simulation-sound extractability of [34]) if the adversary P^* instead has access to a modified simulation oracle $\widetilde{\mathcal{SIM}}_{\mathtt{TK}}(\cdot)$ that responds to all simulation queries without checking that $R(x, y) = 1$ (and hence might also give simulated arguments of false statements). In this work we do not make use of this variation, but state it here because as we will see, this notion has been implicitly used in prior works. However, f-aSE is a stronger notion than f-tSE and is *not needed*, as we will show that f-tSE is sufficient in constructing leakage-resilient signatures and CCA-encryption.

4 Generic Constructions

In this section we give generic constructions of leakage-resilient hard relations (Section 4.1) , leakage-resilient signatures (Section 4.2), leakage-resilient CCA-secure encryption (Section 4.3). In the latter two we use the f-tSE NIZK primitive that we defined in Section 3. Finally, in Section 4.4 we give a construction of f-tSE NIZK arguments.

4.1 Leakage-Resilient Hard Relations

We begin by showing how to generically construct leakage-resilient hard relations from SPR relations. Informally, we say that a relation R is *second-preimage resistant (SPR)* if given a random $(x, y) \in R$ it is difficult to find $x' \neq x$ such that $(x', y) \in R$. We formalize this in the following definition.

Definition 4.1 (Second-Preimage Resistant (SPR) Relation)**.** *A relation R with a randomized PPT sampling algorithm* `KeyGen` *is* second-preimage resistant *if:*

- *For any $(x, y) \leftarrow \mathtt{KeyGen}(1^\lambda)$, we have $(x, y) \in R$.*
- *There is a poly-time algorithm that decides if $(x, y) \in R$.*
- *For any PPT algorithm $\mathcal{A}$, we have* $\Pr\left[(x', y) \in R \wedge x' \neq x \;\middle|\; \begin{array}{c} (x, y) \leftarrow \mathtt{KeyGen}(1^\lambda) \\ x' \leftarrow \mathcal{A}(x, y) \end{array}\right] \leq negl(\lambda)$.

[6] In other words, the adversary wins if the extractor fails to extract a good value z^* which corresponds to at least one valid witness x'; i.e. $f(x') = z^*$. For the identity function, $f(x) = x$, this corresponds to the statement: $R(z^*, y) = 0$.

We define the average-case pre-image entropy *of the SPR relation to be* $\mathbf{H}_{avg}(R) = \widetilde{\mathbf{H}}_\infty(X \mid Y)$ *, where the random variables* (X, Y) *are distributed according to* $\texttt{KeyGen}(1^\lambda)$*. (We refer the reader to Appendix A.1 for the definition of* $\widetilde{\mathbf{H}}_\infty(X \mid Y)$*.)*

Theorem 4.2. *If* $R(x, y)$ *is an SPR relation, then it is also an* ℓ*-leakage resilient hard relation for* $\ell = \mathbf{H}_{avg}(R) - \omega(\log \lambda)$*, where* λ *is the security parameter.*

The proof of Theorem 4.2 is given in Appendix A.1.

4.2 Leakage-Resilient Signatures

In this section, we give a generic construction of leakage-resilient signatures based on leakage-resilient hard relations and tSE-NIZK arguments. Let $R(x, y)$ be an ℓ-leakage resilient hard relation with sampling algorithm $\texttt{KeyGen}_R(1^\lambda)$. Let $\Pi = (\texttt{Setup}, \texttt{Prove}, \texttt{Verify})$ be a tSE-NIZK argument for relation R supporting labels. Consider the following signature scheme:

- $\texttt{KeyGen}(1^\lambda)$: Output $sk = x$ and $vk = (\texttt{CRS}, y)$ where $(x, y) \leftarrow \texttt{KeyGen}_R(1^\lambda)$, $(\texttt{CRS}, \texttt{TK}, \texttt{EK}) \leftarrow \texttt{Setup}(1^\lambda)$.
- $\texttt{Sign}_{sk}(m)$: Output $\sigma = \varphi$ where $\varphi \leftarrow \texttt{Prove}^m(x, y)$. (Note that m is the *label* in the argument.)
- $\texttt{SigVer}_{vk}(m, \sigma)$: Output $\texttt{Verify}^m(y, \sigma)$.

Theorem 4.3. *If* $R(x, y)$ *is an* ℓ*-leakage resilient hard relation and* Π *is a labeled tSE-NIZK argument for R, then the above signature scheme is an* ℓ*-leakage resilient signature scheme.*

The proof of Theorem 4.3 is given in Appendix A.2.

4.3 Leakage-Resilient CCA-Secure Encryption

In this section, we give a generic construction of leakage-resilient CCA-secure encryption from leakage-resilient CPA-secure encryption and strong f-tSE NIZK arguments. Let $\mathcal{E} = (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$ be an ℓ-LR-CPA secure encryption scheme and let $\Pi = (\texttt{Setup}, \texttt{Prove}, \texttt{Verify})$ be a one-time, strong f-tSE NIZK argument for the relation

$$R_{enc} = \{ \ ((m, r) \ , \ (pk, c) \ ) \quad | \quad c = \texttt{Enc}_{pk}(m; r) \ \}.$$

where $f(m, r) = m$ (i.e. the extractor only needs to extract the message m, but not the randomness r of encryption). We show how to use $\mathcal{E}, \Pi$ to construct an ℓ-LR-CCA encryption scheme $\mathcal{E}^*$.

Define $\mathcal{E}^* = (\texttt{KeyGen}^*, \texttt{Enc}^*, \texttt{Dec}^*)$ by:

$\texttt{KeyGen}^*(1^\lambda)$**:** Output $pk = (pk_0, \texttt{CRS})$, $sk = sk_0$ where
$(pk_0, sk_0) \leftarrow \texttt{KeyGen}(1^\lambda)$, $(\texttt{CRS}, \texttt{TK}, \texttt{EK}) \leftarrow \texttt{Setup}(1^\lambda)$.

$\texttt{Enc}^*_{pk}(m; r)$**:** Output $C = (c, \pi)$ where $c \leftarrow \texttt{Enc}_{pk_0}(m; r)$, $\pi \leftarrow \texttt{Prove}_{\texttt{CRS}}((pk_0, c), (m, r))$.

$\texttt{Dec}^*_{sk}(C)$**:** Parse $C = (c, \pi)$. If the argument π verifies output $\texttt{Dec}_{sk}(c)$, else output $\perp$.

Theorem 4.4. *Assume that* $\mathcal{E}$ *is* ℓ*-LR-CPA secure, and* Π *is a strong one-time* f*-tSE NIZK argument for the relation* R_{enc} *where, for any witness* (m, r)*, we define* $f(m, r) = m$*. Then the scheme* $\mathcal{E}^*$ *defined above is* ℓ*-LR-CCA secure.*

The proof of Theorem 4.4 is given in Appendix A.3. We also note that, if the tSE NIZK construction allows labels, than we can naturally extend our construction above to yield a ℓ-LR-CCA encryption *with labels*, by simply putting the encryption labels into the NIZK proofs (and using them to verify the proofs).

4.4 True-Simulation f-Extractable (f-tSE) NIZK

Let f be any efficiently computable function, and let $R(x, y)$ be an NP relation. We show how to construct an f-tSE NIZK argument Ψ from any labeled CCA-secure encryption scheme, and (standard) NIZK arguments. Let $\mathcal{E} = (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$ be a CCA-secure encryption scheme supporting labels, and let $\Pi = (\texttt{Setup}_\Pi, \texttt{Prove}_\Pi, \texttt{Verify}_\Pi)$ be an NIZK argument for the relation

$$R_\Pi = \{ \; (\; (x, r) \; , \; (y, c, pk, L) \;) \mid R(x, y) = 1 \wedge c = \texttt{Enc}^L_{pk}(f(x); r) \; \}$$

We define f-tSE NIZK argument Ψ (supporting labels) as follows:

- $\texttt{Setup}(1^\lambda)$: Output $\texttt{CRS} = (\texttt{CRS}_\Pi, pk)$, $\texttt{TK} = \texttt{TK}_\Pi$, $\texttt{EK} = sk$ where $(pk, sk) \leftarrow \texttt{KeyGen}(1^\lambda)$, $(\texttt{CRS}_\Pi, \texttt{TK}_\Pi) \leftarrow \texttt{Setup}_\Pi(1^\lambda)$.
- $\texttt{Prove}^L(x, y; r)$: Output $\varphi = (c, \pi)$ where $c \leftarrow \texttt{Enc}^L_{pk}(f(x); r)$, $\pi \leftarrow \texttt{Prove}_\Pi((x, r), (y, c, pk, L))$.
- $\texttt{Verify}^L(y, \varphi)$: Parse $\varphi = (c, \pi)$ and run $\texttt{Verify}_\Pi((y, c, pk, L), \pi)$.

Theorem 4.5. *If $\mathcal{E}$ is a labeled CCA-secure encryption scheme and Π is an NIZK argument for relation R_Π, then Ψ is a f-tSE NIZK argument for relation R.*

The proof of Theorem 4.5 is given in Appendix A.4.

4.5 Comparison of Our Generic Constructions to Prior Work

The idea of using an SPR relation to construct a leakage-resilient hard relation was implicit in [4, 44], and explicitly described in [5] for the case of leakage-resilient one-way functions.

Our constructions of leakage-resilient CCA encryption and signatures from tSE NIZKs bear significant resemblance to prior constructions. In particular, we observe that an alternate construction of tSE NIZK to that of Section 4.4, could be achieved by using a CPA-secure encryption scheme instead of a CCA-secure one, and a ss-NIZK argument system [56] instead of a standard one. In fact, the resulting construction would yield an *any*-simulation extractable (aSE) NIZK argument. This instantiation of aSE NIZKs is implicitly used by [44], in their construction of leakage-resilient signature schemes. It is also used implicitly in the Naor-Yung "double-decryption" paradigm [50, 56, 47] for CCA security, which was also later used in [49] to construct leakage-resilient CCA-encryption. However, as we have seen, tSE is sufficient for constructing *both* leakage-resilient signatures and CCA-encryption and thus, the stronger notion of aSE is not needed. Furthermore, given the current state of efficient encryption schemes and NIZK, the difference in efficiency between ss-NIZK and standard NIZK is *significantly* greater than the difference between CCA and CPA-secure encryption[7], thus making tSE superior in both simplicity and efficiency.

We note that our construction of tSE NIZKs (based on CCA encryption and standard NIZKs) was implicitly used by [34] to construct signatures of group elements. It was also implicitly used by [13] to construct efficient CCA-secure encryption scheme with key-dependent message (KDM) security out of a CPA version of such scheme. Still, the abstraction of tSE has not been explicitly defined in prior work despite its apparent usefulness.

5 Instantiations

ASSUMPTIONS. We review several standard hardness assumptions on which we will base our constructions.

[7]Informally, the difference between CCA and CPA-secure encryption is only 2 group elements, whereas the size of a ss-NIZK proof is *more than twice* the size of a standard NIZK proof.

Decisional Diffie-Hellman (DDH). Let $\mathbb{G}$ be a group of primer order q. Let $g_1, g_2 \overset{\$}{\leftarrow} \mathbb{G}$ and $r, r_1, r_2 \overset{\$}{\leftarrow} \mathbb{Z}_q$. The decisional Diffie-Hellman (DDH) assumption states that the following two distributions are computationally indistinguishable: $(\mathbb{G}, g_1, g_2, g_1^{r_1}, g_2^{r_2})$ and $(\mathbb{G}, g_1, g_2, g_1^{r}, g_2^{r})$.

Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be groups of prime order q and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a non-degenerate efficiently computable bilinear map.

Symmetric External Diffie-Hellman (SXDH) [59, 10, 6, 29, 61]. The symmetric external Diffie-Hellman assumption (SXDH) is that the DDH problem is hard in *both* groups $\mathbb{G}_1$ and $\mathbb{G}_2$. The assumption is clearly invalid for symmetric pairings (when $\mathbb{G}_1 = \mathbb{G}_2$), but is believed to hold when there is no efficiently computable mapping between $\mathbb{G}_1$ and $\mathbb{G}_2$.

K-Linear [40, 60] and DLIN [10]. Let $\mathbb{G}$ be a group of primer order q and let $K \geq 1$ be constant. Let $g_0, g_1, \ldots, g_K \overset{\$}{\leftarrow} \mathbb{G}$ and $x_0, x_2, \ldots, x_K \overset{\$}{\leftarrow} \mathbb{Z}_q$. The K-Linear assumption states that the following two distributions are computationally indistinguishable: $(\mathbb{G}, g_0, g_1, \ldots, g_K, g_1^{x_1}, \ldots, g_K^{x_K}, g_0^{x_0})$, and $(\mathbb{G}, g_0, g_1, \ldots, g_K, g_1^{x_1}, \ldots, g_K^{x_K}, g_0^{X})$, where $X = \sum_{i=1}^{K} x_i$.

Note that for $K = 1$, the K-Linear is the same as DDH, and that it does not hold when working with symmetric pairings. In that setting, the 2-Linear assumption is usually assumed to hold, and is often referred to as the Decisional Linear (DLIN) assumption. *Throughout this paper we assume the K-Linear assumption holds in both $\mathbb{G}_1$ and $\mathbb{G}_2$, which is the case when working with symmetric pairings, and slightly abuse notation when $K = 1$ and assume SXDH holds in that case.*

OUR INSTANTIATIONS. We show efficient instantiations of the leakage-resilient signature and CCA-secure encryption constructions described in Sections 4.2 and 4.3, respectively. For each scheme, we give two instantiations based on bilinear maps: one secure under the symmetric external Diffie-Hellman (SXDH) assumption, and a second, secure under the Decision Linear (DLIN) assumption. The first can be used with asymmetric pairings, while the second applies to the case of symmetric pairings. We give details of all instantiations in Appendix C but give a high-level idea below.

