Continuous Collision Resistance and its Applications

Tibor Jager and Rafael Kurek

Paderborn University
{tibor.jager, rafael.kurek}@upb.de

Abstract. We introduce a new, simple and non-interactive complexity assumption for cryptographic hash functions, which seems very reasonable for standard functions like SHA-3. We describe how this assumption can be leveraged to obtain standard-model constructions that previously seemed to require a programmable random oracle: a generic construction of identity-based key encapsulation (ID-KEM) with full adaptive security from a scheme with very weak security ("selective and non-adaptive chosen-ID security"), a similar generic construction for digital signatures, and the first constructions of ID-KEMs and signatures over bilinear groups, where a ciphertext or signature consists of only a *single* group element and which achieve full adaptive security without random oracles.

Continuous collision resistance can be viewed as a way to realize certain potential applications of *extremely lossy functions* (ELFs; Zhandry, CRYPTO 2016) with a standard cryptographic primitive. Furthermore, known ELF constructions had only "nearly black-box" security proofs, because the reduction was assumed to "know" sufficiently close approximations of the running time and success probability of a given adversary. In contrast, our constructions allow for full black-box security proofs without this requirement. The main drawback of our schemes, from a practical perspective, is that the reductions in the security proof are very non-tight, and some are based on strong "q-type" assumptions. Therefore our results are mainly of conceptual interest, but not yet suitable for practical deployment.

1 Introduction

The random oracle model (ROM) [4] is often used to analyze the security of cryptosystems in a hypothetical setting, where a cryptographic hash function is modeled as an oracle that implements a truly random function. This provides a very strong handle for formal security proofs. For example, an adversary in this model has to explicitly query the oracle to evaluate the hash function, and it is possible to adaptively "program" the hash function to map certain input values to specific output values in the security proof.

It is well-known that random oracles do not exist [20]. Therefore the hypothetical random oracle can be used only in the security proof, and is instantiated

in practice with a standard cryptographic hash function, like SHA-3. This incurs the additional assumption that this hash function is "secure enough" for the given application. The major drawback of this approach is that the random oracle essentially provides a "perfect" hash function, which provides not only the standard security properties for cryptographic hash functions, like onewayness and collision resistance, but essentially all imaginable security properties of a cryptographic hash function simultaneously. Therefore a security proof in the random oracle model does not explain which precise security properties of a hash function are actually necessary and sufficient for a given application. This is very undesirable, as we want to understand the required security properties and we want to provide cryptanalysts with clearly-defined cryptanalytic goals to attack the "security" of cryptographic hash functions. Therefore the ROM is often seen as only a first step towards the goal of provably-secure construction in the standard model.

The only known security proofs for many important cryptographic constructions seem to inherently require an adaptively-programmable random oracle [26, 28, 25]. There are many primitives for which it is still unknown if and how they can be instantiated without random oracles, and where classical complexity assumptions on cryptographic hash functions seem not sufficient for a standard-model security proof. Several previous works isolated specific properties of random oracles, such as programmability [31] or extreme lossyness [40], and realized these properties with standard-model constructions of special-purpose functions and based on algebraic public-key techniques, which are relatively inefficient in comparison to standard cryptographic hash functions. In this work we ask the following question, which is orthogonal to these previous works:

Which reasonable (=simple and non-interactive) complexity assumptions on cryptographic hash functions are sufficient to obtain instantiations of cryptographic tools that currently require the ROM?

Contributions. We introduce a new complexity assumption called *continuous collision resistance*, which basically demands that there is no algorithm that finds collisions significantly faster than the standard birthday collision algorithm, even when (short) *prefixes* of hash values are considered. More precisely, let $H : \{0,1\}^* \to \{0,1\}^k$ be a random element from a family $\mathcal{H}$ of cryptographic hash function, and write $H_j(x)$ to denote the first j bits of $H(x)$. Continuous collision resistance requires that two input values x, x' with $x \neq x'$ and $H_j(x) = H_j(x')$ can not be found for *any* value of j with significantly better time-to-success ratio than the standard birthday collision algorithm.

We show that this new assumption, which in contrast to the ROM provides an explicit and well-defined goal for the cryptanalysis of hash functions, yields several interesting new cryptographic constructions without random oracles. This includes adaptively-secure identity-based key encapsulation schemes (ID-KEMs) with very short ciphertexts and adaptively-secure digital signatures over bilinear groups, where a signature consists of only a single group element. We furthermore describe generic constructions of adaptively-secure IBE and digital signatures

from IBE and signatures with extremely weak "selective" and "non-adaptive" security properties.

Relation to ELFs. From a high-level perspective, continuously collision-resistant (CCR) hash functions are related to *extremely lossy functions* (ELFs), which were recently introduced by Zhandry [40]. In some applications, CCR and ELFs can be used in a very similar way to argue in a security proof, and it seems that some of our applications can also be achieved by using an ELF instead. ELFs furthermore allow for some additional applications, like the construction of *output-intractable* hash functions or a standard-model instantiation of full-domain hash (the latter in combination with indistinguishability obfuscation).

The first main difference between Zhandry's work and ours is that [40] gives new constructions of hash functions based on public-key techniques and the reasonable exponential hardness assumption of the decisional Diffie-Hellman problem in algebraic groups. Instead, we use standard cryptographic hash functions (like e.g. SHA-3) that are already widely-used in practice, and propose a new, but similarly reasonable complexity assumption for such functions. This partially resolves the open problem posed in [40] of constructing ELFs using symmetric-key techniques: while we do not construct full ELFs, we show how certain potential applications of ELFs can be realized based on standard hash functions. We furthermore show how CCR can be used to obtain interesting new constructions: the first ID-KEM with full adaptive security and very short ciphertexts (only a single element from a bilinear group), and the first digital signature scheme with full adaptive security and very short signatures (again, only single element from a bilinear group).

A second major difference is that the security proofs in [40] require to know the running time $t_{\mathcal{A}}$ and success probability $\epsilon_{\mathcal{A}}$ of an adversary explicitly, and thus are only "nearly" black-box. We also start with the same setting, but merely for simplicity, as it isolates the core proof technique that exploits continuous collision resistance. Then we show how our results can be easily lifted to the fully black-box setting, which does not require any prior knowledge about the adversary. We also discuss why a similar lifting seems not possible for the known ELF instantiations from [40].

On assuming exponential hardness. Both the work of Zhandry [40] and our work assume exponential hardness of the underlying computational problems. The construction of ELFs from [40] assumes the exponential hardness of the DDH assumption in suitable algebraic groups. This is a strong assumption, but it appears reasonable e.g. in certain elliptic curve groups, where the best known algorithms for solving the DDH problem have exponential complexity. Furthermore, note that this matches the choice of elliptic curve groups in practice, where typically a group of prime order $\approx 2^{2k}$ is expected to achieve "k-bit security".

Similarly, we assume that for a cryptographic hash function there exists no significantly better collision attack than the generic birthday collision algorithm, which also has exponential complexity. Note also that the standard way to choose

the output size of a hash function in practice is to take twice the desired "security in bits", which means that one essentially assumes already that there is no significantly better full collision attack than the generic birthday algorithm, and that this seems achievable by "good" hash functions like SHA-3. Our new complexity assumption generalizes this assumption to prefixes of the hash value. This appears very reasonable for standard hash functions like SHA-3, where finding a significantly more efficient collision attack, even only for prefixes, would already be a major cryptanalytic achievement.

We point out that achieving adaptive security from selective security is sometimes also possible by directly assuming the exponential hardness of breaking the underlying selectively-secure scheme, and then using complexity leveraging. The main advantage of the modular approach of [40] and this work is that it outsources the exponential hardness assumption to a single and generic cryptographic primitive, where this assumption may be much more plausible, and which can also be applied to constructions that are based on hardness assumption for which the exponential hardness assumption does not hold.

Applications to identity-based key encapsulation. Recall that the commonly accepted standard security notion for identity-based key encapsulation (ID-KEM) is *adaptive chosen-ID* security (IND-ID-CPA), where the adversary in the security experiment may adaptively choose *both* the "challenge identity" (i.e., the identity for which it receives a challenge ciphertext) and the "key-query identities" (i.e., the identities for which it requests a user secret key). A much weaker common standard security notion is *selective challenge-ID security* (IND-sID-CPA), where the adversary has to announce the challenge identity and at the very beginning of the security experiment, even before seeing the master public key, but may adaptively query the for user secret keys.

In this work, we consider the even weaker notion with *selective* challenge-identity and *non-adaptive* key-queries (IND-snaID-CPA), where the adversary has to announce *both* the challenge identity and the key-query identities at the very beginning of the security experiment, even before seeing the master public key (see Section 3.1 for formal definitions). Existing standard techniques to build an adaptively-secure ID-KEM from a selectively-secure one, e.g. by using admissible [8] or programmable [31, 39] hash functions, work only non-generically for certain schemes with specific properties. The standard *generic* way to turn a IND-snaID-CPA-secure scheme into a fully IND-ID-CPA-secure one is to use the programmable ROM.

As a first application of continuous collision resistance, we describe a simple generic construction of fully adaptively IND-ID-CPA-secure ID-KEM from *any* ID-KEM which is only IND-snaID-CPA-secure. This shows that if continuously collision-resistant hash functions exist, then ID-KEMs with full IND-ID-CPA-security are implied by IND-snaID-CPA-secure ID-KEMs. The latter are usually significantly easier to construct. This result also introduces a technique for leveraging continuous collision-resistance (CCR) in security proofs. The generic conversion is relatively efficient: it increases the size of public parameters, user secret keys, and ciphertexts by a factor of only $\mathcal{O}(\log k)$, where k is the secu-

rity parameter. Its major drawback is that the security reduction is extremely non-tight (but polynomial-time).[1]

We also show how CCR can be used to obtain the first ID-KEM with full IND-ID-CPA-security in the nearly black-box setting, without random oracles, and with very short ciphertexts, where the ciphertext overhead is only a *single* element from a prime-order group. The only previously known ID-KEM with such short ciphertexts and adaptive security is the construction of Boneh and Franklin [11], which only has a security proof in the ROM. Our scheme is based on the selectively-secure Boneh-Boyen IBE scheme [7] and proven secure under a very strong (but non-interactive) q-type assumption, therefore we only view this as a feasibility result that shows that adaptively-secure ID-KEMs with such short ciphertext overhead exist.

Applications to digital signatures. Recall that the commonly accepted security notion for digital signatures is *existential unforgeability under adaptive chosen-message attacks* (EUF-CMA). There are several different ways to turn signatures schemes with weaker security properties into one with full EUF-CMA-security, even without random oracles. These are either based on one-time signatures [24] or chameleon hash functions [35, 15, 38], and work generically for any signature scheme. However, all these generic constructions start from an *existentially-unforgeable* scheme, where the adversary has to select the "chosen-message queries", for which it requests a signature, even before seeing the public key, but is able to choose the "target-message" for which it forges a signatures adaptively (EUF-naCMA-security, see Section 4.1 for formal definitions).

In this work, we consider the much weaker notion of *selective unforgeability under non-adaptive chosen-message attacks* (SUF-naCMA) [32, 16], where the adversary has to select both the "target-message" for which it forges a signatures and the chosen-message queries for which it requests a signature already before seeing the public key. We describe a generic construction of EUF-CMA-secure digital signatures from signatures that are only SUF-naCMA-secure. This construction is also relatively efficient: it increases the size of public keys, secret keys, and signatures by a factor of only $\mathcal{O}(\log k)$, where k is the security parameter. Again, the major drawback is that the security reduction is extremely non-tight (but polynomial-time).

We also consider the construction of digital signature schemes with very short signatures. The first scheme where a signature consists only of a *single* element of a prime-order bilinear group and which achieves full EUF-CMA-security is due to Boneh, Lynn and Shacham [13, 14]. However, this scheme is only known to be secure in the programmable ROM. A scheme with standard-model security proof and very short signatures of only a single group element is due to Boneh and Boyen [9]. This scheme can be proven secure without random oracles, but it achieves only non-adaptive EUF-naCMA-security. One can make this scheme

[1] However, we note that a corresponding ELF-based construction seems to incur a tightness loss of similar size. Even constructions in the random oracle model often require an inherent security loss [22, 34, 1], but smaller than in our case.

adaptively-secure in the standard model by applying the aforementioned generic conversions [24, 35, 15, 38], however, all these constructions increase the size of signatures by adding at least one additional element of bit-size $\Omega(k)$, where k is the security parameter. There are direct constructions of adaptively-secure signature schemes for bilinear groups, but for all these schemes the size of signatures is larger than one group element. This includes, for instance, the scheme of Waters [39] and its variants [30, 6], where a signature consists of two group elements, or the short signature schemes from [31, 29], where a signature consists of one group element plus an additional random string (whose size depends on an upper bound on the number of signatures to be issued per public key).