Signatures. Recall that in order to instantiate the signature scheme from Section 4.2, we need a leakage-resilient hard relation R (which we will derive from an SPR relation) and a true-simulation extractable (tSE) NIZK argument, which we build from CCA-secure encryption and a standard NIZK argument for the relation $\{ \; (\; (x, r) \; , \; (y, c, pk, L) \;) \mid R(x, y) = 1 \wedge c = \mathtt{Enc}^{L}_{pk}(f(x); r) \; \}$. We show our choice of instantiations for these components:

- *CCA-Secure Encryption:* Under both the SXDH and DLIN assumptions, we use efficient encryption schemes in the style of Cramer-Shoup [17, 60].
- *NIZK Argument:* We use the Groth-Sahai proof system [36], which can be instantiated both under SXDH and DLIN. See Appendix B for a brief description of the proof system.
- *SPR Relation:* Previous constructions of leakage-resilient primitives use the SPR function $g_1^{x_1} g_2^{x_2} \ldots g_n^{x_n}$. However, this function has the problem that the witness lies in the exponent. This means that we cannot combine it with an encryption scheme for elements in $\mathbb{G}$ (unless each witness component is committed bit by bit which, among other things, results in proofs growing linearly with the security parameter), and unfortunately encryption schemes for messages in $\mathbb{Z}_q$ cannot be combined with the Groth-Sahai system. We therefore construct two new SPR relations based on pairing-product equations. For our SXDH instantiation, we use the relation $e(h_1, x_1)\; e(h_2, x_2) \ldots e(h_n, x_n) = e(y, \tilde{g})$, where $\tilde{g}$ is a generator of $\mathbb{G}_2$. We prove that this relation is SPR under the SXDH assumption. In the DLIN case, we use the relation:

$e(h_1, x_1)\ e(h_2, x_2) \dots e(h_n, x_n) = e(y_1, g)$, $e(\hbar_1, x_1)\ e(\hbar_2, x_2) \dots e(\hbar_n, x_n) = e(y_2, g)$, where g is a generator of $\mathbb{G}$. We prove that this relation is SPR under the DLIN assumption.

To achieve $(1-\epsilon)|sk|$ leakage resilience, we let n (the number of witness components) in the SPR relation be inversely proportional to ϵ.

Theorem 5.1. *Let $\mathbb{G}_1, \mathbb{G}_2$ be groups of primer order q. For any $\epsilon > 0$, there exists a $(1-\epsilon)|sk|$-leakage resilient signature scheme, secure under the SXDH assumption, using signatures consisting of $(9/\epsilon)(1 + \omega(\log \lambda)/\log q) + 24$ group elements and 2 elements in $\mathbb{Z}_q$. Similarly, for any $\epsilon > 0$, there exists a $(1-\epsilon)|sk|$-leakage resilient signature scheme, secure under the DLIN assumption, using signatures consisting of $(19/\epsilon)(2 + \omega(\log \lambda)/\log q) + 70$ group elements and 6 elements in $\mathbb{Z}_q$.*

CCA-Secure Encryption. Recall that for leakage-resilient encryption, we need leakage-resilient CPA-secure encryption, standard CCA-secure encryption and strong tSE NIZK, which we can get from combining regular tSE NIZK with a strong one-time signature. We build regular tSE NIZK from CCA-secure encryption and regular NIZK. We describe our choices for each of these below.

- *LR-CPA-Secure Encryption:* We construct a new leakage-resilient CPA-secure encryption scheme for our purpose in the style of ElGamal (similar to ones used in [49, 13] but making it more efficient).
- *CCA-Secure Encryption:* Under both the SXDH and DLIN assumptions, we use efficient encryption schemes in the style of Cramer-Shoup [17, 60].
- *NIZK Argument:* We use the Groth-Sahai proof system [36], which can be instantiated both under SXDH and DLIN. See Appendix B for a brief description of the proof system.
- *One-Time Signature:* We observe that *any* strong one-time signature secure under these assumptions can be used. Here, we opt for the scheme of [34], secure under the Discrete Log assumption (implied by both SDXH and DLIN), because its signature size is small, namely 2 elements in $\mathbb{Z}_q$.

The leakage that our new CCA-secure encryption tolerates is the same as the leakage for the CPA-secure scheme. Informally, we achieve $(1-\epsilon)|sk|$ leakage resilience in the CPA-secure scheme by increasing the number of generators used in the public key and ciphertext. This number will be inversely proportional to ϵ.

Theorem 5.2. *Let $\mathbb{G}_1, \mathbb{G}_2$ be groups of primer order q. For any $\epsilon > 0$, there exists a $(1-\epsilon)|sk|$-leakage resilient encryption scheme, secure under the SXDH assumption, using ciphertexts consisting of $(2/\epsilon)(2+\lambda/\log q)+15$ group elements and 2 elements in $\mathbb{Z}_q$. Similarly, for any $\epsilon > 0$, there exists a $(1-\epsilon)|sk|$-leakage resilient encryption scheme, secure under the DLIN assumption, using ciphertexts consisting of $(3/\epsilon)(3+\lambda/\log q)+34$ group elements and 2 elements in $\mathbb{Z}_q$.*

6 Other Applications

6.1 Leakage-Resilient ID Schemes

Recall that, in an identification scheme, an honest prover chooses a public/secret key pair (pk, sk) and publishes pk. An identification scheme is a protocol in which the *prover* uses sk to identify herself to a *verifier* that only knows pk. The security property of an identification scheme considers an adversary $\mathcal{A}$ that acts in two stages: a learning stage and an impersonation stage. In the learning stage, $\mathcal{A}$ repeatedly interacts with the honest prover while taking the role of a *malicious* verifier in an attempt to learn some non-trivial information about sk. In the impersonation stage, the honest prover "goes away" and $\mathcal{A}$ attempts to impersonate the prover's identity to an honest verifier. We say such a scheme is secure if the adversary has only a negligible probability of succeeding in the impersonation stage.

Leakage-resilient identification schemes were first studied and constructed in [4]. Two distinct notions of leakage-resilience were considered: *pre-impersonation* leakage and *anytime* leakage. In the former notion, the attacker can only get leakage on the secret key during the learning stage, while in the latter notion, the adversary might also get some additional leakage during the impersonation stage, possibly after seing some "challenges" from the verifier (see [4] for formal definitions). It was shown in [4] that the Okamoto identification scheme [51], and in fact any Σ-Protocol for an SPR hard-relation, is leakage-resilient. Moreover, for any $\epsilon > 0$, there is a generalization of the Okamoto ID scheme which is $(1-\epsilon)|sk|$-leakage-resilient for *pre-impersonation leakage*. Unfortunately, due to the rewinding nature of the security proof, the scheme was only shown to be $(1/2-\epsilon)|sk|$-leakage-resilient for anytime leakage.

We recall that a simple, well-known, identification-scheme based on signatures consists of the verifier choosing a random message m and the prover replying with $\texttt{Sign}_{sk}(m)$ which the verifier validates using pk. It is easy to see that this scheme is leakage-resilient in the *anytime leakage* setting as long as the signature scheme is leakage-resilient (with the same bound ℓ). Therefore, using our $(1-\epsilon)|sk|$-LR signature schemes from Section 5 we get an efficient identification scheme with optimal leakage-resilience in the anytime leakage model.

Theorem 6.1. *There exists a construction of ℓ-LR identification schemes w.r.t. anytime leakage from any ℓ-LR signature scheme, preserving the public-key size, secret-key size, and efficiency of the underlying signature.*

Interestingly, another well-known identification-scheme based on CCA-encryption, consists of the verifier encrypting a random message m and sending $c = \texttt{Enc}_{pk}(m)$ to the prover who decrypts and replies with m. Although this scheme seems secure w.r.t. pre-impersonation leakage if the encryption scheme is LR-CCA secure, it does not seem secure w.r.t. anytime leakage, since, in this setting, the leakage on the encryption secret-key *can* depend on the ciphertext.

6.2 Leakage-Resilient Authenticated Key Agreement

Using our leakage-resilient signature scheme from Section 4.2 and our leakage-resilient CCA-secure encryption scheme from Section 4.3 (and instantiating them as described in Appendix C), we construct two $(1-\epsilon)|sk|$-leakage resilient authenticated key agreement (AKA) schemes. We prove perfect forward security in the unauthenticated-links model. We refer the reader to [15, 4] for a detailed description of the model and definitions of security, but give a high level idea of the problem and solution below.

MODEL AND SECURITY DEFINITIONS. We consider the problem of two parties, Alice and Bob, who need to establish a shared cryptographic key in the presence of an adversary, and want to have the guarantee that the privacy of such key is conserved. At the same time, Alice wants to be sure that she has exchanged a key with Bob, and similarly, Bob wants to be sure that he has indeed exchanged a key with Alice (and not an adversarial third party). In the leakage setting, the adversary is a "man-in-the-middle" attacker that has the power to learn arbitrary information about Alice's and Bob's long-term secrets sk_A, sk_B. We model this by giving the adversary access to leakage oracles $\mathcal{O}^{\lambda,\ell}_{sk_A}, \mathcal{O}^{\lambda,\ell}_{sk_B}$, which he can access *before* the key-agreement execution but not during. The adversary is also able to observe (and possibly intervene in) key exchanges between Alice and/or Bob, and other parties. Our constructions satisfy the notion of *perfect forward security*, which guarantees that the privacy of a key is conserved even if the adversary learns the *entire* long-term secret keys sk_A, sk_B after the exchange had been completed and the key has been deleted from memory.

OUR CONSTRUCTIONS. Our first construction follows from directly applying the general result of [4], who show that any leakage-resilient signature scheme is sufficient to achieve leakage-resilient AKA. The protocol eSig-DH of [4] is simply the (passive) Diffie-Hellman key agreement, authenticated with a signature scheme: a party authenticates to his peer by signing the message he received from him. Our second construction of leakage-resilient AKA (shown in Figure 1) is based on leakage-resilient CCA-secure encryption. This new protocol, which we refer to as Enc-DH, is a modification of the Diffie-Hellman key agreement protocol, in which both parties authenticate to each other by correctly decrypting a ciphertext encrypted with their corresponding public key. Intuitively, this achieves authentication since given a ciphertext encrypted under a certain public key, only the party in possession of the corresponding secret key is able to correctly decrypt the ciphertext.

Our second construction also satisfies *deniability*. Informally, this means that without knowing the long-term secrets of the parties participating in an execution of the protocol, it is possible to simulate a transcript of the execution that is computationally indistinguishable from the real transcript. We do not formalize the notion of deniability here, but it is easy to see that simulating a transcript of an Enc-DH execution can be achieved by simply choosing all internal state variables and encrypting them using the parties' public keys. Notice that AKA schemes that use signatures (in particular, the eSig-DH construction of [4]) do not satisfy deniability, since we cannot "simulate" a signature without knowing the signing key (which is the long-term key of the protocol).

The protocol Enc-DH can also be used in the leakage-free setting using standard (not necessarily leakage-resilient) CCA-secure encryption. To the best of our knowledge this construction is *new* and is therefore of independent interest.

Theorem 6.2. *Let* $\mathcal{E} = (\texttt{KeyGen}, \texttt{Enc}, \texttt{Dec})$ *be an* ℓ*-leakage resilient CCA-secure encryption scheme supporting labels. Then* Enc-DH *is an* ℓ*-SK-secure key agreement protocol with perfect forward security in the unauthenticated-links model under the DDH assumption.*

We notice that both eSig-DH and Enc-DH preserve the leakage-tolerance of the underlying signature and encryption scheme, respectively. Thus, plugging in our $(1-\epsilon)|sk|$-leakage resilient signature scheme into eSig-DH and our $(1-\epsilon)|sk|$-leakage resilient CCA-secure encryption scheme into Enc-DH yields two different constructions of $(1-\epsilon)|sk|$-leakage resilient AKA. As described above, the latter construction also satisfies deniability.

References

[1] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-preserving signatures and commitments to group elements. In *CRYPTO*, 2010. To Appear.

[2] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In O. Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.

[3] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs. Public-key encryption in the bounded-retrieval model. Cryptology ePrint Archive, Report 2009/512. To Appear at Eurocrypt, 2010.

[4] J. Alwen, Y. Dodis, and D. Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In Halevi [38], pages 36–54.

[5] J. Alwen, Y. Dodis, and D. Wichs. Survey: Leakage resilience and the bounded retrieval model. In *ICITS*, 2009.

[6] L. Ballard, M. Green, B. de Medeiros, and F. Monrose. Correlation-resistant storage via keyword-searchable encryption. Cryptology ePrint Archive, Report 2005/417, 2005.

[7] M. Bellare, A. Boldyreva, and J. Staddon. Randomness re-use in multi-recipient encryption schemeas. In Y. Desmedt, editor, *Public Key Cryptography*, volume 2567 of *Lecture Notes in Computer Science*, pages 85–99. Springer, 2003.

[8] E. Biham and A. Shamir. Differential fault analysis of secret key cryptosystems. In B. S. K. Jr., editor, *CRYPTO*, volume 1294 of *Lecture Notes in Computer Science*, pages 513–525. Springer, 1997.

[9] M. Blum, P. Feldman, and S. Micali. Non-interactive zero-knowledge and its applications (extended abstract). In *STOC*, pages 103–112. ACM, 1988.

[10] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO*, pages 41–55, 2004.

[11] D. Boneh, R. A. DeMillo, and R. J. Lipton. On the importance of checking cryptographic protocols for faults (extended abstract). In *EUROCRYPT*, pages 37–51, 1997.

[12] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision diffie-hellman. In D. Wagner, editor, *CRYPTO*, volume 5157 of *Lecture Notes in Computer Science*, pages 108–125. Springer, 2008.

[13] J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In A. Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 351–368. Springer, 2009.

[14] R. Canetti, Y. Dodis, S. Halevi, E. Kushilevitz, and A. Sahai. Exposure-resilient functions and all-or-nothing transforms. In *EUROCRYPT*, pages 453–469, 2000.

[15] R. Canetti and H. Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In B. Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer, 2001.

[16] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *STOC*, pages 494–503, 2002.

[17] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In H. Krawczyk, editor, *CRYPTO*, volume 1462 of *Lecture Notes in Computer Science*, pages 13–25. Springer, 1998.

[18] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In L. R. Knudsen, editor, *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 45–64. Springer, 2002.

[19] G. D. Crescenzo, R. J. Lipton, and S. Walfish. Perfectly secure password protocols in the bounded retrieval model. In Halevi and Rabin [39], pages 225–244.

[20] Y. Dodis, S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *TCC*, pages 361–381, 2010.

[21] Y. Dodis, Y. T. Kalai, and S. Lovett. On cryptography with auxiliary input. In *STOC*, pages 621–630, 2009.

[22] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.

[23] Y. Dodis, A. Sahai, and A. Smith. On perfect and adaptive security in exposure-resilient cryptography. In *EUROCRYPT*, pages 301–324, 2001.

[24] S. Dziembowski. Intrusion-resilience via the bounded-storage model. In Halevi and Rabin [39], pages 207–224.

[25] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302, 2008.

[26] S. Faust, E. Kiltz, K. Pietrzak, and G. N. Rothblum. Leakage-resilient signatures. In *TCC*, pages 343–360, 2010.

[27] S. Faust, T. Rabin, L. Reyzin, E. Tromer, and V. Vaikuntanathan. Protecting against computationally bounded and noisy leakage. In *EUROCRYPT*, 2010. To Appear.

[28] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.

[29] S. D. Galbraith and V. Rotger. Easy decision-diffie-hellman groups. *LMS Journal of Computation and Mathematics*, 7:2004, 2004.

[30] K. Gandolfi, C. Mourtel, and F. Olivier. Electromagnetic analysis: Concrete results. In Çetin Kaya Koç, D. Naccache, and C. Paar, editors, *CHES*, volume 2162 of *Lecture Notes in Computer Science*, pages 251–261. Springer, 2001.

[31] S. Goldwasser, Y. Kalai, C. Peikert, and V. Vaikuntanathan. Robustness of the learning with errors assumption. In *Innovations in Computer Science (ICS)*, 2010.

[32] S. Goldwasser and Y. T. Kalai. On the (in)security of the fiat-shamir paradigm. In *FOCS*, pages 102–, 2003.

[33] J. Groth. Homomorphic trapdoor commitments to group elements. Cryptology ePrint Archive, Report 2009/007, updated version available at `http://www.brics.dk/~jg/`.

[34] J. Groth. Simulation-sound nizk proofs for a practical language and constant size group signatures. In X. Lai and K. Chen, editors, *ASIACRYPT*, volume 4284 of *Lecture Notes in Computer Science*, pages 444–459. Springer, 2006.

[35] J. Groth. Fully anonymous group signatures without random oracles. In *ASIACRYPT*, pages 164–180, 2007.

[36] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In N. P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 415–432. Springer, 2008.

[37] 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. *Commun. ACM*, 52(5):91–98, 2009.

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

[39] S. Halevi and T. Rabin, editors. *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*. Springer, 2006.

[40] D. Hofheinz and E. Kiltz. Secure hybrid encryption from weakened key encapsulation. In *CRYPTO*, pages 553–571, 2007.

[41] Y. Ishai, A. Sahai, and D. Wagner. Private circuits: Securing hardware against probing attacks. In *CRYPTO*, pages 463–481, 2003.

[42] A. Juma, C. Rackoff, and Y. Vahlis. Leakage resilient key proxies.

[43] J. Kamp and D. Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. In *FOCS*, pages 92–101, 2003.

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

[45] P. C. Kocher. Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems. In N. Koblitz, editor, *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 104–113. Springer, 1996.

[46] P. C. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *CRYPTO*, pages 388–397, 1999.

[47] Y. Lindell. A simpler construction of cca2-secure public-keyencryption under general assumptions. *J. Cryptology*, 19(3):359–377, 2006.

[48] S. Micali and L. Reyzin. Physically observable cryptography (extended abstract). In *TCC*, pages 278–296, 2004.

[49] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. In Halevi [38], pages 18–35.

[50] M. Naor and M. Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*, pages 427–437. ACM, 1990.

[51] T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In *CRYPTO*, pages 31–53, 1992.

[52] R. Pass and A. Rosen. Concurrent non-malleable commitments. In *FOCS*, pages 563–572. IEEE Computer Society, 2005.

[53] R. Pass and A. Rosen. New and improved constructions of non-malleable cryptographic protocols. In H. N. Gabow and R. Fagin, editors, *STOC*, pages 533–542. ACM, 2005.

[54] K. Pietrzak. A leakage-resilient mode of operation. In *Eurocrypt 2009, Cologne, Germany*, 2009.

[55] J.-J. Quisquater and D. Samyde. Electromagnetic analysis (ema): Measures and counter-measures for smart cards. In I. Attali and T. P. Jensen, editors, *E-smart*, volume 2140 of *Lecture Notes in Computer Science*, pages 200–210. Springer, 2001.

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

[57] A. D. Santis, G. D. Crescenzo, R. Ostrovsky, G. Persiano, and A. Sahai. Robust non-interactive zero knowledge. In J. Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 566–598. Springer, 2001.

[58] A. D. Santis and G. Persiano. Zero-knowledge proofs of knowledge without interaction (extended abstract). In *FOCS*, pages 427–436. IEEE, 1992.

[59] M. Scott. Authenticated id-based key exchange and remote log-in with simple token and pin number. Cryptology ePrint Archive, Report 2002/164, 2002.

[60] H. Shacham. A cramer-shoup encryption scheme from the linear assumption and from progressively weaker linear variants, 2007. Cryptology ePrint Archive, Report 2007/074.

[61] E. R. Verheul. Evidence that xtr is more secure than supersingular elliptic curve cryptosystems. *J. Cryptology*, 17(4):277–296, 2004.

A Proofs of Theorems

A.1 Hard Relations

Before proving Theorem 4.2, we write a couple of definitions and a lemma that we will use in the proof.

Definition A.1 (Min-Entropy)**.** *The* min-entropy *of a random variable X, denoted as $\mathbf{H}_\infty(X)$ is:* $\mathbf{H}_\infty(X) = -\log(\max_x \Pr[X = x])$.

Definition A.2 (Average-Conditional Min-Entropy [22])**.** *The* average-conditional min-entropy *of a random variable X conditioned on Z, denoted as $\widetilde{\mathbf{H}}_\infty(X|Z)$ is:*

$$\widetilde{\mathbf{H}}_\infty(X|Z) = -\log\left(\mathbb{E}_{z\leftarrow Z}\left[\max_x \Pr[X = x|Z = z]\right]\right) = -\log\left(\mathbb{E}_{z\leftarrow Z}\left[2^{\mathbf{H}_\infty[X|Z=z]}\right]\right)$$

Lemma A.3 ([22])**.** *Let X, Y, Z be random variables where Z takes values in a set of size at most 2^ℓ. Then $\widetilde{\mathbf{H}}_\infty(X|(Y, Z)) \geq \widetilde{\mathbf{H}}_\infty((X, Y)|Z) - \ell \geq \widetilde{\mathbf{H}}_\infty(X|Z) - \ell$, and in particular, $\widetilde{\mathbf{H}}_\infty(X|Y) \geq \mathbf{H}_\infty(X) - \ell$*

We now proceed to prove Theorem 4.2.

Proof of Theorem 4.2: We assume, for the sake of contradiction, that there exists an adversary $\mathcal{A}$ that succeeds in breaking the security of leakage-resilient hard relation R with non-negligible probability ϵ. We construct $\mathcal{B}$ that breaks the security of the SPR relation with non-negligible probability.

On input (x, y), $\mathcal{B}$ emulates $\mathcal{A}$ on input y, responds to $\mathcal{A}$'s leakage queries using x. When $\mathcal{A}$ eventually outputs x^*, $\mathcal{B}$ also outputs x^*.

We know that $\Pr[R(x^*, y) = 1] = \epsilon$ but we need to compute $\Pr[x^* \neq x]$ since $\mathcal{B}$ only breaks the SPR property if $x^* \neq x$. Notice that:

$$\Pr[\mathcal{B} \text{ succeeds}] = \Pr[\mathcal{A} \text{ succeeds} \wedge x \neq x^*] \geq \Pr[\mathcal{A} \text{ succeeds}] - \Pr[x = x^*] = \epsilon - \Pr[x = x^*]$$

Notice that the only information that $\mathcal{A}$ has about x comes from y and the leakage queries. Let X, Y be the random variables for x, y respectively, and let Z be the random variable for the total leakage learned by $\mathcal{A}$. Then $\widetilde{\mathbf{H}}_\infty(X|(Y, Z)) \geq \widetilde{\mathbf{H}}_\infty(X|Y) - \ell$ and

$$\Pr[x = x^*] \leq 2^{-\widetilde{\mathbf{H}}_\infty(X|Y)+\ell} = 2^{-\mathbf{H}_{avg}(R)+\ell}.$$

Assuming that $\ell \leq \mathbf{H}_{avg}(R) - \omega(\log(\lambda))$ we have that $\Pr[\mathcal{B} \text{ succeeds}] \geq \epsilon - 2^{-\omega(\log(\lambda)}$, which is non-neglibible. □

A.2 Signatures

Proof of Theorem 4.3: Consider the following series of games.

Game 0: This is the leakage-resilient game in Definition 2.2. Let $(m^*, \sigma^* = \varphi^*)$ be the message/signature pair that $\mathcal{A}$ outputs.

Game 1: We change the signing oracle in the way it answers $\mathcal{A}$'s queries. Instead of giving a valid argument φ, it answers query m with a simulated proof $\mathtt{Sim}(\mathtt{TK}, y, m)$. Game 0 and Game 1 are indistinguishable by the *zero-knowledge* of Π. Notice that the simulated arguments given to $\mathcal{A}$ as answers to leakage queries are always of true statements. As in the previous game, the winning condition is that $\mathcal{A}$ produces a valid forgery (m^*, σ^*), i.e. $\mathtt{Verify}^{m^*}(y, \sigma^*) = 1$ and m^* was not part of a signature query.

Game 2: We change the winning condition: we say that $\mathcal{A}$ wins iff it produces a valid forgery (m^*, σ^*) *and* $R(z^*, y) = 1$ where $z^* \leftarrow \mathtt{Ext}^{m^*}(y, \varphi^*, \mathtt{EK})$. Game 1 and Game 2 are indistinguishable by the *true-simulation extractability* of Π.

We have proven that $|\Pr_2[\mathcal{A}\ wins] - \Pr_0[\mathcal{A}\ wins]| \le negl(\lambda)$. We need to show that $\Pr_2[\mathcal{A}\ wins] \le negl(\lambda)$. But notice that if $\Pr_2[\mathcal{A}\ wins]$ is non-negligible then this violates the security of the leakage-resilient hard relation R. In other words, we can create an adversary $\mathcal{B}$ that on input y, generates $(\mathtt{CRS}, \mathtt{TK}, \mathtt{EK}) \leftarrow \mathtt{Setup}(1^\lambda)$ and emulates $\mathcal{A}$ on input $pk \leftarrow (\mathtt{CRS}, y)$. $\mathcal{B}$ answers $\mathcal{A}$'s leakage queries using the leakage oracle $\mathcal{O}_{sk}^{\lambda,\ell}(\cdot)$ and answers signing queries m_i by creating simulated arguments $\mathtt{Sim}(\mathtt{TK}, y, m_i)$. Eventually, $\mathcal{A}$ will output a forgery $(m^*, \sigma^* = \varphi^*)$. $\mathcal{B}$ runs $\mathtt{Ext}^{m^*}(y, \varphi^*, \mathtt{EK}) \to z^*$ and outputs z^*. Notice that the probability that $\mathcal{B}$ outputs z^* such that $R(z^*, y) = 1$ (thus breaking the hardness of R) is exactly $\Pr_2[\mathcal{A}\ wins]$. Therefore, we must have that $\Pr_2[\mathcal{A}\ wins] \le negl(\lambda)$. □

A.3 CCA-Secure Encryption

Proof of Theorem 4.4: We do a series of games argument to prove the above theorem. The games are all variants of the ℓ-LR-CCA game, and in all of the games, the adversary gets correctly generated $pk = (pk^*, \mathtt{CRS})$ and adversarial leakage queries are answered using the correctly generated secret key sk. The games will differ in how the challenge ciphertext C is generated, and how the challenger answers decryption queries of ciphertexts $\tilde{C}_i = (\tilde{c}_i, \tilde{\pi}_i)$.