In summary, prior to the present work it was not clear that it is even possible to construct a digital signature scheme over bilinear groups, where signatures consist of only a single group element, and which achieves full EUF-CMA-security based on a non-interactive complexity assumption in the standard model. Constructing such a scheme is a major open problem in the domain of digital signatures. We give a construction which is based on the short signature scheme of Boneh and Boyen [9], but applies a continuously collision-resistant hash function to achieve adaptive security in the nearly black-box setting and without random oracles. Again, this scheme is based on a very strong (but non-interactive) q-type assumption and should be seen as a feasibility result.

Leveraging continuous collision resistance. In order to sketch how continuous collision resistance can be used in a security proof, let us consider the case of short digital signatures as an example. We work in the bilinear group setting, where we have groups $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ of prime order p, and an efficiently computable bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Signatures consist of only one element of $\mathbb{G}_1$. A secret key consist of $\ell = \log 4(k+1)$ elements $x_1, \ldots, x_\ell \in \mathbb{Z}_p$, where k is the security parameter. The corresponding public key consists of one element of $\mathbb{G}_1$ plus $4(k+1)$ elements of $\mathbb{G}_2$. Thus, public keys are larger than for the random-oracle-based short signature scheme of Boneh, Lynn, and Shacham [14], but the signature size is identical.

Computing a signature on a message m works as follows. For a cryptographic hash function $H : \{0,1\}^* \to \{0,1\}^{4(k+1)}$, let us write $H_{2^j}(m)$ to denote the first 2^j bits of $H(m)$. In order to sign a message m, we first compute

$$G(m) = \prod_{j=1}^{\ell} (x_j + H_{2^j}(m)) \bmod p$$

Note that one can perform this computation very efficiently, as it involves only elementary operations over $\mathbb{Z}_p$. Finally, the signature for m is

$$\sigma = g_1^{1/G(m)} \in \mathbb{G}_1$$

where $g_1 \in \mathbb{G}_1$ is a generator. Thus, computing a signature requires to perform only a *single* exponentiation in $\mathbb{G}_1$, plus a small number of additional operations in $\mathbb{Z}_p$.

A signature can be verified by first computing $g_2^{G(m)} \in \mathbb{G}_2$ from the group elements contained in the public key, which involves $\mathcal{O}(k)$ operations in $\mathbb{Z}_p$, $\mathcal{O}(k)$ multiplications in $\mathbb{G}_2$, and then testing whether

$$e(\sigma, g_2^{G(m)}) \stackrel{?}{=} e(g_1, g_2)$$

Note that this test requires only a single application of the bilinear map e to compute $e(\sigma, g_2^{G(m)})$, because the term $e(g_1, g_2)$ is independent of the given message and signature, and can thus be precomputed.

In order to sketch how continuous collision resistance is used in the security proof of this scheme, note that a signature of this scheme has the form

$$\sigma = g_1^{1/\prod_{j=1}^{\ell}(x_j + H_{2^j}(m))} \tag{1}$$

which can be viewed as an aggregation of ℓ signatures of the form

$$\sigma_j = g_1^{1/(x_j + H_{2^j}(m))} \tag{2}$$

In order to describe the intuition behind the security proof, let us view a signature σ therefore as an ℓ-tuple $\sigma = (\sigma_1, \ldots, \sigma_\ell)$ for now, where σ_j is as in (2). We describe later how these signatures can be aggregated to obtain our actual scheme. Note that each σ_j is a Boneh-Boyen signature [9] over the first 2^j bits of $H(m)$. In the security proof, we will choose j such that it simultaneously achieves the following two properties.

1. The index j is *sufficiently small*. Let m^* be the message for which the assumed adversary $\mathcal{A}$ forges a signature. We want that j is small enough, such that that we can guess $H_{2^j}(m^*) \in \{0,1\}^{2^j}$ with reasonable success probability, even *before* the security experiment starts, and we are able to prepare signatures for *all other* values $\{0,1\}^{2^j} \setminus H_{2^j}(m^*)$.
2. At the same time, we will make sure that the index j is *sufficiently large*, such that it is "sufficiently unlikely" that the adversary finds a collision for H_{2^j}. More precisely, we want that it is "sufficiently unlikely" that the adversary ever requests a signature for a message m_i and then outputs a forgery for message m^* with $H_{2^j}(m_i) = H_{2^j}(m^*)$.

The main difficulty of our security analysis lies in the second property, therefore let us consider this one more closely. Continuous collision resistance basically guarantees that there is no algorithm that finds collisions with significantly better time-to-success ratio than the standard birthday collision algorithm, even when prefixes $H_{2^j}(x)$ of hash values $H(x)$ are considered. Of course we will not be able to choose j such that the probability that $\mathcal{A}$ finds a collision is *negligibly* small – at least not without sacrificing the first condition, which we cannot afford. However, we will be able to choose j such that we can argue that the probability that $\mathcal{A}$ finds a collision for H_{2^j} is at most $\epsilon/2$, where ϵ is the success probability of $\mathcal{A}$ in breaking our signature scheme. This is "sufficiently unlikely", because it means: while *sometimes* $\mathcal{A}$ may break the security of the signature scheme

by finding a collision, at least *sometimes* (more precisely: with probability at least $\epsilon/2$) the adversary will also be able to break the signature scheme *without* finding a collision. This allows us to reduce the full EUF-CMA-security of our scheme to the SUF-naCMA-security of the underlying Boneh-Boyen scheme.

Since Boneh-Boyen signatures are not known to be efficiently aggregable in the sense of [12, 36], we have to overcome another hurdle to obtain a short signature scheme. However, this is relatively simple. Note that computing a signature that satisfies (1) essentially yields a "polynomial in the exponent" in unkowns $x_1, \ldots, x_\ell$ of degree ℓ. In order to verify whether a given value σ indeed satisfies (1) using a bilinear map, we therefore must to be able to compute group elements of the form

$$g^{x_1^{b_1} \cdots x_\ell^{b_\ell}} \tag{3}$$

for all possible values of $b_1, \ldots, b_\ell \in \{0, 1\}$. Note that these are 2^ℓ different values, but we have $\ell = \mathcal{O}(\log k)$ This allows us to include all required values of the form (3) in the public key, which yields a public key of size $\mathcal{O}(k)$.

Further related works. Further works which aim at instantiating random oracles include the work of Bellare, Hoang, and Keelvedhi [2] proposes a new security property of hash functions, called *Universal Computational Extractors*. This is not a single assumption, but rather a framework of very strong assumptions, some of which are unattainable or have seriously been questioned [17, 5]. It is currently unclear and subject to ongoing research how "strong" or "realistic" these assumptions exactly are. Furthermore, the programmable hash functions (PHFs) of Hofheinz and Kiltz [31] realize a (restricted, but very useful) form of programmability in the standard model. All known constructions of PHFs [31, 29, 27, 21] are based on algebraic public-key techniques.

There exists several generic and semi-generic constructions of strongly-secure signatures from signatures with weaker security [24, 15, 38, 32, 16]. All these works have in common that they can be applied only to signature scheme, but not to identity-based schemes like ID-KEMs, because they are either probabilistic (e.g., based on ephemeral one-time signatures or chameleon hash functions), or consider a setting with a non-adaptive adversary, which is forced to output all chosen-message queries before seeing the public key. The main difference between our work and the prefix-guessing technique of [32, 16] is that we essentially guess a short prefix of the hash of the "target message" *directly*, exploiting the continuous collision resistance to argue that this hash can not be equal to the hash of any chosen-message query. We do not have to know any chosen-message queries of the adversary to do this, which makes the technique also applicable to identity-based schemes like ID-KEMs. In contrast, the prefix-guessing technique of [32, 16] guesses the shortest prefix of the target message that is not equal to a prefix of a chosen-message query, which depends on the chosen-message queries made by the adversary and therefore can only be used to construct non-adaptively secure signatures (adaptive security is then achieved in a second step, e.g. using [24]).

2 Continuous Collision-Resistant Hashing

In this section we will formalize our new security notion for cryptographic hash functions. Intuitively, we will need a hash function H with the property that there exists no algorithm which finds collisions with significantly better "work factor" [3] than the trivial birthday collision algorithm. This should hold even if the output of the hash function is *truncated*.

Computational model. We consider algorithms as Turing machines, which operate on a binary alphabet and write their output bit-by-bit to an output tape, where each written bit takes one elementary machine operation. The *running time* of an algorithm is defined as the number of elementary operations performed by this machine.

Continuous collision resistance. Let $\mathcal{H}$ be a family of hash functions with domain $\{0,1\}^*$ and range $\{0,1\}^\alpha$. For $H \in \mathcal{H}$ let H_i denote the function obtained by evaluating H and truncating its output to the first i bits. Thus, $H_1(x)$ consists of the first bit of $H(x)$, and $H_\alpha(x) = H(x)$ for all $x \in \{0,1\}^*$.

Definition 1. *We say that $\mathcal{H}$ is a family of* continuously collision resistant *(CCR) hash functions, if for each adversary $\mathcal{B}$ that runs in time $t_\mathcal{B}$ holds that*

$$\Pr\left[\begin{array}{l} H \xleftarrow{\$} \mathcal{H}, (x_0, \ldots, x_q) \xleftarrow{\$} \mathcal{B}(H) : \\ \exists u, v \ s.t.\ H_i(x_u) = H_i(x_v) \wedge x_u \neq x_v \end{array}\right] \leq \frac{t_\mathcal{B}(t_\mathcal{B}-1)}{2^{i+1}}$$

for all $i \in \{1, \ldots, \alpha\}$.

The bound $t_\mathcal{B}(t_\mathcal{B}-1)/2^{i+1}$ in the above definition is derived from the birthday bound, which states that an adversary which evaluates a random function with range 2^i at most q times will find a collision with probability at most $q(q-1)/2^{i+1}$. Thus, Definition 1 requires essentially that no adversary is able to find collisions significantly better than by executing a standard birthday attack.

Constructing hash families from standard hash functions. Let H' be any standard cryptographic hash function, such as SHA-3, for example. We can construct a hash function family $\mathcal{H}$ as

$$\mathcal{H} := \{H : H(x) := H'(r||x), r \in \{0,1\}^k\}$$

A uniformly random hash function H from the family is chosen by selecting a uniformly random bit string $r \in \{0,1\}^k$. $H(x)$ is evaluated by computing $H'(r||x)$.

Strength of the CCR assumption. We view continuous collision resistance as a very natural security property for cryptographic hash functions. In particular, it seems to be satisfied by standard hash functions like SHA-256 or SHA-3. The standard way to determine the size of the output of a hash function in practice

is to fix a security parameter k, and to take a hash function of output size $2k$. For example, choosing SHA-256 (which has 256-bit hash values) for $k = 128$ is considered an adequate choice in practice, if collision-resistance is required. Note that one essentially assumes here already that there is no significantly better collision attack than the generic birthday algorithm. Our new complexity assumption generalizes this assumption to *prefixes* of the hash function, which appears very reasonable for standard hash functions.

We also note that for our applications given below we will require hash functions with output length of $4(k+1)$, rather than the "minimal" $2k$. For example, for $k = 127$ we would use SHA-512.

Choice of computational model and weakening the CCR assumption. Assume a computational model where an algorithm is able to output many pairwise distinct values $x_0, \ldots, x_q$ in a *single* elementary machine operation, and thus within a single time unit. Note that such an algorithm would be able to trivially break the CCR-assumption. To overcome this obstacle, we are working in a computational model where algorithms are assumed to write their output bit-by-bit to an output tape.

In order to generalize this is to a computational model where algorithms are able to output any constant number of bits in parallel in a single step, we can weaken Definition 1 by increasing the size of the prefix for which the adversary has to find a collision. To this end, one would replace the requirement

$$H_i(x_u) = H_i(x_v)$$

in Definition 1 with

$$H_{i+c}(x_u) = H_{i+c}(x_v)$$

for some small constant value c (e.g., $c \in \{1, ..., 10\}$). This also allows to add some additional "safety margin" to the CCR assumption, if desired, at the cost of an additional constant tightness loss factor of 2^c in the security proofs of our constructions. In the remainder of the paper, we will work with the original Definition 1, as it simplifies the exposition of our main results.

Useful technical lemma. We now state a technical lemma, which will be useful to leverage continuous collision resistance in security proofs. Intuitively, the lemma will provide bounds to ensure in our security proofs that for each adversary with some running time t and success probability ϵ there always exists a "good" index j such that H_{2^j} is "sufficiently collision-resistant", but at the same time the range $\{0,1\}^{2^j}$ of H_{2^j} is "sufficienly small". As usual, all logarithms are to base 2 in the sequel.