Game 1: This is the original ℓ-LR-CCA attack game (in definition 2.3) against the scheme $\mathcal{E}^*$ where the challenge ciphertext and the decryption queries are generated/answered correctly. In other words:

$$\text{Challenge: } c \leftarrow \mathtt{Enc}_{pk}(m_b; r), \pi \leftarrow \mathtt{Prove}_{\mathtt{CRS}}(\ (pk, c)\ ,\ (m, r)). \quad \text{Decrypt using: } \mathtt{Dec}_{sk}(\tilde{c}_i).$$

where m_b is one of the messages m_0, m_1 chosen by the adversary, and b is chosen randomly by the challenger.

Game 2: In this game the $\mathtt{CRS}$ for Π is generated together with a simulation trapdoor $\mathtt{TK}$ and the arguments π are simulated using $\mathtt{Sim}_{\mathtt{TK}}(pk, c)$ so that:

$$\text{Challenge: } c \leftarrow \mathtt{Enc}_{pk}(m_b; r), \pi \leftarrow \mathtt{Sim}_{\mathtt{TK}}(pk, c). \quad \text{Decrypt using: } \mathtt{Dec}_{sk}(\tilde{c}_i).$$

Games 1 and 2 are indistinguishable by the *NIZK* property of the argument Π.

Game 3: In this game the $\mathtt{CRS}$ for Π is generated together with a simulation trapdoor $\mathtt{TK}$ and an extraction trapdoor $\mathtt{EK}$. The decryption queries $\tilde{C}_i = (\tilde{c}_i, \tilde{\pi}_i)$ are answered by running the extractor on the arguments $\tilde{\pi}_i$ to extract $f(m_i, r_i) = m_i$.

$$\text{Challenge: } c \leftarrow \mathtt{Enc}_{pk}(m_b; r), \pi \leftarrow \mathtt{Sim}_{\mathtt{TK}}(pk, c). \quad \text{Decrypt using: } \mathtt{Ext}((pk, \tilde{c}_i), \tilde{\pi}_i, \mathtt{EK}).$$

Games 2 and 3 are indistinguishable by the strong one-time true-simulation f-extractability of Π. This is because the adversary only gets a single simulated argument of a *true* statement (pk, c), and therefore cannot produce any new statement, argument pair $(\tilde{c}_i, \tilde{\pi}_i) \ne (c, \pi)$ for which the argument $\tilde{\pi}_i$ verifies but the extractor fails to extract the correct m_i.

Game 4: In this game, the challenge ciphertext c is generated by encrypting the message 0 [8] so that:

$$\text{Challenge: } c \leftarrow \mathtt{Enc}_{pk}(0; r), \pi \leftarrow \mathtt{Sim}_{\mathtt{TK}}(pk, c). \quad \text{Decrypt using: } \mathtt{Ext}((pk, \tilde{c}_i), \tilde{\pi}_i, \mathtt{EK}).$$

Games 3 and 4 are indistinguishable by the *ℓ-LR CPA* security of $\mathcal{E}$. Recall that leakage queries are always answered using sk and so we need to rely on leakage-resilience here. However, CPA security now suffices since the decryption secret-key sk is never used otherwise in Games 3,4.

Notice that Game 4 is completely independent of the challenger's bit b, and hence the advantage of any adversary in Game 4 is exactly 0 (the probability of guessing b is exactly $\frac{1}{2}$). Therefore, the advantage of any adversary in Game 0 must be at most $negl(\lambda)$, since the games are indistinguishable, which concludes the proof. □

[8]. . . or any fixed message in the message domain

A.4 True-Simulation f-Extractable (f-tSE) NIZK

Proof of Theorem 4.5: Correctness and soundness follow from the correctness and soundness properties of Π. We show that the zero-knowledge and true-simulation extractability hold as well.

Zero-Knowledge. We construct $\mathtt{Sim}$ as follows: On input $(\mathtt{TK}, y)$ and label L, $\mathtt{Sim}$ lets $c \leftarrow \mathtt{Enc}^L(0)$ and $\pi \leftarrow \mathtt{Sim}_\Pi(y, c, pk, L)$, and outputs $\varphi = (c, \pi)$. By the *CCA-security* of $\mathcal{E}$ and the *zero-knowledge* of Π, we have that the distribution of a simulated argument $\mathtt{Sim}^L(\mathtt{TK}, y)$ is computationally indistinguishable from a real argument $\mathtt{Prove}^L(x, y; r)$.

True-Simulation Extractability. We construct $\mathtt{Ext}$ as follows: On input $(y, \varphi = (c, \pi), \mathtt{EK} = sk)$ and label L, $\mathtt{Ext}$ lets $x \leftarrow \mathtt{Dec}^L_{sk}(c)$ and outputs x. Consider the following sequence of games:

Game 0: This is the game described in Definition 3.1. Let $(x_1, y_1), \ldots, (x_q, y_q)$ be P^*'s simulation queries, and let $(y^*, L^*, \varphi^* = (c^*, \pi^*))$ be the output of P^*. Note that the challenger uses x_j only to check $R(x_j, y_j)$; in other words, the answer $\varphi_j = (c_j, \pi_j)$ to query (x_j, y_j, L_j) is a simulated argument and therefore contains an encryption of 0 (not of $f(x_j)$).

Game $1.i$ (for $i = 1, \ldots, q$): We change the simulation oracle so that in Game $1.i$, for $j \leq i$ the oracle answers query (x_j, y_j, L_j) as follows: if $R(x_j, y_j) = 0$ the challenger returns $\perp$ as before, but if $R(x_j, y_j) = 1$ it lets $c_j \leftarrow \mathtt{Enc}^{L_j}(x_j)$ and $\pi_j \leftarrow \mathtt{Sim}_\Pi(y_j, c_j, pk, L_j)$, and outputs $\varphi_j = (c_j, \pi_j)$. Games 0 and 1.1, and Games $1.i$ and $1.(i+1)$ for $i = 1, \ldots, q-1$ are indistinguishable by the *CCA-security* of $\mathcal{E}$. This is because if adversary $\mathcal{A}$ can distinguish between them, we could construct adversary $\mathcal{B}$ that given pk runs $(\mathtt{CRS}_\Pi, \mathtt{TK}_\Pi, \mathtt{EK}_\Pi) \leftarrow \mathtt{Setup}_\Pi(1^\lambda)$ and emulates $\mathcal{A}$ on $\mathtt{CRS} = (pk, \mathtt{CRS}_\Pi)$. Notice that we need to rely on the stronger notion of CCA-security (instead of CPA-security) since $\mathcal{B}$ needs to decrypt the ciphertext c^* from $\mathcal{A}$'s output in order to extract a value z^* and check the f-tSE winning condition.

Game 2: We change the simulator oracle so that the challenger answers query (x_j, y_j, L_j) as follows: if $R(x_j, y_j) = 0$ the challenger returns $\perp$ as before, but if $R(x_j, y_j) = 1$ it lets $c_j \leftarrow \mathtt{Enc}^{L_j}(f(x_j))$ and $\pi_j \leftarrow \mathtt{Prove}_\Pi(x_j, (y_j, c_j, pk, L_j))$, and outputs $\varphi_j = (c_j, \pi_j)$. Games 2 and $1.q$ are indistinguishable by the *zero-knowledge* of Π.

Notice that if adversary $\mathcal{A}$ wins Game 2, then it must be the case that $\mathtt{Verify}^{L^*}(y^*, \varphi^*) = 1$. But if this is the case then by *soundness* of Π we have that with high probability $R(x^*, y^*) = 1$. Otherwise, we could construct an adversary $\mathcal{B}$ that on input $\mathtt{CRS}_\Pi$, computes $(sk, pk) \leftarrow \mathtt{KeyGen}(1^\lambda)$ and emulates $\mathcal{A}$ on $\mathtt{CRS} = (pk, \mathtt{CRS}_\Pi)$, answering simulation queries by encrypting $f(x_j)$ and running $\mathtt{Prove}_\Pi(x_j, (y_j, c_j, pk, L_j))$ on its own. When $\mathcal{A}$ eventually outputs $(y^*, L^*, \varphi^* = (c^*, \pi^*))$, $\mathcal{B}$ outputs φ^*. Since we assume that Π is sound, we must have that $\Pr_2[\mathcal{A}\ wins] \leq negl(\lambda)$ and it follows that $\Pr_0[\mathcal{A}\ wins] \leq negl(\lambda)$. This concludes the proof of the theorem. □

A.5 Authenticated Key Agreement

We prove that the AKA protocol Enc-DH in Figure 1 has perfect forward security in the unauthenticated-links model. We refer the reader to [15, 4] for a detailed description of the model and definitions of security, but give a high level idea of the problem and solution below.

MODEL AND SECURITY DEFINITIONS. We prove security in the unauthenticated-links model with erasures of [15] (with the modifications of [4]), where we consider a "man-in-the-middle" adversary that plays against concurrent sessions of the protocol between n players $\mathcal{P}_1, \ldots, \mathcal{P}_n$. We allow the adversary to schedule the start of each session and determine its participants. We also give the adversary the power to corrupt players, perform leakage queries on their long-term secrets (via n leakage oracles $\mathcal{O}^{\lambda,\ell}_{\mathsf{sk}_i}$), and learn their ephemeral states. The

Public Parameters: $\mathbb{G}$ be a DDH group with generator g and order q.
Common Input: public keys $(\mathsf{pk}_i, \mathsf{pk}_j)$ for encryption

Initiator $\mathcal{P}_i(\mathsf{sk}_i)$		Responder $\mathcal{P}_j(\mathsf{sk}_j)$
$a \xleftarrow{\$} \mathbb{Z}_q$, $\alpha = g^a$ $C_i = \mathtt{Enc}^{\mathcal{P}_i}_{\mathsf{pk}_j}(\alpha)$ register session $(\mathcal{P}_j, \alpha)$	$\xrightarrow{\mathcal{P}_i, C_i}$	
	$\xleftarrow{\mathcal{P}_i, \mathcal{P}_j, C_j}$	$b \xleftarrow{\$} \mathbb{Z}_q$, $\beta = g^b$ $\hat{\alpha} = \mathtt{Dec}^{\mathcal{P}_i}_{\mathsf{sk}_j}(C_i)$ $C_j = \mathtt{Enc}^{\mathcal{P}_i,\mathcal{P}_j}_{\mathsf{pk}_i}(\hat{\alpha}, \beta)$ register session $(\mathcal{P}_i, \hat{\alpha}, \beta)$
$(\hat{\alpha}, \hat{\beta}) = \mathtt{Dec}^{\mathcal{P}_i,\mathcal{P}_j}_{\mathsf{sk}_i}(C_j)$ output peer = $\mathcal{P}_j$, sid = $(\alpha, \hat{\beta})$, output session key $\gamma_i = \hat{\beta}^a$ delete a	$\xrightarrow{\mathcal{P}_i, \hat{\beta}}$	
mark session complete		output peer =$\mathcal{P}_i$, sid =$(\hat{\alpha}, \beta)$ output session key $\gamma_j = \hat{\alpha}^b$ delete b mark session complete

Sanity Checks:

- If $\mathcal{P}_i$ receives a round-2 message $(\mathcal{P}_i, \mathcal{P}_j, C_j)$ where $\mathtt{Dec}^{\mathcal{P}_i,\mathcal{P}_j}_{\mathsf{sk}_i}(C_j) = (\hat{\alpha}, \hat{\beta})$ but has not registered a session $(\mathcal{P}_j, \hat{\alpha})$ then $\mathcal{P}_i$ ignores the message. Similarly, if $\mathcal{P}_j$ receives a round-3 message $(\mathcal{P}_i, \hat{\beta})$ but has not registered a session $(\mathcal{P}_i, \hat{\beta})$ then $\mathcal{P}_j$ ignores the message.
- If $\mathcal{P}_j$ receives a round-1 message $(\mathcal{P}_i, C_i)$ and the decryption of C_i fails then $\mathcal{P}_j$ ignores the message. Similarly, if $\mathcal{P}_i$ receives a round-2 message $(\mathcal{P}_i, \mathcal{P}_j, C_j)$ and the decryption of C_j fails then $\mathcal{P}_i$ ignores the message.

Figure 1: Protocol Enc-DH

goal of the adversary is to learn the session key for a test session of its choice, performed between players $\mathcal{P}_i$ and $\mathcal{P}_j$, also chosen by the adversary. We do not allow the adversary to corrupt $\mathcal{P}_i, \mathcal{P}_j$ or to learn their ephemeral states during the test session, as this compromises the underlying Diffie-Hellman key agreement protocol. In terms of leakage, we require that before the test session, the adversary learns at most ℓ bits of information from leakage queries, and does not perform any leakage queries during the test session.

Enc-DH also satisfies *perfect forward security*, which guarantees that the privacy of a session key is conserved even if the adversary learns the entire long-term secret keys of the participating parties after the session is complete and the session key has been deleted from their memory.

Proof of Theorem 6.2: To prove that the construction in Figure 1 is an ℓ-leakage resilient authenticated key agreement scheme, we prove that it satisfies the completeness and privacy properties.

Completeness: Consider two uncorrupted parties $\mathcal{P}_i, \mathcal{P}_j$. If their two sessions are matching, we have that $(\hat{\alpha}, \beta) = (\alpha, \hat{\beta})$ and so $\hat{\alpha} = \alpha$ and $\hat{\beta} = \beta$. Therefore $\gamma_i = \beta^a = g^{ab}$ and $\gamma_j = \alpha^b = g^{ab}$.

Privacy: We follow the approach of [4]. Let $\mathcal{A}$ be an adversary attacking Enc-DH and consider the following two cases:

1. There is a non-negligible probability that in a winning test session $\mathcal{A}$ produces a round-2 or round-3

message that passes the sanity check performed by the owner (thus allowing $\mathcal{A}$ to impersonate the peer and learn the owner's session key).

2. There is a negligible probability that in a winning test session $\mathcal{A}$ produces a round-2 or round-3 message that passes the sanity check performed by the owner.

In the first case, we prove that the privacy of Enc-DH reduces to the CCA-security of $\mathcal{E}$. In the second case, we prove that the privacy of Enc-DH, reduces to the DDH assumption.

Claim A.4. *Let $\mathcal{A}$ be an adversary attacking* Enc-DH. *If there is a non-negligible probability ϵ that in a* winning test session $\mathcal{A}$ *produces a round-2 or round-3 message that passes the sanity check performed by the* owner *(therefore breaking the privacy of* Enc-DH*), then there exists an attacker $\mathcal{B}_1$ that breaks the CCA-security of $\mathcal{E}$ with probability polynomial in ϵ.*

Proof. Let Q be an upper bound on the number of sessions started by $\mathcal{A}_1$. We construct $\mathcal{B}_1$ breaking the CCA-security of $\mathcal{E}$.

- $\mathcal{B}_1$ receives pk from the CCA challenger, chooses $r \xleftarrow{\$} [1, Q]$ and $\mathcal{P} \xleftarrow{\$} \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ (it guesses that the rth session will be the test session and that $\mathcal{P}$ will be the peer).
- $\mathcal{B}_1$ runs $\mathcal{A}_1$ against $\{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ with the modification that it publishes pk for $\mathcal{P}$ and uses the CCA decryption oracle to decrypt incoming messages, and the CCA leakage oracle to answer leakage queries about sk.
- If the rth session is not the test session or $\mathcal{P}$ is not the peer, then $\mathcal{B}_1$ halts. Otherwise, we consider two cases:
  - If $\mathcal{P}$ is the responder: $\mathcal{B}_1$ chooses $\alpha_0, \alpha_1 \xleftarrow{\$} \mathbb{Z}_q$ and sends them to the CCA challenger along with label $\mathcal{P}_i$, and receives the challenge ciphertext $c^* = \mathtt{Enc}^{\mathcal{P}_i}(\alpha_b)$. It sends c^* as the round-1 message and receives $\hat{\alpha}$ as part of the round-2 message. (Notice that $\mathcal{B}_1$ knows sk_i so it is able to decrypt the ciphertext sent in round-2). $\mathcal{B}_1$ outputs $\hat{b}$ such that $\hat{\alpha} = \alpha_{\hat{b}}$.
  - If $\mathcal{P}$ is the initiator: $\mathcal{B}_1$ receives round-1 message containing $\hat{\alpha}$ and chooses $\beta_0, \beta_1 \xleftarrow{\$} \mathbb{Z}_q$. It sends $(\hat{\alpha}, \beta_0), (\hat{\alpha}, \beta_1)$ to the CCA challenger along with label $(\mathcal{P}_i, \mathcal{P}_j)$, and receives the challenge ciphertext $c^* = \mathtt{Enc}^{\mathcal{P}_i, \mathcal{P}_j}(\hat{\alpha}, \beta_b)$. It sends c^* as the ciphertext in round-2, and receives $\hat{\beta}$ as part of the round-3 message. $\mathcal{B}_1$ outputs $\hat{b}$ such that $\hat{\beta} = \beta_{\hat{b}}$.

We now analyze the probability that $\mathcal{B}_1$ succeeds in guessing b (the probability that $\hat{b} = b$). Let E be the event that in a winning test session $\mathcal{A}$ produces a round-2 or round-3 message that passes the sanity check performed by the owner. By assumption, $\Pr[E] = \epsilon$. Let E_1 be the event that E occurs, the rth session is the test session and $\mathcal{P}$ is the peer. Then $\Pr[E_1] = \epsilon/Qn$.

Conditioning on E_1 gives that the message sent by $\mathcal{A}$ is the correct decryption of the ciphertext sent by the owner (since E_1 implies that the owner's sanity check passed). In other words, if $\mathcal{P}$ is the responder then $\hat{\alpha} = \mathtt{Dec}(c^*)$ and if $\mathcal{P}$ is the initiator then $\hat{\beta} = \mathtt{Dec}(c^*)$. Therefore, if E_1 occurs then B_1 breaks the CCA-security of $\mathcal{E}$. This happens with probability ϵ/Qn which is polynomial in ϵ.

□

Claim A.5. *Let $\mathcal{A}$ be an adversary attacking* Enc-DH. *If there is a negligible probability that in a* winning test session $\mathcal{A}$ *produces a round-2 or round-3 message that passes the sanity check performed by the* owner, *and $\mathcal{A}$ breaks the privacy of* Enc-DH *with probability ϵ, then there exists an attacker $\mathcal{B}_2$ that breaks the DDH assumption with probability polynomial in ϵ.*

Proof. We construct $\mathcal{B}_2$, which on input $(\alpha^*, \beta^*, \gamma^*)$ determines whether or not it's a DDH tuple.

- $\mathcal{B}_2$ chooses $r \xleftarrow{\$} [1, Q]$ (it guesses that the rth session will be the test session).
- If the rth session is not the test session then $\mathcal{B}_2$ halts. Otherwise, it sends $\mathtt{Enc}^{\mathcal{P}_i}_{\mathsf{pk}_j}(\alpha^*)$ as the round-3 message, and $\mathtt{Enc}^{\mathcal{P}_i,\mathcal{P}_j}_{\mathsf{pk}_i}(\hat{\alpha}, \beta^*)$ as the round-2 message.
- $\mathcal{B}_2$ gives γ^* to $\mathcal{A}$ as the challenge $\mathsf{session\ key}$, and outputs according to $\mathcal{A}$'s output.

Let E be the event that in a $\mathsf{winning\ test\ session}$ $\mathcal{A}$ produces a round-2 or round-3 message that passes the sanity check performed by the owner. By assumption, $\Pr[E] \le negl(\lambda)$. Let E_1 be the event that the test session is the rth session and the execution is winning. Then $\Pr[E_1] = \epsilon/Q$. We have that $\Pr[E_1 \wedge \neg E] = \Pr[E_1] - \Pr[E_1 \wedge E] \ge \epsilon/Q - \Pr[E] \ge \epsilon/Q - negl(\lambda)$.

Conditioning on the event $E_2 = E_1 \wedge \neg E$, gives that $\mathsf{sid} = (\alpha^*, \beta^*)$ and so if $(\alpha^*, \beta^*, \gamma^*)$ is a DDH tuple, then the challenge key presented to $\mathcal{A}$ is the real $\mathsf{session\ key}$. Otherwise, the challenge key is a random element in $\mathbb{G}$. Therefore, if E_2 occurs then B_2 breaks DDH (since E_2 implies that the execution is winning), and this happens with probability at least $\epsilon/Q - negl(\lambda)$, which is polynomial in ϵ.

$\square$

To prove perfect forward security, notice that if an adversary $\mathcal{A}$ learns the decryption keys of one or both parties at some point *after* the session is complete, then the only information it can learn is the decryption of the ciphertexts interchanged during the session. In particular, the only "new" information that $\mathcal{A}$ learns is α, and so $\mathcal{A}$ only knows α and β. By the DDH assumption, we know that given only this information $\mathcal{A}$ cannot learn the $\mathsf{session\ key}$ γ. This concludes the proof of Theorem 6.2.

$\square$

B The Groth-Sahai (GS) Proof System

In this section, we review the NIZK proof system of Groth and Sahai [36] for proving that a system of equations is satisfiable. We give details for the type of equations used in this paper, i.e. pairing-product (one-sided in the DLIN case) and one-sided multi-exponentiation. For full details and more general form of these types refer to [36]. In fact, we use the system as a NIZK argument system, achieving only computational soundness. This can be done by running all the algorithms with a simulated CRS. Note that in the GS proof system, there are two types of CRS and those are computationally indistinguishable: one (called real) gives perfectly sounds proofs and another (called simulated) yields perfect witness indistinguishable proofs, which could in many cases be transformed into zero-knowledge proofs.

When working under the K-Linear assumption ($K = 1$ for the SXDH assumption and $K = 2$ for the DLIN assumption), the common reference strings for the proof system Π consists of $\vec{u}_0, \vec{u}_1, \ldots, \vec{u}_K, \vec{u}$. Regardless of whether the CRS is real or simulated, $\vec{u}_i = (u_0, 1, \ldots, 1, u_i, 1, \ldots, 1)$, $i = 1, \ldots, K$, where $u_0, \ldots, u_k$ are randomly chosen group elements in $\mathbb{G}_1^*$. Let's denote with $\mathfrak{U}$ the $span(\vec{u}_1, \ldots, \vec{u}_K)$; note that $(g, 1, \ldots, 1) \notin \mathfrak{U}$. For the real CRS, which yields perfectly sound proofs, $\vec{u}_0 \xleftarrow{\$} \mathfrak{U}$ and $\vec{u} \xleftarrow{\$} \mathbb{G}_q^{K+1} \backslash \mathfrak{U}$. When the CRS is simulated, $\vec{u}_0 \xleftarrow{\$} \mathbb{G}_1^{K+1} \backslash \mathfrak{U}$ and $\vec{u} \xleftarrow{\$} \mathfrak{U}$. In the case of asymmetric pairings, i.e. in the SXDH setting, another set of vectors $\vec{v}_0, \vec{v}_1, \ldots, \vec{v}_K, \vec{v} \in \mathbb{G}_2^{K+1}$ is defined analogously for randomly chosen $v_0, \ldots, v_n \in \mathbb{G}_2^*$. Although for symmetric pairings we use only one-sided equations and a second set of vectors is not needed, we set $\vec{v} = \vec{u}$ and $\vec{v}_i = \vec{u}_i$, $i = 0, \ldots, K$, and use the two sets of vectors interchangeably for consistent notation (in the two settings).

To commit to a witness member $x \in \mathbb{G}_2$, choose a random $\vec{s} = (s_0, s_1, \ldots, s_K) \xleftarrow{\$} \mathbb{Z}_q^{K+1}$ and compute $\vec{\delta}_x \leftarrow Com_\Pi(x; \vec{s}) = (x, 1, \ldots, 1) \prod_{j=0}^{K} \vec{v}_j^{\,s_j}$, where vector multiplication is defined component-wise. To commit to a witness $\chi \in \mathbb{Z}_q$, for equations in $\mathbb{G}_2$, select $\vec{t} = (t_1, \ldots, t_K) \xleftarrow{\$} \mathbb{Z}_q^K$, and compute $\vec{\gamma}_\chi \leftarrow Com_\Pi(\chi; \vec{t}) = \vec{u}^\chi \prod_{j=1}^{K} \vec{u}_j^{\,t_j}$.

The GS proof system gives a proof for a set of equations being satisfiable by committing to each witness component separately and computing corresponding proof elements for each of the equations. Next we described how those proof elements are computed for each type of equations and how the satisfiability of the equations is verified; some of the notation is borrowed from [13].

One-sided Multi-exponentiation Equations

For an equation of the following type:

$$g_0 = g_1^{\chi_1} g_2^{\chi_2} \dots g_n^{\chi_n}$$

where $g_0, \dots, g_n \in \mathbb{G}_2$ are constants (one could view an equation being described by those constants) and $\chi_1, \dots, \chi_n \in \mathbb{Z}_q$ are variables (the witness for which the equation is satisfiable), the proof elements are $p_1, \dots, p_K$:

$$p_j = \prod_{i=1}^{n} g_i^{t_{ij}},\ j = 1, \dots, K,$$

where $\vec{t_i}$ is the randomness used to commit to χ_i, i.e. $\vec{\gamma}_{\chi_i} = Com_\Pi(\chi_i; \vec{t_i})$.

When verifying a proof, for each equation $g_0 = g_1^{\chi_1} g_2^{\chi_2} \dots g_n^{\chi_n}$ the verifier checks that the proof elements corresponding to the equation and the commitments satisfy

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

where $E : \mathbb{G}_1^{K+1} \times \mathbb{G}_2 \to \mathbb{G}_T^{K+1}$, sending $((\alpha_0, \dots, \alpha_K), \beta)$ to $(e(\alpha_0, \beta), \dots, e(\alpha_K, \beta))$, is a bilinear map.

The proofs for multi-exponentiation equations are zero knowledge (ZK). The size of a proof for set of S such equations being satisfiable with a witness of size N is $(K + 1)N + KS$ group elements. Note again that $K = 1$ when working under the SXDH and $K = 2$ under DLIN.

(One-sided) Pairing Product Equations

For an equation

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

where $h_1, \dots, h_n \in \mathbb{G}_1$ and $T \in \mathbb{G}_T$ are constants and $x_1, \dots, x_n \in \mathbb{G}_2$ are variables, the proof elements $p_0, \dots, p_K$:

$$p_j = \prod_{i=1}^{n} h_i^{s_{ij}},\ j = 0, \dots, K,$$

where $\vec{s_i}$ is the randomness used to commit to x_i, i.e. $\vec{\delta}_{x_i} = Com_\Pi(x_i; \vec{s_i})$. When verifying a proof, for each equation $\prod_{i=1}^{n} e(h_i, x_i) = T$ the verifier checks that the proof elements corresponding to the equation and the commitments satisfy

$$\prod_{i=1}^{n} E(h_i, \vec{\delta_i}) = (T, 1, 1, \dots, 1) \prod_{j=0}^{K} E(p_j, \vec{v_j}),$$

where $E : \mathbb{G}_1 \times \mathbb{G}_2^{K+1} \to \mathbb{G}_T^{K+1}$, sending $(\alpha, (\beta_0, \dots, \beta_K))$ to $(e(\alpha, \beta_0), \dots, e(\alpha, \beta_K))$, is a bilinear map.