Lemma 1. *Let $t \in \mathbb{N}$ and $\epsilon \in (0, 1]$ with $t/\epsilon < 2^k$, and $j := \left\lfloor \log\log(4t^2/\epsilon) \right\rfloor + 1$. Then it holds that*

$$j \in \{1, \ldots, \log 4(k+1)\} \qquad \textit{and} \qquad \frac{4t^2}{2^{2^j+1}} < \frac{\epsilon}{2} \qquad \textit{and} \qquad 2^{2^j} \leq \left(\frac{4t^2}{\epsilon}\right)^2$$

PROOF. We first show that $j \in \{1, \ldots, 2 + \log(k+1)\}$. Since $\epsilon \neq 0$, we trivially have $j \geq 1$. Additionally using that $\epsilon \in (0,1]$ and $t/\epsilon < 2^k$, we obtain

$$\begin{aligned} j &= \left\lfloor \log\log(4t^2/\epsilon) \right\rfloor + 1 \leq \log\log(4t^2/\epsilon) + 1 \\ &< \log\log(2^{2k+2}) + 1 = \log 4(k+1) \end{aligned}$$

To show $4t^2/2^{2^j+1} < \epsilon/2$, we compute

$$\frac{4t^2}{2 \cdot 2^{2^j}} = \frac{4t^2}{2 \cdot 2^{2^{\lfloor \log\log(4t^2/\epsilon) \rfloor + 1}}} < \frac{4t^2}{2 \cdot 2^{2^{\log\log(4t^2/\epsilon)}}} = \frac{4t^2}{2 \cdot (4t^2/\epsilon)} = \frac{\epsilon}{2}$$

Finally, we get $2^{2^j} \leq \left(4t^2/\epsilon\right)^2$ from

$$2^{2^j} = 2^{2^{\left\lfloor \log\log(4t^2/\epsilon) \right\rfloor + 1}} \leq 2^{2 \cdot 2^{\log\log(4t^2/\epsilon)}} = (4t^2/\epsilon)^2$$

□

3 Identity-based key encapsulation

In this section, we show how continuous collision resistance yields a generic construction of adaptively-secure ID-KEMs from ID-KEMs with weaker security, and a construction of ID-KEMs with very short ciphertexts and without random oracles.

3.1 Definitions and security notions

Definition 2. *An ID-KEM consists of the following four PPT algorithms:*

$\mathsf{Setup}(1^k)$ *returns the public parameters PP and the master secret key MSK. We assume that PP defines (implicitly or explicitly) an identity space $\mathcal{I}$, a key space $\mathcal{K}$ and a ciphertext space $\mathcal{C}$.*

$\mathsf{KeyGen}(MSK, id)$ *returns the user secret key USK_{id} for identity $id \in \mathcal{I}$.*

$\mathsf{Encap}(PP, id)$ *returns a tuple (C, K), where $K \in \mathcal{K}$ is a key and $C \in \mathcal{C}$ is a ciphertext encapsulating K with respect to identity id.*

$\mathsf{Decap}(USK_{id}, C, id)$ *returns the decapsulated key $K \in \mathcal{K}$ or an error symbol $\perp$.*

For perfect correctness we require that for all $k \in \mathbb{N}$, all pairs (PP, MSK) generated by $\mathsf{Setup}(1^k)$, all identities $id \in \mathcal{I}$, all (K, C) output by $\mathsf{Encap}(PP, id)$ and all USK_{id} generated by $\mathsf{KeyGen}(MSK, id)$:

$$\Pr[\mathsf{Decap}(USK_{id}, C, id) = K] = 1.$$

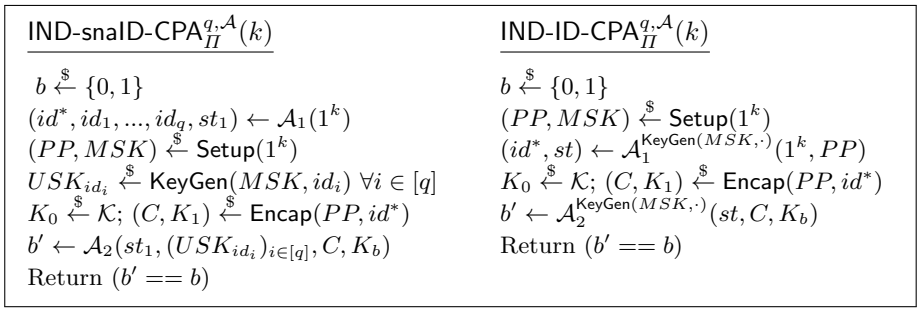

Fig. 1. The security experiments for ID-KEMs, executed with scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ and adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The oracle $\mathsf{KeyGen}(MSK, id)$ returns $USK_{id} \xleftarrow{\$} \mathsf{KeyGen}(MSK, id)$ with the restriction that $\mathcal{A}$ is not allowed to query oracle $\mathsf{KeyGen}(MSK, \cdot)$ for the target identity id^*.

Adaptive security. Let us first recall the standard CPA-security notion for ID-KEMs. To this end, consider the IND-ID-CPA security experiment depicted in Figure 1. Note that the adversary may choose both the challenge-identity id^* and the chosen-key query identities $id_1, \ldots, id_q$ adaptively.

Definition 3. *We say that adversary $\mathcal{A}$ $(t_{\mathcal{A}}, q, \epsilon_{\mathcal{A}})$- breaks the* IND-ID-CPA *security of Π, if*

$$\Pr[\mathsf{IND\text{-}ID\text{-}CPA}^{q,\mathcal{A}}_{\Pi}(k) \Rightarrow 1] - \frac{1}{2} \geq \epsilon_{\mathcal{A}}$$

and $t_{\mathcal{A}}$ is the running time of $\mathcal{A}$ including *the* IND-ID-CPA *security experiment.*

Remark 1. Including the running time of the security experiment into the running time of the adversary $\mathcal{A}$ will later allow us to simplify the analysis of our security reduction.

Selective and non-adaptive security. We also define a very weak security notion for ID-KEMs. Consider the IND-snaID-CPA security experiment depicted in Figure 1, where the attacker has to commit to both the challenge-ID id^* the key-query identities $id_1, \ldots, id_q$ non-adaptively and even before receiving the master public key PP.

Definition 4. *We say that $\mathcal{A}$ $(t_{\mathcal{A}}, q, \epsilon_{\mathcal{A}})$-breaks the* IND-snaID-CPA *security of Π, if it runs in time $t_{\mathcal{A}}$ and*

$$\Pr[\mathsf{IND\text{-}snaID\text{-}CPA}^{q,\mathcal{A}}_{\Pi}(k) \Rightarrow 1] - \frac{1}{2} \geq \epsilon_{\mathcal{A}}.$$

3.2 From weak security to adaptive security

Construction. Let $\mathcal{H}$ be a family of continuously collision-resistant hash functions $H : \{0,1\}^* \to \{0,1\}^{4(k+1)}$ and let

$$\ell := \log(4(k+1))$$

Let $\Pi' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Encap}', \mathsf{Decap}')$ be an ID-KEM. We construct scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ as follows. Below we will prove that the IND-ID-CPA-security of Π is implied by the IND-snaID-CPA-security of Π'.

Setup. Compute $(PP_i, MPK_i) \xleftarrow{\$} \mathsf{Setup}'(1^k)$ for all $i \in \{1, \ldots, \ell\}$, choose $H \xleftarrow{\$} \mathcal{H}$ and define

$$PP = (PP_1, \ldots, PP_\ell, H) \qquad \text{and} \qquad MSK = (MSK_1, \ldots, MSK_\ell, H).$$

and output (PP, MSK).

User Key Generation. To create a private key for the identity id, compute $USK_i \xleftarrow{\$} \mathsf{KeyGen}'(MSK_i, H_{2^i}(id))$ for all $i \in \{1, \ldots, \ell\}$. Define

$$USK_{id} := (USK_1, \ldots, USK_\ell)$$

and output USK_{id}.

Encapsulation. On input $PP = (PP_1, \ldots, PP_\ell, H)$ and id, compute $(K_i, C_i) \xleftarrow{\$} \mathsf{Encap}'(PP_i, H_{2^i}(id))$ for all $i \in \{1, \ldots, \ell\}$. Then define

$$K := \bigoplus_{i=1}^{\ell} K_i,$$

where $\bigoplus$ denotes the XOR-operation and output $(C, K) = ((C_1, ..., C_\ell), K)$.

Decapsulation. On input $C = (C_1, ...C_\ell)$ and USK_{id}, compute

$$K_i = \mathsf{Decap}'(usk_i, C_i).$$

for all $i \in \{1, \ldots, \ell\}$ and output

$$K := \bigoplus_{i=1}^{\ell} K_i.$$

The correctness of Π follows immediately from the correctness of Π'.

Security analysis. Recall that we have $\ell := \log 4(k+1)$, and that Lemma 1 shows that for each adversary $\mathcal{A}$ with $t_\mathcal{A}/\epsilon_\mathcal{A} < 2^k$, there exists an index $j \in \{1, \ldots, \ell\}$ such that

$$j = \left\lfloor \log \log 4t_\mathcal{A}^2/\epsilon_\mathcal{A} \right\rfloor + 1 \tag{4}$$

is satisfied. The following theorem assumes that this value of j is given. Thus, exactly as in Zhandry's ELF paper [40], the reduction is non-black-box. Since only an approximation of running time and success probability of the adversary is required, this was called "nearly" black-box in [40]. We will later generalize this to fully black-box reductions.

Theorem 1. *Let $\mathcal{A}$ be an adversary that $(t_{\mathcal{A}}, q_{\mathcal{A}}, \epsilon_{\mathcal{A}})$-breaks the* IND-ID-CPA*-security of Π with $\epsilon_{\mathcal{A}} > 0$ and $t_{\mathcal{A}}/\epsilon_{\mathcal{A}} < 2^k$. Given $\mathcal{A}$ and an index j such that (4) is satisfied, we can construct an adversary $\mathcal{B}_j$ that $(t_{\mathcal{B}}, q_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-breaks the* IND-snaID-CPA*-security of Π' with*

$$t_{\mathcal{B}} = \mathcal{O}(t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2), \qquad q_{\mathcal{B}} < 4t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2 \quad \textit{and} \quad \epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}^3}{32t_{\mathcal{A}}^4}.$$

Note that the theorem considers adversaries that for a given security parameter k have "work factor" $t_{\mathcal{A}}/\epsilon_{\mathcal{A}}$ below 2^k. This deviates from the common asymptotic definition, where $t_{\mathcal{A}}$ is polynomially-bounded and $\epsilon_{\mathcal{A}}$ is non-negligible. Assuming $t_{\mathcal{A}}/\epsilon_{\mathcal{A}} < 2^k$ is an alternative way of expressing that a cryptosystem is secure with respect to a given security parameter k that originates from the "concrete security" approach of Bellare and Ristenpart [3]. Note also that the security loss of reduction $\mathcal{B}$ is polynomially-bounded, but relatively large.

Proof. Before we are able to construct $\mathcal{B}_j$, we have to make a couple of modifications to the IND-ID-CPA security experiment. Consider the following sequence of games, where we denote with G_i the event that Game i outputs 1.

Game 0. This is the $\mathsf{IND\text{-}ID\text{-}CPA}_{\Pi}^{q,\mathcal{A}}(k)$ security experiment. By definition, we have

$$\Pr[G_0] = \Pr\left[\mathsf{IND\text{-}ID\text{-}CPA}_{\Pi}^{q,\mathcal{A}}(k) \Rightarrow 1\right].$$

Game 1. From this game on, we use that an index $j \in \{1, \ldots, \ell\}$ is given, such that (4) is satisfied. Furthermore, we define event coll_j, which occurs when the adversary $\mathcal{A}$ in the IND-ID-CPA experiment ever finds a collision for H_{2^j}. More precisely, $\mathcal{A}$ queries a user secret key or challenge ciphertext for identities id, id' such that $id \neq id'$, but

$$H_{2^j}(id) = H_{2^j}(id').$$

If coll_j occurs, then Game 1 outputs a random bit and aborts.