These proofs are only witness indistinguishable (WI), and for a set of S pairing product equations satisfiable with a witness of size N, the proof size is $(K + 1)(N + S)$.

When representation of T as a pairing product is know it could be transformed into ZK [36] but resulting in somewhat larger proofs. However, in our case $T^{-1} = e(h_0, x_0)$ where both h_0 and x_0 are constants. So, we could transform the above equation into an equation $\prod_{i=0}^{n} e(h_i, x_i) = 1$ and give a WI proof accordingly treating x_0 as a part of the witness. Then, we produce a second commitment of $\vec{\delta}'_{x_0} = Com_\Pi(x_0; \vec{s'_0})$, include its randomness $\vec{s'_0}$ and a NIZK proof that $\vec{\delta}_{x_0}$ and $\vec{\delta}'_{x_0}$ are commitment to the same message using a set of one-sided multi-exponentiation equations. This way, when the simulator has to produce a ZK proof for the equation, it samples any $(x'_1, \ldots, x'_n)$ along with the appropriate x'_0, and gives a simulated proof that $\vec{\delta}_{x_0} = Com_\Pi(x'_0; \vec{s_0})$ and $\vec{\delta}'_{x_0} = (x_0; \vec{s'_0})$ are commitments to the same message. That results in additional $2(K+1)^2$ group elements and $(K+1)$ $\mathbb{Z}_q$-elements per equation to achieve ZK. (The count is as follows: $(K+1)$ group elements for $\vec{\delta}_{x_0}$, $(K+1)$ $\mathbb{Z}_q$-elements for $\vec{s'_0}$, $(K+1)^2$ group elements for the commitments to each component of $\vec{s_0}$, and $K(K+1)$ group elements for the NIZK proof of $\vec{\delta}'_{x_0}$ and $\vec{\delta}'_{x_0}$ being commitments of the same value (using $(K+1)$ one-sided multi-exponentiation equations).

So, under DLIN we get ZK proofs of size $3N + 21S$ *elements in* $\mathbb{G}$ *and* $3S$ *elements in* $\mathbb{Z}_q$ for a set of S equations being satisfiable with a witness which has size N.

In the SXDH setting, the equation is no longer one-sided as $T = e(y, \tilde{g})$ and $y \in \mathbb{G}_1$ whereas $x_i \in \mathbb{G}_2$. However, we could still apply the idea of treating y as a part of the witness and computing a second commitment $\vec{\gamma}'_y = Com_\Pi(y; \vec{s'_y})$, and then showing that the commitments $\vec{\gamma}_y$ and $\vec{\gamma}'_y$ are commitments of the same message. According to [36], the WI GS proofs under SXDH are of size $2N + 8S$ for a set of S equations being satisfiable and the witness being of size N. Combining this with the extra group elements we need per equations to achieve ZK, we get proofs of size $2N + 16S$ *elements in either group and* $2S$ *elements in* $\mathbb{Z}_q$ when working under SXDH.

C Instantiations

C.1 Preliminaries

A NOTE ON NOTATION. We follow the notation of [13]: for $\vec{g} = (g_1, g_2, \ldots, g_n) \in \mathbb{G}^n$ and $\vec{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{Z}_q$ we define:

$$\langle \vec{g}, \vec{x} \rangle := g_1^{x_1} \ldots g_n^{x_n}$$

When we write $\prod_{i=1}^n \vec{g_i} \in \mathbb{G}^n$ for vectors $\vec{g_i} \in \mathbb{G}^n$, we mean the component-wise product of each of the n terms.

CCA-SECURE ENCRYPTION BASED ON K-LINEAR. In our instantiations of both leakage-resilient signatures and CCA-secure encryption, we will need to use a (standard) CCA-secure encryption scheme. Since our instantiations are based on the K-Linear assumption, we will use the Linear Cramer-Shoup encryption scheme from [60], modified to support labels as in [13]. We review it here. We use the paradigm of [7] to transform it into a multi-message randomness-reuse encryption scheme, which we further optimize by reusing the consistency ciphertext element. Let $\mathbb{G}$ be a group of prime order q, and let $H : \{0,1\} \to \mathbb{Z}_q$ be a collision resistant hash function. The label space is $\{0,1\}^*$.

- `KeyGen`(1^λ) :

  1. Choose $g_0, g_1, \ldots, g_K \xleftarrow{\$} \mathbb{G}$ and choose $\vec{x_1}, \ldots, \vec{x_N}, \vec{y}, \vec{z} \xleftarrow{\$} \mathbb{Z}_q^{K+1}$.
  2. Define vectors $\vec{g_1}, \ldots, \vec{g_K} \in \mathbb{G}^{K+1}$ as follows:

  $$\vec{g_1} = (g_0, g_1, 1, \ldots, 1, 1), \vec{g_2} = (g_0, 1, g_2, 1, \ldots, 1, 1), \ldots, \vec{g_K} = (g_0, 1, \ldots, 1, g_K)$$

  3. For $i = 1, \ldots, K$ and $j = 1, \ldots, N$: let $d_{ji} \leftarrow \langle \vec{g_i}, \vec{x_j} \rangle$, $e_i \leftarrow \langle \vec{g_i}, \vec{y} \rangle$, $f_i \leftarrow \langle \vec{g_i}, \vec{z} \rangle$
  4. Ouput $sk = (\vec{x_1}, \ldots, \vec{x_N}, \vec{y}, \vec{z})$ and $pk = (\{g_i\}_{i=0}^K, \{d_{ji}\}_{i=1,j=1}^{K,N}, \{e_i\}_{i=1}^K, \{f_i\}_{i=1}^K)$

- $\mathtt{Enc}^L_{pk}(\vec{m} = (m_1, \ldots, m_N))$: Pick $\vec{r} \xleftarrow{\$} \mathbb{Z}_q^K$. For $i = 1, \ldots, K$: define $\vec{g_i}$ as in $\mathtt{KeyGen}$. Output

$$\vec{c} = (\vec{g}, a_1, \ldots, a_N, b) = \left(\prod_{i=1}^{K} \vec{g_i}^{r_i}, m_1 \cdot \prod_{i=1}^{K} d_{1i}^{r_i}, \ldots, m_N \cdot \prod_{i=1}^{K} d_{Ni}^{r_i}, \prod_{i=1}^{K} (e_i f_i^t)^{r_i} \right),$$

  where $t = H(\vec{g}, a_1, \ldots, a_N, L)$

- $\mathtt{Dec}^L_{sk}(c = (\vec{g}, a_1, \ldots, a_N, b,))$: Let $t \leftarrow H(\vec{g}, a_1, \ldots, a_N, L)$. If $b \neq \langle \vec{g}, \vec{y} + t\vec{z} \rangle$, output $\perp$. Else, for $j = 1, \ldots, N$, let $m_j \leftarrow a_j / \langle \vec{g}, \vec{x_j} \rangle$.

Notice that for $K = 1$, the encryption scheme described above is the Cramer-Shoup (multi-message randomness-reuse) encryption scheme.

C.2 Leakage-Resilient Signatures

In order to efficiently instantiate the construction in Section 4.2, we need to give an SPR relation R, a CCA-secure encryption scheme, and an efficient NIZK argument for relation R_Π. We will use the CCA-secure scheme described in the preliminaries and the NIZK argument system from Appendix B. We now discuss our choice of SPR relation. Henceforth, we let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be groups of prime order q and $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a non-degenerate bilinear map that is efficiently computable. We let g be a random generator of $\mathbb{G}_1$ and let $\tilde{g}$ be a random generator of $\mathbb{G}_2$.

C.2.1 SPR Relations

Previous constructions of leakage-resilient primitives often use the function $g_1^{x_1} g_2^{x_2} \ldots g_n^{x_n}$, but this does not allow an efficient extraction of the witness $(x_n, \ldots, x_n)$ when using GS proofs (unless each witness in committed bit by bit which, among other things, results in proofs growing linearly with the security parameter). To overcome this problem, we use SPR functions based on bilinear maps. For our SXDH instantiation, we use the SPR relation $e(h_1, x_1)\ e(h_2, x_2) \ldots e(h_n, x_n) = e(y, \tilde{g})$, where $\tilde{g}$ is a generator of $\mathbb{G}_2$. In the DLIN case, we use the relation: $e(h_1, x_1)\ e(h_2, x_2) \ldots e(h_n, x_n) = e(y_1, g)\ \wedge\ e(\hbar_1, x_1)\ e(\hbar_2, x_2) \ldots e(\hbar_n, x_n) = e(y_2, g)$. Both cases allow for easy extraction of the witness $(x_1, \ldots, x_n)$ and a seamless combination with the encryption scheme. As a side note, we use an SPR relation instead of an SPR *function* in order to achieve zero-knowledge in the Groth-Sahai arguments. In general, GS proofs are witness indistinguishable for pairing product equations but can be made zero-knowledge if we can represent the equation product itself as a product of one or more pairings.

We show the details of our SPR constructions below, but first we review two assumptions that we will use in our SPR proofs.

Double Pairing [1, 33]. The double pairing assumption states that given two random elements $g_1, g_2 \in \mathbb{G}_1$, it is hard to find a non-trivial couple $(z_1, z_2) \in \mathbb{G}_2^2$ such that $e(g_1, z_1)e(g_2, z_2) = 1$. It is easy to check that the Double Pairing assumption is implied by SXDH (see [1, 33] for details).

Simultaneous Triple Pairing (STP) [1, 33]. The simultaneous triple pairing assumption states that given six random elements $g_1, g_2, g_3, g_1', g_2', g_3' \in \mathbb{G}_1$, it is hard to find a non-trivial triple $(z_1, z_2, z_3) \in \mathbb{G}_2^3$ such that $e(g_1, z_1)e(g_2, z_2)e(g_3, z_3) = 1$ and $e(g_1', z_1)e(g_2', z_2)e(g_3', z_3) = 1$. It was shown in [33] that the STP assumption is implied by the DLIN assumption.

Based on SXDH. Let $n \geq 2$ and $h_1, h_2, \ldots, h_n$ be random elements in $\mathbb{G}_1$, and let $\tilde{g}$ be a generator of $\mathbb{G}_2$. We construct the SPR relation:

- `KeyGen`(1^λ) : Output $\vec{x} = (x_1, \ldots, x_n)$ and y where:
  $r_1, \ldots, r_n \xleftarrow{\$} \mathbb{Z}_q$, $\{x_i \leftarrow \tilde{g}^{r_i}\}_{i=1}^n$, $y \leftarrow \prod_{i=1}^n h_i^{r_i}$.
- $R(x, y)$: Output 1 if $\prod_{i=1}^n e(h_i, x_i) = e(y, \tilde{g})$. Otherwise output 0.

Claim C.1. *Under the SXDH assumption, the relation R described above is SPR with average-case preimage entropy $\mathbf{H}_{avg}(R) = (n-1)\log(q)$.*

Proof. For any fixed choice of y, the conditional distribution of $\vec{x}$ is uniform over some $n-1$ dimensional subspace of $\mathbb{G}_2^n$, which gives us the worst-case preimage entropy of $(n-1)\log(q)$.

We prove that R is SPR under the double-pairing assumption. Since the SXDH assumption implies the double-pairing assumption, the claim holds.

Consider an adversary $\mathcal{A}$ that given $h_1, \ldots, h_n, \vec{x}, y$ such that $\prod_{i=1}^n e(h_i, x_i) = T$, where $T = e(y, \tilde{g})$, finds $\vec{x^*} \neq \vec{x}$ such that $\prod_{i=1}^n e(h_i, x_i^*) = T$, with probability $\varepsilon > negl(\lambda)$. We construct adversary $\mathcal{B}$ that breaks the double pairing assumption.

$\mathcal{B}$ takes as input g_1, g_2, chooses $\alpha_1, \beta_1, \ldots, \alpha_n, \beta_n \leftarrow \mathbb{Z}_q$, and sets $h_i = g_1^{\alpha_i} g_2^{\beta_i}$, for $i = 1, \ldots, n$. $\mathcal{B}$ then samples $(\vec{x}, y)$ and gives $h_1, \ldots, h_n, \vec{x}, y$ to $\mathcal{A}$. With probability ε, $\mathcal{A}$ returns $\vec{x^*} \neq \vec{x}$ for which $\prod_{i=1}^n e(h_i, x_i^*) = T$. Dividing the two pairing product equations:

$$\prod_{i=1}^n e(h_i, x_i/x_i^*) = e(g_1, \prod_{i=1}^n (x_i/x_i^*)^{\alpha_i}) e(g_2, \prod_{i=1}^n (x_i/x_i^*)^{\beta_i}) = e(g_1, z_1) e(g_2, z_2) = 1.$$

It remains to prove that $(z_1, z_2) = (\prod_{i=1}^n (x_i/x_i^*)^{\alpha_i}, \prod_{i=1}^n (x_i/x_i^*)^{\beta_i}) \neq (1, 1)$. There exists $j \in [1, \ldots, n]$ for which $x_j/x_j^* \neq 1$ and α_j is information theoretically hidden. Therefore, $z_1 \neq 1$ with probability $(1 - 1/q)$. $\mathcal{B}$ outputs (z_1, z_2) and with probability $\varepsilon(1 - 1/q) > negl(\lambda)$, $e(g_1, z_1)e(g_2, z_2) = 1$ and $(z_1, z_2) \neq (1, 1)$. Thus, $\mathcal{B}$ breaks the double pairing assumption with non-negligible probability. □

Based on DLIN. Let $n \geq 3$ and $h_1, \ldots, h_n, \hbar_1, \ldots, \hbar_n$ be $2n$ elements in $\mathbb{G}$ and let g be a generator of $\mathbb{G}$. We construct the SPR relation:

- `KeyGen`(1^λ) : Output $\vec{x} = (x_1, \ldots, x_n)$ and $\vec{y} = (y_1, y_2)$ where:
  $r_1, \ldots, r_n \xleftarrow{\$} \mathbb{Z}_q$, $\{x_i \leftarrow g^{r_i}\}_{i=1}^n$, $y_1 \leftarrow \prod_{i=1}^n h_i^{r_i}$, $y_2 \leftarrow \prod_{i=1}^n \hbar_i^{r_i}$.
- $R(x, y)$: Output 1 if $\prod_{i=1}^n e(h_i, x_i) = e(y_1, g)$ and $\prod_{i=1}^n e(\hbar_i, x_i) = e(y_2, g)$. Otherwise output 0.