Note that $\Pr[G_1 \wedge \neg\mathsf{coll}_j] = \Pr[G_0 \wedge \neg\mathsf{coll}_j]$, and therefore

$$|\Pr[G_0] - \Pr[G_1]| \leq \Pr[\mathsf{coll}_j]$$

by the Difference Lemma [37]. In particular, since $\Pr[G_0] \geq \Pr[G_1]$ (as Game 0 considers an adversary with advantage that can only be decreased by our abort condition), we have

$$\Pr[G_1] \geq \Pr[G_0] - \Pr[\mathsf{coll}_j] = 1/2 + \epsilon_{\mathcal{A}} - \Pr[\mathsf{coll}_j].$$

We use the continuous collision resistance of H to show $\Pr[\mathsf{coll}_j] \leq \epsilon/2$. Consider an algorithm $\mathcal{C}$, which runs the IND-ID-CPA security experiment with $\mathcal{A}$ and outputs the list of identities $(id_1, \ldots, id_q, id^*)$ for which $\mathcal{A}$ requests a user secret key or the challenge ciphertext. Note that the running time of $\mathcal{C}$ is at most $2 \cdot t_{\mathcal{A}}$, since $t_{\mathcal{A}}$ includes the running time of the security experiment (cf.

Definition 3) and the only additional operation performed by $\mathcal{C}$ is to output the list of identities, which takes at most time $t_{\mathcal{A}}$. By construction, $\mathcal{C}$ finds a collision for H_{2^j} if and only if coll_j occurs. Continuous collision resistance guarantees that this probability is at most $2t_{\mathcal{A}}(2t_{\mathcal{A}}-1)/2^{2^j+1}$, which yields

$$\Pr[\mathsf{coll}_j] \leq 2t_{\mathcal{A}}(2t_{\mathcal{A}}-1)/2^{2^j+1} \leq 4t_{\mathcal{A}}^2/2^{2^j+1} < \epsilon/2,$$

where the last inequality applies Lemma 1 and the choice of j. This shows that

$$\Pr[G_1] > 1/2 + \epsilon_{\mathcal{A}}/2.$$

Game 2. In Game 2, we additionally guess $ID^* \xleftarrow{\$} \{0,1\}^{2^j}$ uniformly random, and raise event $\mathsf{abort}_{\mathsf{chal}}$, output a random bit, and abort, if adversary $\mathcal{A}$ requests a challenge ciphertext for identity id^* with $H_{2^j}(id^*) \neq ID^*$.

Since ID^* is chosen uniformly random and independent of the view of the adversary, and $G_2 \wedge \neg\mathsf{abort}_{\mathsf{chal}} \iff G_1 \wedge \neg\mathsf{abort}_{\mathsf{chal}}$ we have

$$\begin{aligned}
\Pr[G_2] &= \Pr[G_2 \mid \mathsf{abort}_{\mathsf{chal}}] \cdot \Pr[\mathsf{abort}_{\mathsf{chal}}] + \Pr[G_2 \wedge \neg\mathsf{abort}_{\mathsf{chal}}] \\
&= \Pr[G_2 \mid \mathsf{abort}_{\mathsf{chal}}] \cdot \Pr[\mathsf{abort}_{\mathsf{chal}}] + \Pr[G_1 \wedge \neg\mathsf{abort}_{\mathsf{chal}}] \\
&= 1/2 \cdot (1 - \Pr[\neg\mathsf{abort}_{\mathsf{chal}}]) + \Pr[G_1] \cdot \Pr[\neg\mathsf{abort}_{\mathsf{chal}}] \\
&= 1/2 + (\Pr[G_1] - 1/2) \cdot \Pr[\neg\mathsf{abort}_{\mathsf{chal}}] \\
&> 1/2 + \frac{\epsilon_{\mathcal{A}}}{2} \cdot \Pr[\neg\mathsf{abort}_{\mathsf{chal}}].
\end{aligned}$$

Since ID^* is chosen uniformly random from a set of size 2^{2^j}, the probability of guessing ID^* correctly is $\Pr[\neg\mathsf{abort}_{\mathsf{chal}}] = 2^{-2^j}$. By applying Lemma 1, which guarantees that $2^{2^j} \leq \left(\frac{4t^2}{\epsilon}\right)^2$ for our choice of j, we get

$$\Pr[G_2] > 1/2 + \frac{\epsilon_{\mathcal{A}}}{2} \cdot \Pr[\neg\mathsf{abort}_{\mathsf{chal}}] = 1/2 + \frac{\epsilon_{\mathcal{A}}}{2} \cdot \frac{1}{2^{2^j}} \geq 1/2 + \frac{\epsilon_{\mathcal{A}}^3}{32t_{\mathcal{A}}^4}.$$

Now we are ready to construct our reduction algorithm $\mathcal{B}_j$, which simulates Game 2 for $\mathcal{A}$.

Construction of algorithm $\mathcal{B}_j$. Algorithm $\mathcal{B}_j$ receives as input a security parameter k from the $\mathsf{IND\text{-}snaID\text{-}CPA}$ experiment. It simulates Game 2 (including the handling of events coll_j and $\mathsf{abort}_{\mathsf{chal}}$) as follows.

$\mathcal{B}_j$ samples a challenge identity $ID^* \xleftarrow{\$} \{0,1\}^{2^j}$ uniformly random and defines $2^{2^j}-1$ identities $ID_1, \ldots, ID_{2^{2^j}-1}$, consisting of all values in $\{0,1\}^{2^j} \setminus \{ID^*\}$. It outputs these values to the $\mathsf{IND\text{-}snaID\text{-}CPA}$ experiment, which then generates and responds with a key pair $(PP', MSK') \xleftarrow{\$} \mathsf{Setup}'(1^k)$, user secret keys USK'_{ID_i}, $i \in \{1, \ldots, 2^{2^j}-1\}$, for the requested identities, and a challenge ciphertext (C', K'), where $(C', K) \xleftarrow{\$} \mathsf{Encap}'(PP', ID^*)$ and either $K' = K$ or K' is uniformly random.

Simulation of the public key. In order to simulate the public key, $\mathcal{B}_j$ generates $(PP_i, MPK_i) \xleftarrow{\$} \mathsf{Setup}'$ for all $i \in \{1, \ldots, \ell\} \setminus \{j\}$ and sets

$$PP = (PP_1, ..., PP_{j-1}, PP', PP_{j+1}, ..., PP_l).$$

Finally, $\mathcal{B}_j$ outputs PP to $\mathcal{A}$. Note that this is a correctly distributed master public key for scheme Π.

Answering key queries. $\mathcal{B}_j$ knows MSK_i for all $i \neq j$ and user secret keys for USK'_{ID_s} for all $ID_s \in \{0,1\}^{2^j} \setminus \{ID^*\}$. Therefore it is able to compute and return valid user secret keys to $\mathcal{A}$ for all identities id_z with $H_{2^j}(id_z) \neq ID^*$.

More precisely, whenever $\mathcal{A}$ requests a user secret key for an identity $id_z \in \{0,1\}^*$, $\mathcal{B}_j$ proceeds as follows. $\mathcal{B}_j$ will first check if $H_{2^j}(id_z) = H_{2^j}(id^*)$ (if id^* is already defined), or there exists an index $z' \in \{1, \ldots, q\}$ such that $H_{2^j}(id_z) = H_{2^j}(id_{z'})$. If this holds, then $\mathcal{B}_j$ raises event $\mathsf{coll_j}$, aborts the simulation and outputs a random bit. Furthermore, if $H_{2^j}(id_z) = ID^*$ then $\mathcal{B}_j$ raises event $\mathsf{abort_{chal}}$, aborts and outputs a random bit.

If there is no abort, then $\mathcal{B}_j$ computes $(USK_i) \xleftarrow{\$} \mathsf{KeyGen}'(MSK_i, H_{2^i}(id_z))$ for all $i \in \{1, \ldots, \ell\} \setminus \{j\}$. Recall that $\mathcal{B}_j$ has requested user secret keys for all values $H_{2^j}(id_z) \in \{0,1\}^{2^j}$ with $H_{2^j}(id_z) \neq ID^*$, in particular for $ID_s \in \{0,1\}^{2^j}$ such that $ID_s = H_{2^j}(id_z)$. Therefore it is able to efficiently determine and output

$$USK_{id_z} = (USK_1, ..., USK_{j-1}, USK'_{ID_s}, USK_{j+1}, ..., USK_l).$$

Computing the challenge ciphertext. When adversary $\mathcal{A}$ outputs a challenge identity id^*, $\mathcal{B}_j$ will first check if there exists an index $z \in \{1, \ldots, q\}$ such that $H_{2^j}(id_z) = H_{2^j}(id^*)$. If this holds, then $\mathcal{B}_j$ raises event coll_j, aborts the simulation and outputs a random bit. Else $\mathcal{B}_j$ checks whether $H_{2^j}(id^*) = ID^*$ and if this does *not* hold then, $\mathcal{B}_j$ raises event $\mathsf{abort_{chal}}$ and aborts, outputting a random bit. Otherwise, for all $i \in \{1, \ldots, \ell\} \setminus \{j\}$ it computes $(C_i, K_i) \xleftarrow{\$} \mathsf{Encap}'(PP_i, H_{2^i}(id^*))$, and then

$$K := \bigoplus_{i=1, i\neq j}^{\ell} K_i \oplus K' \text{ and } C := (C_1, \ldots, C_{j-1}, C', C_{j+1}, \ldots, C_\ell),$$

where (C', K') is the tuple received from the $\mathsf{IND\text{-}snaID\text{-}CPA}$-experiment. B_j returns (C, K) to $\mathcal{A}$ and outputs whatever $\mathcal{A}$ outputs.

Success probability of $\mathcal{B}_j$. Note that if K' is a "real" key, which holds with probability $1/2$, then so is K, while if K' is "random", then so is K. Hence, B_j simulates Game 2 perfectly, and we have

$$\Pr\left[\mathsf{IND\text{-}snaID\text{-}CPA}^{q,\mathcal{B}_j}_{\Pi'}(k) \Rightarrow 1\right] = \Pr[G_2] > 1/2 + \frac{\epsilon^3_{\mathcal{A}}}{32 t^4_{\mathcal{A}}}.$$

Running time of $\mathcal{B}_j$. The running time $t_{\mathcal{B}_j}$ of $\mathcal{B}_j$ consists of the time needed to execute $\mathcal{A}$, the time required to simulate the IND-ID-CPA security experiment, and the time required to request the $2^{2^j} - 1$ user secret keys from the IND-snaID-CPA experiment, plus a minor number of additional operations. Making use of Lemma 1, we get

$$t_j \approx t_{\mathcal{A}} + \mathcal{O}\big(2^{2^j} - 1\big) \approx t_{\mathcal{A}} + \mathcal{O}\big(\frac{t_{\mathcal{A}}^4}{\epsilon_{\mathcal{A}}^2}\big) = \mathcal{O}\big(\frac{t_{\mathcal{A}}^4}{\epsilon_{\mathcal{A}}^2}\big).$$

Note also that $\mathcal{B}_j$ issues $q_{\mathcal{B}} = 2^{2^j} - 1 < 4t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2$ user key queries. This completes the proof of Theorem 1. □

From "nearly" black-box to full black-box. Recall that Theorem 1 assumes that an index $j \in \{1, \ldots, \log 4(k+1)\}$ is given that satisfies $j = \big\lfloor \log \log 4t_{\mathcal{A}}^2/\epsilon_{\mathcal{A}} \big\rfloor + 1$ for adversary $\mathcal{A}$. Therefore the reduction $\mathcal{B}_j$ constructed to prove Theorem 1 is only "nearly black-box" in the sense of [40], because it essentially needs to know at least sufficiently close approximations of the running time $t_{\mathcal{A}}$ and the advantage $\epsilon_{\mathcal{A}}$ of $\mathcal{A}$. We will show that we can also construct a reduction $\mathcal{B}$ that simply guesses this value j, and otherwise proceeds exactly like the algorithm $\mathcal{B}_j$ from the proof of Theorem 1. This is possible in our setting, because the choice of j is completely oblivious to the adversary $\mathcal{A}$, unless the game is aborted. This holds even for computationally unbounded adversaries, because $\mathcal{B}_j$ provides a *perfect* simulation of the original security experiment, provided that it does not abort. We construct an algorithm $\mathcal{B}$ which simply guesses an index j uniformly random from $\{1, \ldots, \log 4(k+1)\}$ and then proceeds exactly like $\mathcal{B}_j$, hoping that j satisfies (4).

A minor difficulty that we face here is that the running time $t_{\mathcal{B}}$ of $\mathcal{B}$ depends exponentially on j, such that we can only get a reasonable bound on $t_{\mathcal{B}}$ if the guessed value of j approximates the running time and advantage of the adversary well-enough via the bounds from (4). The reduction may become inefficient if it guesses j too large, and may cease to have reasonable advantage if j is too small. Therefore, strictly speaking, adversary $\mathcal{B}$ is not an algorithm that runs in strict or expected polynomial time in terms of classical complexity theory. Nevertheless, if it guesses j correctly, which happens with probability at least $1/(\log 4(k+1))$, then it has a polynomially-bounded running time and non-negligible advantage. This yields a somewhat cumbersome formulation of the following theorem, but is still sufficient to establish that we can turn any efficient adversary $\mathcal{A}$ against Π into an efficient adversary $\mathcal{B}$ against Π', and thus to prove the construction secure *without* requiring *a priori* knowledge about $\mathcal{A}$'s running time or success probability.

Theorem 2. *Let $\mathcal{A}$ be an adversary that $(t_{\mathcal{A}}, q_{\mathcal{A}}, \epsilon_{\mathcal{A}})$-breaks the* IND-ID-CPA*-security of Π with $\epsilon_{\mathcal{A}} > 0$ and $t_{\mathcal{A}}/\epsilon_{\mathcal{A}} < 2^k$. Given $\mathcal{A}$, we can construct an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, q_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-breaks the security of Π' in the* IND-snaID-CPA*-security experiment such that when $\mathcal{B}$ is executed, then with probability at least*

$1/\log 4(k+1)$ *all the following bounds hold simultaneously:*

$$t_{\mathcal{B}} \approx \mathcal{O}\left(\frac{t_{\mathcal{A}}^4}{\epsilon_{\mathcal{A}}^2}\right), \qquad q_{\mathcal{B}} < 4t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2 \qquad \text{and} \qquad \epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}^3}{32t_{\mathcal{A}}^4}.$$

Thus, the total advantage of $\mathcal{B}$ *is at least* $1\epsilon_{\mathcal{A}}^3/(32t_{\mathcal{A}}^4 \log 4(k+1))$.

Proof. Adversary $\mathcal{B}$ picks $j \xleftarrow{\$} \{1, ..., \log 4(k+1)\}$ uniformly random. Then it proceeds exactly like adversary $\mathcal{B}_j$ from the proof of Theorem 1.

Success probability of $\mathcal{B}$. To analyze the probability of $\mathcal{B}$, let corr denote the event that $\mathcal{B}$ guesses j correctly, that is, the value of j chosen by $\mathcal{B}$ satisfies (4) with respect to $\mathcal{A}$. Furthermore, write

$$X := \mathsf{IND\text{-}snaID\text{-}CPA}_{\Pi'}^{q,\mathcal{B}}(k) \Rightarrow 1$$

and set $\ell := \log 4(k+1)$ to simplify our notation. Then we have

$$\begin{aligned}
\Pr[X] &= \Pr[X \mid \mathsf{corr}] \cdot \Pr[\mathsf{corr}] + \Pr[X \mid \neg\mathsf{corr}] \cdot \Pr[\neg\mathsf{corr}] \\
&= \Pr[X \mid \mathsf{corr}] \cdot \frac{1}{\ell} + \Pr[X \mid \neg\mathsf{corr}] \cdot \left(1 - \frac{1}{\ell}\right) \\
&= \Pr[X \mid \neg\mathsf{corr}] + \frac{1}{\ell} \cdot (\Pr[X \mid \mathsf{corr}] - \Pr[X \mid \neg\mathsf{corr}]).
\end{aligned}$$

Writing $\Pr[X \mid \neg\mathsf{corr}] = 1/2 + \alpha$ for some $\alpha \in [-1/2, 1/2]$, we obtain

$$\begin{aligned}
\Pr[X] &= 1/2 + \alpha + \frac{1}{\ell} \cdot (\Pr[X \mid \mathsf{corr}] - 1/2 - \alpha) \\
&= 1/2 + \frac{1}{\ell} \cdot (\Pr[X \mid \mathsf{corr}] - 1/2) + \frac{\ell - 1}{\ell}\alpha \\
&= 1/2 + \frac{1}{\ell} \cdot \frac{\epsilon_{\mathcal{A}}^3}{32t_{\mathcal{A}}^4} + \frac{\ell - 1}{\ell}\alpha,
\end{aligned}$$

where the last line follows from the bound on the advantage of $\mathcal{B}_j$ given by Theorem 1. In order to prove the theorem, it suffices to show that it always holds that $\alpha \geq 0$. To this end, let us consider the probability $\Pr[X \mid \neg\mathsf{corr}]$, recalling that

$$1/2 + \alpha = \Pr[X \mid \neg\mathsf{corr}].$$

That is, we consider the probability of event X, given that $\mathcal{B}$ did not guess j such that (4) is satisfied. We distinguish between two cases:

1. Adversary $\mathcal{B}_j$ aborts. Note that then it outputs a uniformly random bit, and therefore we have $\alpha = 0$.
2. Adversary $\mathcal{B}_j$ does not abort. Even though j is not guessed correctly, the view simulated by $\mathcal{B}$ to $\mathcal{A}$ is perfectly indistinguishable from the original security experiment, and thus we have $\alpha \geq 0$.

More formally, let denote abort the event that $\mathcal{B}_j$ aborts the simulation. Then we can compute

$$\begin{aligned}
1/2+\alpha =& \Pr\left[X \mid \neg\mathsf{corr}\right] \\
=& \Pr\left[X \mid \neg\mathsf{corr} \wedge \mathsf{abort}\right] \cdot \Pr\left[\mathsf{abort} \mid \neg\mathsf{corr}\right] \\
&+ \Pr\left[X \mid \neg\mathsf{corr} \wedge \neg\mathsf{abort}\right] \cdot \Pr\left[\neg\mathsf{abort} \mid \neg\mathsf{corr}\right] \\
=& 1/2 \cdot \Pr\left[\mathsf{abort} \mid \neg\mathsf{corr}\right] + (1/2+\epsilon_{\mathcal{A}}) \cdot \Pr\left[\neg\mathsf{abort} \mid \neg\mathsf{corr}\right] \\
=& 1/2 + \epsilon_{\mathcal{A}} \cdot \Pr\left[\neg\mathsf{abort} \mid \neg\mathsf{corr}\right] \\
\geq& 1/2.
\end{aligned}$$

Running time of $\mathcal{B}$. Assuming that the guess of j was right in the sense of (4) we get the same running time like for $\mathcal{B}_j$ in Theorem (1). This happens with probability $1/\log 4(k+1)$. □

Adopting this approach to the ELF-based setting [40]. A natural question to ask is why our result can be lifted from the "nearly" black-box to the fully black-box setting, while this seems not possible for the ELF-based constructions of [40]. The reason is that [40] uses a hybrid argument with a sequence of "DDH-steps", where the number of these steps depends on the guess of $t_{\mathcal{A}}$ and $\epsilon_{\mathcal{A}}$ (which corresponds to our guessing of index j), and is not information-theoretically hidden from $\mathcal{A}$. Thus, even if the simulation is not aborted, the view of the adversary is not independent of this guess. Hence, it may be possible that a given adversary $\mathcal{A}$ has a positive advantage only if the guess is correct, but a negative advantage if the guess is incorrect, such that in total the advantage of the reduction becomes void.

3.3 Adaptively secure ID-KEM with short ciphertexts

The generic construction of adaptively-secure ID-KEMs described in Section 3.2 increases the size of keys and ciphertexts by a factor of $\mathcal{O}(\log k)$. This overhead is not overly huge, but it is still interesting to ask whether it is possible to obtain more efficient schemes based on specific, number-theoretic constructions. In this section we describe a variant of the Boneh-Boyen IBE scheme [7] with extremely short ciphertexts consisting of only a *single* group element and full adaptive security.

Building block: simplified Boneh-Boyen ID-KEM. The following ID-KEM is based on the IBE scheme of Boneh and Boyen [7]. Let $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ be groups of prime order p and let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ an efficiently computable pairing. We will use the implicit notation of Escala *et al.* [23], and write $[x]_s$ shorthand for g_s^x for all $s \in \{1, 2, T\}$ and generators g_1, g_2, g_T of $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$, respectively.

Simple ID-KEM based on the Boneh-Boyen IBE scheme. We use the following scheme as a building block for our adaptively-secure ID-KEM.

Setup Choose random generators $[1]_1 \in \mathbb{G}_1, [1]_2 \in \mathbb{G}_2$ and two random elements $x, y \in \mathbb{Z}_p$. Then define $X = [x]_1$ and $\nu = e([1]_1, [1]_2^y)$. The published public parameters PP and the master secret key MSK are defined as

$$PP = ([1]_1, [x]_1, \nu) \text{ and } MSK = (x, y)$$

Key Generation To create a private key for identity $id \in \mathbb{Z}_p$, compute and return

$$USK_{id} = [y/(id + x)]_2.$$

Encapsulation. To encapsulate a key $K \in \mathbb{G}_T$ under public key $id \in \mathbb{Z}_p$, pick a random $r \in \mathbb{Z}_p$ and output

$$(C, K) = ([id + x]_1^r, \nu^r) \in \mathbb{G}_1 \times \mathbb{G}_T$$

Decapsulation. To decapsulate C using the private key USK_{id}, compute and output

$$e(C, USK_{id})$$

Correctness. This follows from

$$e([id + x]_1^r, [y/(id + x)]_2) = e([1]_1, [y]_2)^r = \nu^r.$$

Proving security of the simplified Boneh Boyen IBE. Consider the following experiment q-BDDHI(1^k), which was generalized to asymmetric bilinear groups in [10]. With regard to the security parameter k, the challenger generates an asymmetric pairing group and chooses $x \in \mathbb{Z}_p$ uniformly at random. Then it chooses $T \xleftarrow{\$} \mathbb{G}_T$ and defines

$$T_0 := ([1]_1, [x]_1, [1]_2, [x]_2, \ldots [x^q]_2, T)$$
$$T_1 := ([1]_1, [x]_1, [1]_2, [x]_2, \ldots [x^q]_2, e([1]_1, [1]_2)^{\frac{1}{x}}).$$

Finally, it flips a fair binary coin β and outputs T_β to the adversary. The task of adversary $\mathcal{B}$ is to determine β.

Definition 5. *We say that adversary $\mathcal{B}$ (t, ϵ)-solves the q-BDDHI problem, if it runs in time t and*

$$|\Pr[\mathcal{B}(T_0)] - \Pr[\mathcal{B}(T_1)]| \geq \epsilon$$

It is straightforward to prove the IND-snaID-CPA-security of our simplified Boneh-Boyen using standard techniques from [7, 10], therefore we state the following theorem without proof.

Theorem 3. *From an adversary $\mathcal{A}$ that $(t_\mathcal{A}, q_s, \epsilon_\mathcal{A})$-breaks the* IND-snaID-CPA*-security of the simplified Boneh-Boyen ID-KEM one can construct an algorithm $\mathcal{B}$ that $(t_\mathcal{B}, \epsilon_\mathcal{B})$-solves the q-BDDHI problem with $q = q_s + 1$ such that*

$$t_\mathcal{B} \approx t_\mathcal{A} \quad \text{and} \quad \epsilon_\mathcal{B} = \epsilon_\mathcal{A}$$

Adaptively-secure construction.

Encoding elements of $\{0,1\}^{4(k+1)}$ *as* $\mathbb{Z}_p$*-elements.* In the scheme described below, identities are elements in $\mathbb{Z}_p$. In order to simplify the notation and description of the construction and its security analysis, we will henceforth make the implicit assumption that elements of $\{0,1\}^{4(k+1)}$ can be injectively encoded as elements of $\mathbb{Z}_p$. This is of course easily possible by choosing p large enough, such that $p > 4(k+1)$. However, this would yield an unnaturally large group order (a typical choice in practice is $2k$). In practice, one would map elements of $\{0,1\}^{4(k+1)}$ to elements in $\mathbb{Z}_p$ by using a collision-resistant hash function $h : \{0,1\}^{4(k+1)} \to \mathbb{Z}_p$, which for our purposes is as good as an injective map. However, to simplify the description of our scheme and its security proof we do not make h explicit in the sequel.

The construction. In the sequel, let $\mathcal{H}$ be a family of continuously collision-resistant hash functions $H : \{0,1\}^* \to \{0,1\}^{4(k+1)}$ and define

$$\ell := \log 4(k+1)$$

We construct ID-KEM scheme $\Pi = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encap}, \mathsf{Decap})$ as follows.

Setup. Sample $H \xleftarrow{\$} \mathcal{H}$ and select random generators $[1]_1 \in \mathbb{G}_1$, $[1]_2 \in \mathbb{G}_2$ and random elements $y, x_1, \ldots, x_\ell \in \mathbb{Z}_p$ and define the master secret key MSK as

$$MSK = (y, x_1, \ldots, x_\ell, H) \in \mathbb{Z}_p^{\ell+1}.$$

Define $b_i(n)$ for positive integers i as the function that, on input of integer $n \geq 0$, outputs the i-th bit of the binary representation of n. Let $F(MSK, n)$ be the function that on input of $MSK = (x_1, \ldots, x_\ell)$ and an integer $n \geq 0$ outputs

$$F(MSK, n) = \prod_{i=1}^{\ell} x_i^{b_i(n)}.$$

The public parameters are defined as

$$PP = ([F(MSK,0)]_1, [F(MSK,1)]_1, \ldots, [F(MSK, 2^\ell - 1]_1, [1]_2, \nu, H),$$

where $\nu = e([1]_1, [1]_2)^y$.