Claim C.2. *Under the DLIN assumption, the relation R described above is SPR with worst-case preimage entropy $\mathbf{H}_{avg}(R) = (n-2)\log(q)$.*

Proof. For any fixed choice of y, the conditional distribution of $\vec{x}$ is uniform over some $n-2$ dimensional subspace of $\mathbb{G}^n$, which gives us the worst-case preimage entropy of $(n-2)\log(q)$.

We prove that R is SPR under the simultaneous triple pairing assumption (STP). Since the DLIN assumption implies the STP assumption, we have that R is SPR under the DLIN assumption.

The proof is analogous to that of Claim C.1. If $g_1, g_2, g_3, g_1', g_2', g_3'$ is the instance for which $\mathcal{B}$ tries to break the STP, it computes $h_i = g_1^{\alpha_i} g_2^{\beta_i} g_3^{\gamma_i}$ and $\hbar_i = (g_1')^{\alpha_i} (g_2')^{\beta_i} (g_3')^{\gamma_i}$, for $i = 1, \ldots, n$, where $\alpha_1, \beta_1, \gamma_1, \ldots, \alpha_n, \beta_n, \gamma_n \leftarrow \mathbb{Z}_q$. Then, $\mathcal{B}$ samples $\vec{x}, \vec{y}$ and runs $\mathcal{A}$ with the appropriate input. With probability $\epsilon > negl(\lambda)$, $\mathcal{A}$ returns $\vec{x^*} \neq \vec{x}$ such that $\prod_{i=1}^n e(h_i, x_i) = e(y_1, g)$ and $\prod_{i=1}^n e(\hbar_i, x_i) = e(y_2, g)$. But notice that

$$e(g_1, \prod_i (x_i/x_i^*)^{\alpha_i})\ e(g_2, \prod_i (x_i/x_i^*)^{\beta_i})\ e(g_3, \prod_i (x_i/x_i^*)^{\gamma_i}) = 1, \quad \text{and}$$

$$e(g_1', \prod_i (x_i/x_i^*)^{\alpha_i})\ e(g_2', \prod_i (x_i/x_i^*)^{\beta_i})\ e(g_3', \prod_i (x_i/x_i^*)^{\gamma_i}) = 1.$$

$\mathcal{B}$ outputs $(z_1, z_2, z_3) = (\prod_i (x_i/x_i^*)^{\alpha_i}, \prod_i (x_i/x_i^*)^{\beta_i}, \prod_i (x_i/x_i^*)^{\gamma_i})$. Since there exists $j \in [1, \ldots, n]$ such that $x_j \neq x_j^*$ (because $\vec{x^*} \neq \vec{x}$) and α_j is information theoretically hidden (which is easily observed and is fully explained in [33]), then with non-negligible probability $(z_1, z_2, z_3) \neq (1, 1, 1)$, $e(g_1, z_1)e(g_2, z_2)e(g_3, z_3) = 1$ and $e(g'_1, z_1)e(g'_2, z_2)e(g'_3, z_3) = 1$. Hence, with non-negligible probability $\mathcal{B}$ breaks the STP assumption. $\square$

We now show instantiations of the construction described in Section 4.2, meeting the parameters of Theorem 5.1.

C.2.2 Instantiation 1: Based on SXDH

Our first instantiation is based on SXDH when working with asymmetric bilinear groups.

SPR Relation. We use the SPR relation described in Section C.2.1.

- $\mathtt{KeyGen}(1^\lambda)$: Output $\vec{x} = (x_1, \ldots, x_n)$ and y where:
  $r_1, \ldots, r_n \xleftarrow{\$} \mathbb{Z}_q$, $\{x_i \leftarrow \tilde{g}^{r_i}\}_{i=1}^n$, $y \leftarrow \prod_{i=1}^n h_i^{r_i}$.
- $R(x, y)$: Output 1 if $\prod_{i=1}^n e(h_i, x_i) = e(y, \tilde{g})$. Otherwise output 0.

Recall that this relation has average-case preimage entropy of $(n-1)\log(q)$.

CCA-Secure Encryption. We use the Cramer-Shoup encryption scheme described in Section C.1, working in the group $\mathbb{G}_2$. We encrypt $\vec{x} = (x_1, \ldots, x_n)$ under the same randomness r. More formally, for a public key $pk = (g_0, g_1, d_1, \ldots, d_n, e, f)$, we encrypt $\vec{x}$ with label m as

$$C = (c_1, c_2, c_3, \ldots, c_{n+2}, c_{n+3}) \leftarrow \mathtt{Enc}_{pk}^m(x_1, x_2; r) = (g_0^r, g_1^r, x_1 d_1^r, \ldots, x_n d_n^r, (ef^t)^r),$$

where $t = H(c_1, \ldots, c_{n+2}, m)$.

The total size of the ciphertext is $n + 3$.

NIZK Argument. We use the NIZK proofs described in Appendix B to prove that "$R_{spr}(x, y) = 1$ and $C = \mathtt{Enc}_{pk}^m(x; r)$". First we show that $R(\vec{x}, y) = 1$ by creating a commitment $\delta_i = Com_\Pi(x_i; (s_{i0}, s_{i1}))$ for each component x_i of $\vec{x} = (x_1, \ldots, x_n)$ and producing proof elements which show that the committed values satisfy the pairing product equation $\prod_{i=1}^n e(g_i, x_i) = e(y, \tilde{g})$. Then, we show that $C = \mathtt{Enc}_{pk}^m(\vec{x}; r)$ using a system of one-sided multi-exponentiation equations with witness $(r, s_{10}, s_{11}, \ldots, s_{n0}, s_{n1})$ to show that the plaintext encrypted in C is equal to the committed values in the δ_i's. Details follow. Let

$$\vec{\delta}_1 = (x_1, 1)\vec{v}_0^{\,s_{10}}\vec{v}_1^{\,s_{11}}, \quad \ldots, \quad \vec{\delta}_n = (x_n, 1)\vec{v}_0^{\,s_{n0}}\vec{v}_1^{\,s_{n1}},$$

and, as defined above, $C = (c_1, \ldots, c_{n+3})$. Proving equality of the plaintext and the committed values reduces to proving the satisfiability of the following system of $2n + 3$ equations:

$$\begin{array}{c} \frac{\vec{\delta}_1}{(c_3,1)} = \vec{v}_0^{\,s_{10}}\vec{v}_1^{\,s_{11}}(d_1^{-1}, 1)^r \ , \quad \ldots \quad , \frac{\vec{\delta}_n}{(c_{n+2},1)} = \vec{v}_0^{\,s_{n0}}\vec{v}_1^{\,s_{n1}}(d_n^{-1}, 1)^r \ , \\ c_1 = g_0^r \ , \quad c_2 = g_1^r \ , \quad c_{n+3} = e^r(f^t)^r \ . \end{array}$$

The total size of the argument is $8n + 21$ group elements and 2 $\mathbb{Z}_q$-elements.

Combining the ciphertext and the NIZK argument makes the size of the signature $9n + 24$ group elements and 2 elements in $\mathbb{Z}_q$. By Theorem 4.2 and Theorem 4.3, we know that the above instantiation gives us a $((n-1)\log q - 1)$-leakage resilient signature scheme. To translate this into $(1-\epsilon)|sk|$ leakage tolerance, we need

$$n \geq \frac{1}{\epsilon} + \frac{\omega(\log\lambda)}{\epsilon \cdot \log q} = \frac{1}{\epsilon} \cdot \left(1 + \frac{\omega(\log\lambda)}{\log q}\right)$$

This gives us signatures of size $(9/\epsilon)(1 + \omega(\log\lambda)/\log q) + 24$ group elements and 2 elements in $\mathbb{Z}_q$.

C.2.3 Instantiation 2: Based on DLIN

In the case of $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$, we give an instantiation under the DLIN assumption.

SPR Relation. We use the SPR relation described in Section C.2.1.

- $\texttt{KeyGen}(1^\lambda)$: Output $\vec{x} = (x_1, \dots, x_n)$ and $\vec{y} = (y_1, y_2)$ where:
  $r_1, \dots, r_n \xleftarrow{\$} \mathbb{Z}_q$, $\{x_i \leftarrow g^{r_i}\}_{i=1}^n$, $y_1 \leftarrow \prod_{i=1}^n h_i^{r_i}$, $y_2 \leftarrow \prod_{i=1}^n \hbar_i^{r_i}$.
- $R(x, y)$: Output 1 if $\prod_{i=1}^n e(h_i, x_i) = e(y_1, g)$ and $\prod_{i=1}^n e(\hbar_i, x_i) = e(y_2, g)$. Otherwise output 0.

Recall that this relation has average-case preimage entropy of $(n-2)\log(q)$.

CCA-Secure Encryption. We use the Cramer-Shoup encryption scheme described in Section C.1. We encrypt $\vec{x} = (x_1, \dots, x_n)$ under the same randomness r and label m. More formally, for a public key

$$pk = (g_0, g_1, g_2, d_{11}, d_{12}, \dots, d_{n1}, d_{n2}, e_1, e_2, f_1, f_2),$$

we compute the ciphertext

$$\begin{aligned} C = (c_1, \dots, c_{n+4}) &\leftarrow \texttt{Enc}_{pk}^m(\vec{x}; (r_1, r_2)) \\ &= (g_0^{r_1+r_2},\ g_1^{r_1},\ g_2^{r_2},\ x_1 d_{11}^{r_1} d_{12}^{r_2},\ \dots,\ x_n d_{n1}^{r_1} d_{n2}^{r_2},\ (e_1 f_1^t)^{r_1}(e_2 f_2^t)^{r_2}), \end{aligned}$$

where $t = H(c_1, \dots, c_{n+3}, m)$.

The size of the ciphertext is $n + 4$.

NIZK Argument. First we prove that $R(x, y) = 1$ using the pairing product equations

$$\begin{aligned} e(h_1, x_1) \dots e(h_n, x_n) &= e(g, y_1) \quad \text{and} \\ e(\hbar_1, x_1) \dots e(\hbar_n, x_n)) &= e(g, y_2). \end{aligned}$$

We create commitments $\delta_i = Com_\Pi(x_i; \vec{s_i}) = (x_i, 1, 1)v_0^{s_{i0}} v_1^{s_{i1}} v_2^{s_{i2}}$, for each component x_i of $\vec{x} = (x_1, \dots, x_n)$ using randomness $\vec{s_i} = (s_{i0}, s_{i1}, s_{i2})$. Then we prove that the plaintext of $C = \texttt{Enc}_{pk}^m(x_1, x_2, x_3; \vec{r})$ is the committed values in the δ_i's by proving that the following system of $3n+4$ one-sided multi-exponentiation equations is satisfiable with a witness $(r_1, r_2, \vec{s_1}, \dots, \vec{s_n})$:

$$\begin{aligned} \frac{\vec{\delta}_1}{(c_4,1,1)} &= \vec{v}_0^{\,s_{10}} \vec{v}_1^{\,s_{11}} \vec{v}_2^{\,s_{12}} (d_{11}^{-1}, 1, 1)^{r_1} (d_{12}^{-1}, 1, 1)^{r_2} \ , \\ &\dots \\ \frac{\vec{\delta}_n}{(c_{n+3},1,1)} &= \vec{v}_0^{\,s_{n0}} \vec{v}_1^{\,s_{n1}} \vec{v}_2^{\,s_{n2}} (d_{n1}^{-1}, 1, 1)^{r_1} (d_{n2}^{-1}, 1, 1)^{r_2} \ , \end{aligned}$$

$$c_1 = g_0^{r_1} g_0^{r_2} \ , \quad c_2 = g_1^{r_1} \ , \quad c_3 = g_2^{r_2} \ , \quad c_{n+4} = (e_1 f_1^t)^{r_1}(e_2 f_2^t)^{r_2} \ .$$

The total size of the proof is 18n+66 group elements and 6 $\mathbb{Z}_q$-elements.