Key Generation. The private key for identity id is computed as

$$USK_{id} = [y/u(id)]_2,$$

where

$$u(id) = \prod_{i=1}^{\ell} (H_{2^i}(id) + x_i) \in \mathbb{Z}_p. \tag{5}$$

Encapsulation. Observe that

$$u(id) = \prod_{i=1}^{\ell} (H_{2^i}(id) + x_i) = d_0 + \sum_{n=1}^{2k+1} \big(d_n \prod_{i=1}^{\ell} x_i^{b_i(n)}\big),$$

where the constants d_i are efficiently computable from $H(id)$.
To encapsulate a key, first $[u(id)]_1$ is computed. Note that this is possible from $H(id)$ and the values $F(MSK, n)$ contained in the public parameters (in particular, without knowing $x_1, \ldots, x_\ell$ explicitly), by computing

$$[u(id)]_1 = \left[d_0 + \sum_{n=1}^{2k+1} \big(d_n \prod_{i=1}^{\ell} x_i^{b_i(n)}\big)\right]_1 = [d_0]_1 \cdot \prod_{n=1}^{2k+1} [F(MSK, n)]_1^{d_n}.$$

Finally, the ciphertext and key are computed as

$$(C, K) = ([u(id)]_1^r, \nu^r) \in \mathbb{G}_T^2.$$

for uniformly random $r \xleftarrow{\$} \mathbb{Z}_p$.

Decapsulation. To recover K from a ciphertext C for identity id and a matching user secret key $[y/(u(id))]_2$, compute and output $e(C, USK_{id})$.

Correctness. The correctness follows from

$$e(C, USK_{id}) = e([u(id)]_1^r, [y/u(id)]_2) = e([1]_1, [y]_2)^r = \nu^r$$

Note that the scheme described above has extremely short ciphertexts of size only one element of $\mathbb{G}_1$, and also very efficient decapsulation, which takes only a single pairing evaluation. However, the public parameters have size $\mathcal{O}(k)$, and encapsulation is relatively expensive, as it costs $\mathcal{O}(k)$ exponentiations.

Again we consider the "nearly" black-box case first. The adoption to full security works then exactly as for Theorem 2.

Theorem 4. *Let $\mathcal{A}$ be an adversary that $(t_\mathcal{A}, q_\mathcal{A}, \epsilon_\mathcal{A})$-breaks the* IND-ID-CPA*-security of Π with $\epsilon_\mathcal{A} > 0$ $t_\mathcal{A}/\epsilon_\mathcal{A} < 2^k$. Given $\mathcal{A}$ and an index j satisfying (4), we can construct an adversary $\mathcal{B}_j$ that $(t_\mathcal{B}, q_\mathcal{B}, \epsilon_\mathcal{B})$-breaks the* IND-snaID-CPA*-security of the simplified BB ID-KEM $\Pi' = (\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Encap}', \mathsf{Decap}')$ with*

$$t_\mathcal{B} = \mathcal{O}(t_\mathcal{A}^4/\epsilon_\mathcal{A}^2), \quad q_\mathcal{B} < 4t_\mathcal{A}^4/\epsilon_\mathcal{A}^2 \quad \text{and} \quad \epsilon_\mathcal{B} \geq \frac{\epsilon_\mathcal{A}^3}{32t_\mathcal{A}^4}$$

Proof. The proof of Theorem 4 is almost identical to the proof of Theorem 1. The main difference is that we additionally use the algebraic structure of the underlying Boneh-Boyen ID-KEM to achieve short ciphertexts:

Setup and initial input. Just like in the proof of Theorem 1, $\mathcal{B}$ picks a random value $ID^* \xleftarrow{\$} \{0,1\}^{2^j}$ and requests a challenge ciphertext for identity ID^* and user secret keys for all $4t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2 - 1$ identities in the set $\{0,1\}^{2^j} \setminus \{ID*\}$. In response, $\mathcal{B}$ receives public parameters $PP' = ([1]_1, [x_j]_1, \nu)$ from the IND-snaID-CPA experiment, as well as user secret keys

$$[y/(ID + x_j)]_2$$

for all $ID \neq ID^*$ and a challenge ciphertext (C, K).

Additionally, $\mathcal{B}$ chooses $\ell - 1$ integers x_i for all $i \in \{1, \ldots, \ell\} \setminus \{j\}$.

Simulation of the public parameters. Note that $\mathcal{B}$ is not able to compute the function $F((x_1, \ldots, x_\ell), n) = \prod_{i=1}^{\ell} x_i^{b_i(n)}$ for all values of n efficiently, since it does not know x_j. However, $\mathcal{B}$ is able to efficiently compute

$$[F((x_1, \ldots, x_\ell), n)]_1 = \left[\prod_{i=1}^{\ell} x_i^{b_i(n)}\right]_1$$

for all values of n from $[x_j]_1$ and the x_i, $i \in \{1, \ldots, \ell\} \setminus \{j\}$. This is sufficient to properly simulate a public key of scheme Π.

Simulation of user secret keys. Using the user secret keys received from the IND-snaID-CPA challenger, $\mathcal{B}$ is able to answer all secret key queries for all identities id with $H_{2^j}(id) \neq ID^*$. To this end, it computes

$$USK_{id} = \left[y/\prod_{i=1}^{\ell}(H_{2^i}(id) + x_i)\right]_2 = [y/(H_{2^j}(id) + x_j)]_2^{1/(\prod_{i=1, i\neq j}^{\ell}(H_{2^i}(id)+x_i))}.$$

Creating the challenge ciphertext. $\mathcal{B}$ creates the challenge ciphertext as follows. If $\mathcal{A}$ has selected a target identity id^* with $H_{2^j}(id^*) = ID^*$, then $\mathcal{B}$ computes $C := C' \prod_{i=1, i\neq j}^{\ell}(H_{2^i}(id^*)+x_i)$ and outputs (C, K). Note that

$$C = [(H_{2^j}(id^*) + x_j)]^{r \prod_{i=1, i\neq j}^{\ell}(H_{2^i}(id^*)+x_i)} = \left[\prod_{i=1}^{\ell}(H_{2^i}(id^*) + x_i)\right]_1^r$$

such that C is a correctly distributed challenge ciphertext, and K is either "real" or "random", depending on the choice of the IND-snaID-CPA security experiment.

Analysis. The analysis of the success probability of $\mathcal{B}$ is identical to the analysis from the proof of Theorem 1, and yields identical bounds for Theorem 4. □

From "nearly" black-box to full black-box. This works exactly as in the proof of Theorem 2, without any significant modifications.

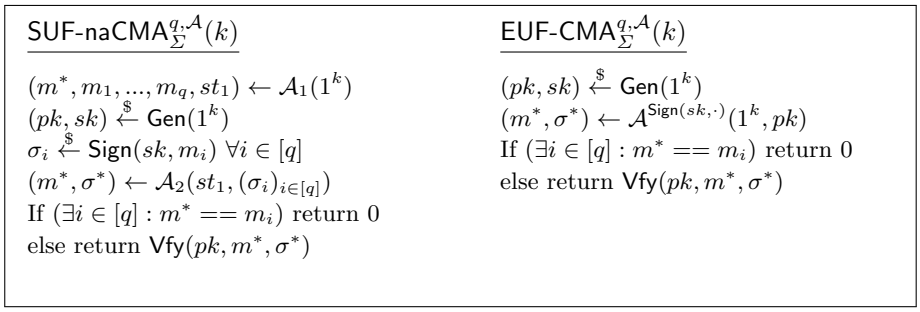

Fig. 2. The security experiments for digital signature schemes, executed with scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ and adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$. The oracle $\mathsf{Sign}(sk, m)$ returns $\sigma \xleftarrow{\$} \mathsf{Sign}(sk, m)$ with the restriction that $\mathcal{A}$ is not allowed to query oracle $\mathsf{Sign}(sk, m^*)$ for the challenge-message m^* and not more than a total of q queries.

4 Digital signatures

In this section, we show how continuous collision resistance can be applied to obtain a generic construction of adaptively-secure digital signatures from signatures with only very weak security, and a concrete number-theoretic construction of an adaptively-secure signature scheme over bilinear groups, where signatures consist of only a single group element.

4.1 Definitions and security notions

Definition 6. *A digital signature scheme consists of three PPT algorithms with the following syntax.*

$\mathsf{Gen}(1^k)$ *outputs a key pair* (pk, sk)*. We assume that* pk *implicitly defines a message space* $\mathcal{M}$*.*

$\mathsf{Sign}(sk, m)$ *on input of* sk *and message* $m \in \mathcal{M}$ *outputs a signature* σ*.*

$\mathsf{Vfy}(pk, m, \sigma)$ *outputs* 1 *if* σ *is a valid signature for* m *with respect to* pk *and else* 0*.*

Adaptive security. We recall the standard security notion *existential unforgeability under adaptive chosen message attack* (EUF-CMA) depicted in Figure 2. Note that the adversary may choose the challenge-message m^* after it has received the public key pk and may adaptively query signatures for messages $m_i \neq m^*$

Definition 7. *We say that adversary* $\mathcal{A}$ $(t_{\mathcal{A}}, q, \epsilon_{\mathcal{A}})$*- breaks the* EUF-CMA *security of* $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$*, if it runs in time* $t_{\mathcal{A}}$ *and*

$$\Pr[\mathsf{EUF\text{-}CMA}^{q,\mathcal{A}}_{\Sigma}(k) \Rightarrow 1] \geq \epsilon_{\mathcal{A}}.$$

and $t_{\mathcal{A}}$ *is the running time of* $\mathcal{A}$ including *the* EUF-CMA *security experiment.*

Selective and non-adaptive security. We also define a very weak security notion for digital signature schemes. Consider the SUF-naCMA security experiment depicted in Figure 2, where the attacker has to commit to both the challenge-message m^* the signing-query messages $m_1, \ldots, m_q$ non-adaptively and even before receiving the public key pk.

Definition 8. *We say that $\mathcal{A}$ $(t_{\mathcal{A}}, q, \epsilon_{\mathcal{A}})$-breaks the* SUF-naCMA *security of Σ, if it runs in time $t_{\mathcal{A}}$ and*

$$\Pr[\mathsf{SUF\text{-}naCMA}_{\Sigma}^{q,\mathcal{A}}(k) \Rightarrow 1] \geq \epsilon_{\mathcal{A}}.$$

4.2 From weak security to adaptive security

Construction. Let $\mathcal{H}$ be a family of continuously collision-resistant hash functions $H : \{0,1\}^* \to \{0,1\}^{4(k+1)}$ and $\Sigma' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vfy}')$ a SUF-naCMA secure digital signature scheme. In the sequel, let

$$\ell := \log 4(k+1)$$

We construct our EUF-CMA digital signature $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ as follows.

- **Key Generation.** Algorithm Gen samples $H \xleftarrow{\$} \mathcal{H}$ and computes $(pk_i, sk_i) \xleftarrow{\$} \mathsf{Gen}'(1^k)$ for all $i \in \{1, \ldots, \ell\}$, defines

  $$pk := (pk_1, \ldots, pk_\ell, H) \text{ and } sk = (sk_1, \ldots, sk_\ell, H)$$

  and outputs (pk, sk).
- **Signing.** To sign a message m, compute $\sigma_i \xleftarrow{\$} \mathsf{Sign}'(sk_i, H_{2^i}(m))$ for all $i \in \{1, \ldots, \ell\}$, and return the signature

  $$\sigma = (\sigma_1, \ldots, \sigma_\ell).$$

- **Verification.** To verify a signature $\sigma = (\sigma_1, \ldots, \sigma_\ell)$, compute and return

  $$\bigwedge_{i=1}^{\ell} \mathsf{Vfy}'(pk_i, H_{2^i}(m)) = 1$$

Again, we consider the "nearly" black-box case first, because it is contains the core idea behind the proof. The step to full black-box will then be much simpler than for the ID-KEM case, essentially because forging signatures is a "search" problem, while distinguishing ID-KEM ciphertexts is a "decisional" problem.

Theorem 5. *Given an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, q_{\mathcal{A}}\epsilon_{\mathcal{A}})$-breaks the* EUF-CMA*-security of $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ with $\epsilon_{\mathcal{A}} > 0$ and $t_{\mathcal{A}}/\epsilon_{\mathcal{A}} < 2^k$ and an index j that satisfies (4), we can construct an adversary $\mathcal{B}_j$ that $(t_{\mathcal{B}}, q_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-breaks the* SUF-naCMA*-security of $\Sigma' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vfy}')$ with*

$$t_{\mathcal{B}} = \mathcal{O}(t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2), \quad q_{\mathcal{B}} < 4t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2 \quad \text{and} \quad \epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}^3}{32t_{\mathcal{A}}^4}.$$

The proof of Theorem 5 is nearly identical to the proof of Theorem 1, except that some arguments and computing some bounds works *slightly* differently, because in the ID-KEM setting from Theorem 1 we are considering an "indistinguishability" security experiment, while in the digital signature setting of Theorem 5 we consider a "search problem". Therefore we give the full proof for completeness.

Proof. Again, before we are able to construct $\mathcal{B}$, we have to make a couple of modifications to the $\mathsf{EUF\text{-}CMA}$ security experiment. Consider the following sequence of games, where we denote with G_i the event that Game i outputs 1.

Game 0. This is the $\mathsf{EUF\text{-}CMA}_{\Sigma}^{q_s,\mathcal{A}}(k)$ security experiment. By definition, we have

$$\Pr[G_0] = \Pr\left[\mathsf{EUF\text{-}CMA}_{\Sigma}^{q_s,\mathcal{A}}(k) \Rightarrow 1\right].$$

Game 1. In this game, we assume that an index $j \in \{1, \ldots, \ell\}$ is given, such that (4) is satisfied. Furthermore, we define event coll_j, which occurs when the adversary $\mathcal{A}$ ever finds a collision for H_{2^j}. More precisely, coll_j occurs if $\mathcal{A}$ queries a signature for message m and forges a signature for message m^* such that $m \neq m^*$, but

$$H_{2^j}(m) = H_{2^j}(m^*).$$

If coll_j occurs, then Game 1 is aborted. With the same arguments as in Game 1 from the proof of Theorem 1, we have

$$\Pr[G_1] \geq \epsilon_{\mathcal{A}} - \Pr[\mathsf{coll}_j].$$

We use the continuous collision resistance of $\mathcal{H}$ to show $\Pr[\mathsf{coll}_j] \leq \epsilon/2$. Consider an algorithm $\mathcal{C}$, which runs the $\mathsf{EUF\text{-}CMA}$ security experiment with $\mathcal{A}$ and outputs the list of messages $(m^*, m_1, \ldots, m_q)$ for which $\mathcal{A}$ forges a message or requests a signature. The running time of $\mathcal{C}$ is at most $2 \cdot t_{\mathcal{A}}$, since $t_{\mathcal{A}}$ includes the running time of the security experiment (cf. Definition 7) and the only additional operation performed by $\mathcal{C}$ is to output the list of messages, which takes at most time $t_{\mathcal{A}}$. By construction, $\mathcal{C}$ finds a collision for H_{2^j} if and only if coll_j occurs. Continuous collision resistance guarantees that this probability is at most $2t_{\mathcal{A}}(2t_{\mathcal{A}} - 1)/2^{2^j+1}$. This yields

$$\Pr[\mathsf{coll}_j] \leq 2t_{\mathcal{A}}(2t_{\mathcal{A}} - 1)/2^{2^j+1} \leq 4t_{\mathcal{A}}^2/2^{2^j+1} < \epsilon/2,$$

where the last inequality applies Lemma 1 and the choice of j. Thus we have

$$\Pr[G_1] > \epsilon_{\mathcal{A}}/2.$$

Game 2. In Game 2, we additionally guess $M^* \xleftarrow{\$} \{0,1\}^{2^j}$ uniformly random, and raise event $\mathsf{abort}_{\mathsf{sign}}$ and abort, if adversary $\mathcal{A}$ forges a signature for message m^* with $H_{2^j}(m^*) \neq M^*$.

Since M^* is chosen uniformly random and independent of the view of the adversary, and $G_2 \wedge \neg\mathsf{abort}_{\mathsf{sign}} \iff G_1 \wedge \neg\mathsf{abort}_{\mathsf{sign}}$ we have

$$\begin{aligned}\Pr[G_2] &= \Pr[G_2 \mid \mathsf{abort}_{\mathsf{sign}}] \cdot \Pr[\mathsf{abort}_{\mathsf{sign}}] + \Pr[G_2 \wedge \neg\mathsf{abort}_{\mathsf{sign}}] \\ &= \Pr[G_2 \mid \mathsf{abort}_{\mathsf{sign}}] \cdot \Pr[\mathsf{abort}_{\mathsf{sign}}] + \Pr[G_1 \wedge \neg\mathsf{abort}_{\mathsf{sign}}] \\ &= \Pr[G_1] \cdot \Pr[\neg\mathsf{abort}_{\mathsf{sign}}] \\ &> \frac{\epsilon_{\mathcal{A}}}{2} \cdot \Pr[\neg\mathsf{abort}_{\mathsf{sign}}].\end{aligned}$$

M^* is chosen uniformly random from a set of size 2^{2^j}, so that the probability of guessing M^* correctly is $\Pr[\neg\mathsf{abort}_{\mathsf{sign}}] = 2^{-2^j}$. By applying Lemma 1, which guarantees that $2^{2^j} \leq \left(\frac{4t^2}{\epsilon}\right)^2$ for our choice of j, we get

$$\Pr[G_2] > \frac{\epsilon_{\mathcal{A}}}{2} \cdot \Pr[\neg\mathsf{abort}_{\mathsf{sign}}] = \frac{\epsilon_{\mathcal{A}}}{2} \cdot \frac{1}{2^{2^j}} \geq \frac{\epsilon_{\mathcal{A}}^3}{32 t_{\mathcal{A}}^4}.$$

Now we are ready to describe our reduction algorithm $\mathcal{B}_j$, which simulates Game 2 for $\mathcal{A}$.

Construction of $\mathcal{B}_j$. $\mathcal{B}_j$ samples a random bit string $M^* \xleftarrow{\$} \{0,1\}^{2^j}$ and defines $2^{2^j} - 1$ messages $M_1, \ldots, M_{2^{2^j}-1}$ that cover all values in $\{0,1\}^{2^j} \setminus \{M^*\}$. Then it outputs

$$(M^*, M_1, \ldots, M_{2^{2^j}-1}).$$

The $\mathsf{SUF\text{-}naCMA}$ experiment responds with a public key pk' and $2^{2^j} - 1$ signatures $\sigma_i' \leftarrow \mathsf{Sign}'(sk', M_i)$ for all $i \in \{1, \ldots, 2^{2^j} - 1\}$.

Simulation of the public key. In order to simulate a full public key of scheme Σ, $\mathcal{B}_j$ additionally computes $(pk_i, sk_i) \xleftarrow{\$} \mathsf{Gen}'(1^k)$ for all $i \in \{1, \ldots, \ell\} \setminus \{j\}$ and outputs

$$pk = (pk_1, ..., pk_{j-1}, pk', pk_{j+1}, ..., pk_\ell).$$

Simulation of signatures. Whenever $\mathcal{A}$ requests a signature for a message $m \in \{0,1\}^*$, $\mathcal{B}_j$ proceeds as follows. If $H_{2^j}(m) = M^*$, then $\mathcal{B}_j$ raises event $\mathsf{abort}_{\mathsf{sign}}$ and aborts. Otherwise, it proceeds as follows. $\mathcal{B}_j$ has requested signatures for all values $H_{2^j}(m) \in \{0,1\}^{2^j} \setminus \{M^*\}$. It generates $\sigma_i \xleftarrow{\$} \mathsf{Sign}'(sk_i, m)$ for all $i \in \{1, \ldots, \ell\} \setminus \{j\}$ and outputs

$$\sigma = (\sigma_1, ..., \sigma_{j-1}, \sigma', \sigma_{j+1}, ..., \sigma_\ell),$$

where σ' is the signature for $H_{2^j}(m)$ obtained from the $\mathsf{SUF\text{-}naCMA}$-experiment.

Extraction. If $\mathcal{A}$ outputs (m^*, σ^*), then $\mathcal{B}_j$ proceeds as follows. It first checks whether $H_{2^j}(m^*) = M^*$. If this does *not* hold, then $\mathcal{B}_j$ raises event $\mathsf{abort}_{\mathsf{sign}}$ and aborts. Otherwise it returns M^* and the j-th element σ_j of σ^*. Note that if σ^* is a valid forgery with respect to Σ, message m^* and public key pk, then σ_j is a valid forgery for Σ', message M^*, and public key pk'.

Success probability of $\mathcal{B}_j$. Note that B_j simulates Game 2 perfectly, thus we have

$$\Pr\left[\mathsf{SUF\text{-}naCMA}^{q,\mathcal{B}_j}_{\Sigma'}(k) \Rightarrow 1\right] = \Pr\left[G_2\right] > \frac{\epsilon^3_{\mathcal{A}}}{32t^4_{\mathcal{A}}}.$$

Running time of $\mathcal{B}_j$. The running time $t_{\mathcal{B}_j}$ of $\mathcal{B}_j$ consists of the time needed to execute $\mathcal{A}$, the time required to simulate the $\mathsf{EUF\text{-}CMA}$ security experiment, and the time required to request the $2^{2^j} - 1$ messages from the $\mathsf{SUF\text{-}naCMA}$ experiment, plus a minor number of additional operations. Making use of Lemma 1, we get

$$t_j \approx t_{\mathcal{A}} + \mathcal{O}\left(2^{2^j} - 1\right) \approx t_{\mathcal{A}} + \mathcal{O}\left(\frac{t^4_{\mathcal{A}}}{\epsilon^2_{\mathcal{A}}}\right) = \mathcal{O}\left(\frac{t^4_{\mathcal{A}}}{\epsilon^2_{\mathcal{A}}}\right).$$

Note also that $\mathcal{B}$ issues $q_{\mathcal{B}} = 2^{2^j} - 1 < 4t^4_{\mathcal{A}}/\epsilon^2_{\mathcal{A}}$ signing queries. This completes the proof. □

From "nearly" black-box to full black-box. Theorem 5 is only "nearly black-box", as it assumes that an index $j \in \{1, \ldots, \log 4(k+1)\}$ is given that satisfies $j = \left\lfloor \log\log 4t^2_{\mathcal{A}}/\epsilon_{\mathcal{A}} \right\rfloor + 1$ for adversary $\mathcal{A}$. We will show that we can also construct a reduction $\mathcal{B}$ that simply guesses this value j, and otherwise proceeds exactly like the algorithm $\mathcal{B}_j$ from the proof of Theorem 5. In the case of digital signatures this will be simpler than for ID-KEMs as considered above, because we will not have to argue that the view of $\mathcal{A}$ is independent of j. Reduction $\mathcal{B}$ simply guesses an index j uniformly random from $\{1, \ldots, \log 4(k+1)\}$ and then proceeds exactly like $\mathcal{B}_j$, hoping that j satisfies (4). If j is guessed correctly, which happens with probability at least $1/(\log 4(k+1))$, then Theorem 5 will provide a suitable lower bound on the success probability of the reduction. In contrast to the setting with a "distinguishing" problem considered for ID-KEMs, we do not have to worry that guessing j incorrectly may decrease the advantage. Therefore it seems also possible to lift the results from [40] from the "nearly black-box" setting to the fully black-box setting, if "search problems" like the problem of forging signatures are considered.

Again, the running time $t_{\mathcal{B}}$ of $\mathcal{B}$ depends exponentially on j, such that we can only get a reasonable bound on $t_{\mathcal{B}}$ if the guessed value of j approximates the running time and success probability of the adversary well-enough via the bounds from (4). Thus, again, strictly speaking, adversary $\mathcal{B}$ is not an algorithm that runs in strict or expected polynomial time in terms of classical complexity theory, but it is sufficient to establish that we can turn any efficient adversary $\mathcal{A}$ against Σ into an efficient adversary $\mathcal{B}$ against Σ', and thus to prove the

construction secure *without* requiring *a priori* knowledge about $\mathcal{A}$'s running time or success probability.

Theorem 6. *Given an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, q_{\mathcal{A}}\epsilon_{\mathcal{A}})$-breaks the* EUF-CMA*-security of $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ with $\epsilon_{\mathcal{A}} > 0$ and $t_{\mathcal{A}}/\epsilon_{\mathcal{A}} < 2^k$ and an index j that satisfies (4), we can construct an adversary $\mathcal{B}_j$ that $(t_{\mathcal{B}}, q_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-breaks the* SUF-naCMA*-security of $\Sigma' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vfy}')$ such that when $\mathcal{B}$ is executed, then with probability at least $1/\log 4(k+1)$ all the following bounds hold simultaneously:*

$$t_{\mathcal{B}} = \mathcal{O}(t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2), \quad q_{\mathcal{B}} < 4t_{\mathcal{A}}^4/\epsilon_{\mathcal{A}}^2 \quad \text{and} \quad \epsilon_{\mathcal{B}} \geq \frac{\epsilon_{\mathcal{A}}^3}{32t_{\mathcal{A}}^4}.$$

Thus, the total success probability of $\mathcal{B}$ is at least $\epsilon_{\mathcal{A}}^3/(32t_{\mathcal{A}}^4 \log 4(k+1))$.