Combining the ciphertext and the NIZK argument makes the size of the signature $19n + 70$ group elements and 6 elements in $\mathbb{Z}_q$. By Theorem 4.2 and Theorem 4.3, we know that the above instantiation gives us a $((n-2)\log q - 1)$-leakage resilient signature scheme. To translate this into $(1-\epsilon)|sk|$ leakage tolerance, we need

$$n \geq \frac{2}{\epsilon} + \frac{\omega(\log \lambda)}{\epsilon \cdot \log q} = \frac{1}{\epsilon} \cdot \left(2 + \frac{\omega(\log \lambda)}{\log q}\right)$$

This gives us signature of size $(19/\epsilon)(2 + \omega(\log \lambda)/\log q) + 70$ group elements and 6 elements in $\mathbb{Z}_q$.

C.3 Leakage-Resilient Encryption

In order to use the construction in Section 4.3, we need a $(1-\epsilon)|sk|$-leakage resilient CPA-secure encryption scheme $\mathcal{E}_1 = (\texttt{KeyGen}_1, \texttt{Enc}_1, \texttt{Dec}_1)$ and a strong f-tSE NIZK argument (see Section 3), which we can construct from a CCA-secure encryption scheme supporting labels $\mathcal{E}_2 = (\texttt{KeyGen}_2, \texttt{Enc}_2, \texttt{Dec}_2)$, a strongly-secure one-time time signature scheme $\mathcal{S}$, and an NIZK argument Π for the relation

$$R_{eq} = \{ \ (\ (r_1, r_2, m) \ , \ (c_1, c_2, L) \mid c_1 = \texttt{Enc}_1(m; r_1) \wedge c_2 = \texttt{Enc}_2^L(m; r_2) \ \}.$$

The same technique was used in [13] to construct an efficient CCA-secure encryption scheme with key-dependent message (KDM) security from a CPA-secure version of the scheme. We use the same technique in the leakage-setting, to achieve leakage-resilient CCA-secure encryption from leakage-resilient CPA-secure encryption.

We now show an instantiation of the construction shown in Section 4.3, meeting the parameters of Theorem 5.2.

LR-CPA-Secure Encryption ($\mathcal{E}_1$). We show a $(1-\epsilon)|sk|$-leakage resilient CPA-secure encryption scheme based on the K-Linear assumption. Similar versions of this scheme appear in [49] and [13] (based on the KDM scheme of [12]), but we modify it here to make it more efficient. In particular, our public key and ciphertexts are shorter by a factor of $\log q$.

Let $\mathbb{G}$ be a group of primer order q, and let J be an integer. We define the scheme $\mathcal{E}_1$ by:

- $\texttt{KeyGen}(1^\lambda)$: Choose $f_{01}, \ldots, f_{0J}, f_1, \ldots f_K \xleftarrow{\$} \mathbb{G}$ and $\vec{x} \xleftarrow{\$} \mathbb{Z}_q^{K+J}$. Define vectors $\vec{f_1}, \ldots, \vec{f_K} \in \mathbb{G}^{K+J}$ as follows:

$$\begin{aligned}
\vec{f_1} &= (f_{01}, \ldots, f_{0J}, f_1, 1, \ldots, 1) \\
\vec{f_2} &= (f_{01}, \ldots, f_{0J}, 1, f_2, \ldots, 1) \\
&\vdots \\
\vec{f_K} &= (f_{01}, \ldots, f_{0J}, 1, 1, \ldots, f_K)
\end{aligned}$$

  For $i = 1, \ldots, K$: let $h_i = \langle \vec{f_i}, \vec{x} \rangle$. Let $\vec{h} = (h_1, \ldots, h_K)$. Output $sk = \vec{x}$ and $pk = (\{\vec{f_i}\}_{i=1}^K, \vec{h})$.

- $\texttt{Enc}_{pk}(m)$: Choose $\vec{w} \xleftarrow{\$} \mathbb{Z}_q^K$. Let $\vec{f} = \prod_{i=1}^K \vec{f_i}^{w_i}$ and $a = m \cdot \langle \vec{h}, \vec{w} \rangle$. Output $C = (\vec{f}, a)$.

- $\texttt{Dec}_{sk}(C)$: Output $m \leftarrow a / \langle \vec{f}, \vec{x} \rangle$.

Theorem C.3. *For any $\epsilon > 0$, if $J \geq \frac{1}{\epsilon}(K + \lambda/\log(q) + 1)$, then the above encryption scheme is ℓ-leakage resilient where $\ell = (1-\epsilon)|sk|$. The scheme is secure under the K-linear assumption.*

The proof follows from the same technique as those used to prove leakage-resilience of hash-proof system based schemes in [49]. Indeed, it is relatively simple to see that the above construction is based on an underlying hash-proof system. However, for simplicity, we just prove the leakage resilience of the scheme directly without defining the notion of a hash-proof system formally in this work.

Proof. We do a series of games argument to show that the scheme is ℓ-LR-CPA.

Game 0: This is the ℓ-LR-CPA attack game. The adversary gets ℓ bits of leakage on $sk = \vec{x}$ and the challenge ciphertext is (later) computed as:

$$C = (\vec{f}, a) \text{ where } \vec{w} \xleftarrow{\$} \mathbb{Z}_q^K, \vec{f} = \prod_{i=1}^{K} \vec{f_i}^{w_i}, a = m_b \cdot \langle \vec{h}, \vec{w} \rangle.$$

as an encryption of message m_b where the bit b is chosen by the challenger.

Game 1: In this game the challenge ciphertext is computed using the *secret key* $\vec{x}$ as:

$$C = (\vec{f}, a) \text{ where } \vec{w} \xleftarrow{\$} \mathbb{Z}_q^K, \vec{f} = \prod_{i=1}^{K} \vec{f_i}^{w_i}, a = m_b \cdot \langle \vec{f}, \vec{x} \rangle.$$

Games 0 and games 1 are equivalently distributed since $\langle \vec{f}, \vec{x} \rangle = \langle \vec{h}, \vec{w} \rangle$ (this is essentially the correctness of decryption).

Game 2: In this game the $\vec{f}$ part of the ciphertext is just chosen uniformly at random:

$$C = (\vec{f}, a) \text{ where } \vec{f} \leftarrow \mathbb{G}^{J+K}, a = m_b \cdot \langle \vec{f}, \vec{x} \rangle.$$

The fact that games 1 and 2 are computationally indistinguishable follows from the K-linear assumption, which ensures that a random linear combination of $\vec{f_1}, \dots, \vec{f_K}$ (used to compute $\vec{f}$ in Game 2) is computationally indistinguishable from a uniformly random $\vec{f}$. This holds *even* given all of the secret-key $\vec{x}$, and hence certainly in the presence of limited leakage.

Game 3: In this game, a is chosen uniformly at random so that

$$C = (\vec{f}, a) \text{ where } \vec{f} \leftarrow \mathbb{G}^{J+K}, a \leftarrow \mathbb{G}.$$

We claim that games 2 and 3 are statistically indistinguishable. This is because, in game 2, $\langle \vec{f}, \vec{x} \rangle$ can be thought of as a universal hash function of the secret-key $\vec{x}$ under the hash-key $\vec{f}$. Since, a universal hash function is a good average-case randomness extractor (see [22]), the value $\langle \vec{f}, \vec{x} \rangle$ is statistically indistinguishable from uniform, as long as the conditional entropy of $\vec{x}$ given everything else the adversary sees in game 2 is at least $\log(q) + \lambda$ bits. But the only information that the adversary sees in game 2 which is correlated with $\vec{x}$ is the component $\vec{h}$ of the public-key and the leakage. Therefore, $\vec{x}$ has at least $(K + J)\log(q) - K\log(q) - \ell$ bits of conditional entropy. Since $\ell = (1 - \epsilon)|sk| = (1 - \epsilon)(K + J)\log(q) \leq (J - 1)\log(q) - \lambda$, this means $\vec{x}$ has at least $\log(q) + \lambda$ bits of conditional entropy, as desired.

It is clear that Game 3 is independent of the challenger's bit b and hence the adversary's advantage is 0 (the probability that $b' = b$ is $\frac{1}{2}$). Therefore, by the hybrid argument, the adversary's advantage in Game 0 is negligible. □

For the instantiation, we use the LR-CPA-secure encryption scheme described above, working in the group $\mathbb{G}_2$. We encrypt m under randomness $\vec{w} = (w_1, \dots w_K)$: for a public key $pk = (\vec{f_1}, \dots, \vec{f_K}, \vec{h})$ with $\vec{f_i} = (f_{01}, \dots, f_{0J}, 1, \dots, f_i, \dots, 1)$, we compute $W = \sum_{i=1}^{K} w_i$ and ciphertext

$$C_1 = (c_{11}, \dots, c_{1(J+K+1)}) \leftarrow \texttt{Enc}_1(m; \vec{w}) = (f_{01}^{W}, \dots, f_{0J}^{W}, f_1^{w_1} \dots, f_K^{w_K}, m \prod_{i=1}^{K} h_i^{w_i})$$

The size of the ciphertext is $J + K + 1$.

CCA-secure Encryption ($\mathcal{E}_2$). We use the Linear Cramer-Shoup encryption scheme described in section C.1, working in the group $\mathbb{G}_2$. We encrypt m under randomness $\vec{r} = (r_1, \dots, r_K)$ and label L: for a public key $pk = (\{g_i\}_{i=0}^{K}, \{d_i\}_{i=1}^{K}, \{e_i\}_{i=1}^{K}, \{f_i\}_{i=1}^{K})$, we compute $R = \sum_{i=1}^{K} r_i$ and the ciphertext

$$C_2 = (c_{21}, \dots, c_{2(K+3)}) \leftarrow \texttt{Enc}_2^L(m; \vec{r}) = (g_0^R, g_1^{r_1} \dots, g_K^{r_K}, m \prod_{i=1}^{K} d_i^{r_i}, \prod_{i=1}^{K} (e_i f_i^t)^{r_i}),$$

where $t = H_1(c_{21}, \dots, c_{2(K+2)}, L)$ and H_1 is a collision-resistant hash function.

The size of the ciphertext is $K + 3$.

NIZK Argument System. We use the NIZK proofs described in Appendix B. Let C_1, C_2 be as described above. To prove that there exists (m, r_1, r_2) such that $((m, r_1, r_2), (C_1, C_2, L)) \in R_\Pi$, we use a system of multi-exponentiation equations.

$$\begin{array}{rcll} c_{1j} & = & f_{0j}^{\sum_{i=1}^{K} w_i} & \text{for } j = 1, \ldots, J \\ c_{1(J+i)} & = & f_i^{w_i} & \text{for } i = 1, \ldots, K \\ c_{21} & = & g_0^{\sum_{i=1}^{K} r_i} & \\ c_{2(i+1)} & = & g_i^{r_i} & \text{for } i = 1, \ldots, K \\ c_{2(K+3)} & = & \prod_{i=1}^{K} (e_i f_i^t)^{r_i} & \\ c_{1(J+K+1)}/c_{2(K+2)} & = & \prod_{i=1}^{K} h_i^{w_i} (d_i^{-1})^{r_i} & \end{array}$$

This corresponds to a system of $J + 2K + 3$ equations with witness $(r_1, \ldots, r_K, w_1, \ldots, w_K)$. Using the proofs described in appendix B we can give a proof for the simultaneous satisfiability of the equations using $2K$ commitments and $K \cdot (J + 2K + 3)$ proof elements.

Based on SXDH: In this case we have $K = 1$, so the size of the proof is $J + 9$ group elements.

Based on DLIN: In this case we have $K = 2$, so the size of the proof is $2J + 26$ group elements.

One-Time Signature ($\mathcal{S}$). We use the strongly-secure signature of [34]. Let $H_2 : \{0,1\}^* \to \mathbb{Z}_q$ be a collision-resistant hash function.

- $\texttt{KeyGen}_{\mathcal{S}}(1^\lambda)$: Output $vk = (\mathfrak{g}, \mathfrak{f}, \mathfrak{h}, \mathfrak{z})$ and $sk = (a_1, a_2)$, where:

$$a_1, a_2, b_1, b_2 \xleftarrow{\$} \mathbb{Z}_q^* \ , \ \mathfrak{g} \xleftarrow{\$} \mathbb{G}_2 \ , \ \mathfrak{f} \leftarrow \mathfrak{g}^{a_1}, \mathfrak{h} \leftarrow \mathfrak{g}^{a_2} \ , \ \mathfrak{z} \leftarrow \mathfrak{f}^{b_1} \mathfrak{h}^{b_2}$$

- $\texttt{Sign}_{\mathcal{S}}(m; r)$: Output $\sigma = (r, s)$, where

$$s \leftarrow \left(\frac{(a_1(b_1 - r) + a_2 b_2 - H_2(m))}{a_2} \right)$$

- $\texttt{SigVer}_{\mathcal{S}}(m, \sigma = (r, s))$: Check that $\mathfrak{z} = \mathfrak{g}^{H_2(m)} \mathfrak{f}^r \mathfrak{h}^s$

The size of the one-time signature if 2 elements in $\mathbb{Z}_q$.

Combining both ciphertexts, together with the NIZK argument and the one-time signature, we have that the size of the ciphertext is $2J + 15$ group elements and 2 elements in $\mathbb{Z}_q$ in the SXDH case, and $3J + 34$ group elements and 2 elements in $\mathbb{Z}_q$ in the DLIN case. From Theorem C.3 we need $J \geq \frac{1}{\epsilon}(K + \lambda/\log(q) + 1)$. This gives us the following ciphertext size:

Based on SXDH: The size of the ciphertext is $(2/\epsilon)(2 + \lambda/\log q) + 15$ group elements and 2 elements in $\mathbb{Z}_q$.

Based on DLIN: The size of the ciphertext is $(3/\epsilon)(3 + \lambda/\log q) + 34$ group elements and 2 elements in $\mathbb{Z}_q$.