Proof. Adversary $\mathcal{B}$ picks $j \xleftarrow{\$} \{1, ..., \log 4(k+1)\}$ uniformly random. Then it proceeds exactly like adversary $\mathcal{B}_j$ from the proof of Theorem 5.

Success probability of $\mathcal{B}$. To analyze the probability of $\mathcal{B}$, let corr denote the event that $\mathcal{B}$ guesses j correctly, that is, the value of j chosen by $\mathcal{B}$ satisfies (4) with respect to $\mathcal{A}$. Furthermore, write

$$X := \mathsf{SUF\text{-}naCMA}_{\Sigma'}^{q,\mathcal{B}}(k) \Rightarrow 1$$

and set $\ell := \log 4(k+1)$ to simplify our notation. Then we have

$$\Pr[X] \geq \Pr[X \cap \mathsf{corr}] = \Pr[X \mid \mathsf{corr}] \cdot \Pr[\mathsf{corr}] \geq \frac{\epsilon_{\mathcal{A}}^3}{32t_{\mathcal{A}}^4} \cdot \frac{1}{\ell},$$

where the last step follows from the bound on the success probability of $\mathcal{B}_j$ given by Theorem 6. Note that the first step in the above inequality is not as simple if a decisional problem is considered.

Running time of $\mathcal{B}$. Assuming that the guess of j was right in the sense of (4) we get the same running time like for $\mathcal{B}_j$ in Theorem (6). This happens with probability $1/\ell = 1/\log 4(k+1)$. □

4.3 Very short signatures with adaptive security

The generic construction of adaptively-secure digital signature schemes described in Section 4.2 increases the size of keys and signatures by a factor of $\mathcal{O}(\log(k))$. As for ID-KEMs it is possible to obtain a more efficient scheme based on specific, number-theoretic constructions. In this section we describe a variant of the Boneh-Boyen signature scheme [9] with a signature consisting of only a *single* group element and full adaptive security.

Building block: simplified Boneh-Boyen signatures. Again we use the "implicit notation" introduced by Escala *et al.* [23] to simplify our notation, see Section 3.3 for a definition.

Construction. The Boneh-Boyen signature scheme [9] consists of the following algorithms $\Sigma' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vfy}')$.

Key generation. Algorithm $\mathsf{Gen}'(k)$ chooses a random integer x and defines the keys as

$$pk := ([1]_1, [x]_1, [1]_2) \text{ and } sk := x.$$

Signing. Algorithm Sign' receives as input $sk = x$ and message $m \in \mathbb{Z}_p$, and computes and returns

$$\sigma := [1/(x+m)]_2 \in \mathbb{G}_2.$$

Verification. Algorithm Vfy' takes as input a public key

$$pk = ([1]_1, [x]_1, [1]_2) \in \mathbb{G}_1^2 \times \mathbb{G}_2,$$

message $m \in \mathbb{Z}_p$, and $\sigma \in \mathbb{G}_2$. It returns 1 if and only if

$$e\left([x]_1 \cdot [1]_1^m, \sigma\right) = e\left([1]_1, [1]_2\right).$$

Security. The original paper by Boneh and Boyen [9] proves security of this scheme in the sense of *existential unforgeability under non-adaptive chosen-message attacks* (EUF-naCMA), under the *strong* (or "flexible") q-Diffie-Hellman assumption. We will require only a weaker notion of security, in the sense of *selective unforgeability against non-adaptive chosen message attacks* (SUF-naCMA), which is achievable under a weaker, "non-flexible" q-type assumption.

Definition 9. *We say that adversary $\mathcal{A}$ $(\epsilon_{\mathcal{A}}, t_{\mathcal{A}})$-breaks the* q-Diffie-Hellman assumption *in group $\mathbb{G}$ of order p, if it runs in time $t_{\mathcal{A}}$ and*

$$\Pr\left[h \xleftarrow{\$} \mathcal{A}([1], [x], [x^2], \ldots, [x^q]) : h = [1/x]\right] \geq \epsilon_{\mathcal{A}},$$

where the probability is taken over the random coins of $\mathcal{A}$ and $x \xleftarrow{\$} \mathbb{Z}_p$.

The above assumption is also known as the q-Diffie-Hellman Inversion assumption [41]. By using the "generator-shifting" technique of Hofheinz *et al.* [29], one can prove the following theorem along the lines of the original proof of Boneh and Boyen [9].

Theorem 7. *From an adversary $\mathcal{A}$ that $(t_{\mathcal{A}}, q_s, \epsilon_{\mathcal{A}})$-breaks the* SUF-naCMA*-security of Σ' chosen-message queries, one can canstruct an adversary $\mathcal{B}$ that $(t_{\mathcal{B}}, \epsilon_{\mathcal{B}})$-breaks the q-Diffie-Hellman assumption with $q = q_s + 1$ and*

$$t_{\mathcal{B}} \approx t_{\mathcal{A}} \text{ and } \epsilon_{\mathcal{B}} = \epsilon_{\mathcal{A}}.$$

Encoding elements of $\{0,1\}^{4(k+1)}$ as $\mathbb{Z}_p$-elements. In the scheme described below, messages m are elements in $\mathbb{Z}_p$ for some prime p. In order to simplify the notation and description of the construction and its security analysis, we will henceforth make the implicit assumption that elements of $\{0,1\}^{4(k+1)}$ can be injectively encoded as elements in $\mathbb{Z}_p$ (see also the more detailed comment in Section 3.3).

Construction. Let $k \in \mathbb{N}$ be a security parameter, and let $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$ be groups of prime order p. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a non-degenerate, efficiently computable bilinear map. Let $\mathcal{H}$ be a family of continuously collision-resistant hash functions $H : \{0,1\}^* \to \{0,1\}^{4(k+1)}$ and $\ell = \log 4(k+1)$. We construct signature scheme $\Sigma = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vfy})$ as follows.

Key generation. Algorithm $\mathsf{Gen}(k)$ chooses $H \xleftarrow{\$} \mathcal{H}$ and ℓ random integers $x_1, \ldots, x_\ell \xleftarrow{\$} \mathbb{Z}_p$ and defines the secret key as

$$sk := (x_1, \ldots, x_\ell, H) \in \mathbb{Z}_p^\ell.$$

Note that sk contains only $\ell = \log 4(k+1)$ elements.
The public key is computed as follows. For a positive integer $i \geq 1$, let $b_i(n)$ be the function that, on input of integer $n \geq 0$, outputs the i-th bit of the (canonical) binary representation of n. Let $F(sk, n)$ be the function that, on input of $sk = (x_1, \ldots, x_\ell)$ and integer $n \geq 0$, outputs

$$F(sk, n) := \prod_{i=1}^{\ell} x_i^{b_i(n)}.$$

The public key is defined as

$$pk := ([F(sk, 0)]_1, \ldots, [F(sk, 2^\ell - 1)]_1, [1]_2, H).$$

It contains the group element $[\prod_{x \in X} x]_1$ for each possible subset $X \subseteq \{x_1, \ldots, x_\ell\}$.

Signing. Algorithm Sign receives as input $sk = (x_1, \ldots, x_\ell)$ and message $m \in \{0,1\}^*$. Let $u(m)$ be the function

$$u(m) := \prod_{i=1}^{\ell} (x_i + H_{2^i}(m)) \in \mathbb{Z}_p, \tag{6}$$

where bit strings $H_{2^i}(m)$ are interpreted canonically as integers in $\mathbb{Z}_p$. Recall here that by our assumption on p this is injective for all $i \in \{1, \ldots, \ell\}$.
The signing algorithm computes and returns

$$\sigma := [1/u(m)]_2 \in \mathbb{G}_1.$$

Note that computing signatures is *extremely* efficient. It involves only the computation of $1/u(m) \in \mathbb{Z}_p$, which can be performed over the integers modulo p, where p is the group order, and then a *single* exponentiation in $\mathbb{G}_1$ to compute $g_1^{1/u(m)} \in \mathbb{G}_1$.

Verification. Algorithm Vfy takes as input a public key

$$pk = ([F(sk, 0)]_1, \ldots, [F(sk, 2^\ell - 1)]_1, [1]_2),$$

message $m \in \{0,1\}^*$, and $\sigma \in \mathbb{G}_2$. Note here that $[F(sk, 0)]_1 = [1]_1$. The algorithm returns 1 if and only if

$$e\left([u(m)]_1, \sigma\right) = e\left([1]_1, [1]_2\right). \tag{7}$$

Here $[u(m)]_1$ is computed as follows. Viewing $u(m) = \prod_{i=1}^{\ell}(x_i + H_{2^i}(m))$ as a polynomial in ℓ unknowns $x_1, \ldots, x_\ell$, we can expand the product from (6) to obtain the equation

$$u(m) = \prod_{i=1}^{\ell}(x_i + H_{2^i}(m)) = d_0 + \sum_{n=0}^{2^\ell - 1}\left(d_n \prod_{i=1}^{\ell} x_i^{b_i(n)}\right) \tag{8}$$

for integers d_i which are efficiently computable from $H(m)$. This yields the equation

$$\begin{aligned}[u(m)]_1 &= \left[d_0 + \sum_{n=0}^{2^\ell - 1}\left(d_n \prod_{i=1}^{\ell} x_i^{b_i(n)}\right)\right]_2 \\ &= [d_0]_2 \cdot \prod_{n=0}^{2^\ell - 1} [F(sk, n)]_2^{d_n}. \end{aligned} \tag{9}$$

Therefore the verification algorithms proceeds as follows:

1. From $H(m)$ it computes the integers d_i as in (8) .
2. Then it computes $[u(m)]_1$ as in (9) from the group elements $[F(sk, n)]_1$ contained in the public key.
3. Finally, it outputs 1 if and only if Equation (7) holds.

Theorem 8. *Let $k \in \mathbb{N}$ be a security parameter. Given an adversary $\mathcal{A}$ that $(t_\mathcal{A}, q_\mathcal{A}, \epsilon_\mathcal{A})$-breaks the* EUF-CMA*-security of Σ with $t_\mathcal{A}/\epsilon_\mathcal{A} < 2^k$ and an integer j that satisfies (4), we can construct an adversary $\mathcal{B}$ that $(t_\mathcal{B}, q, \mathcal{B}, \epsilon_\mathcal{B})$-breaks the* SUF-naCMA *security of the Boneh-Boyen signature scheme $\Sigma' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vfy}')$ with*

$$t_\mathcal{B} = \mathcal{O}(t_\mathcal{A}^4/\epsilon_\mathcal{A}^2), \qquad q_\mathcal{B} < 4t_\mathcal{A}^4/\epsilon_\mathcal{A}^2 \qquad \text{and} \qquad \epsilon_\mathcal{B} \geq \frac{\epsilon_\mathcal{A}^3}{32t_\mathcal{A}^4}.$$

The proof of Theorem 8 is almost identical to the proofs of Theorems 4 and 5, therefore we only sketch it.

Proof sketch. Like in the proof of Theorem 4, we will use the algebraic structure of the underlying Boneh-Boyen signature scheme to achieve short signatures.

Setup and initial input. Like in the proof of Theorem 5, $\mathcal{B}$ picks a random value $M^* \xleftarrow{\$} \{0,1\}^{2^j}$ and submits it to the SUF-naCMA experiment. Furthermore it queries signatures for all $4t_\mathcal{A}^4/\epsilon_\mathcal{A}^2 - 1$ messages in the set $\{0,1\}^{2^j} \setminus \{M^*\}$. In response, $\mathcal{B}$ receives the public key $pk' = ([1]_1, [x_j]_1, [1]_2)$ from the SUF-naCMA experiment, as well as signatures

$$[1/(x_j + M)]_2$$

for all $M \neq M^*$.

Additionally, $\mathcal{B}$ chooses $\ell - 1$ integers x_i for all $i \in \{1, \ldots, \ell\} \setminus \{j\}$.

Simulation of the public parameters. Note that $\mathcal{B}$ is not able to compute the function $F((x_1,\ldots,x_\ell),n) = \prod_{i=1}^{\ell} x_i^{b_i(n)}$ for all values of n efficiently, since it does not know x_j. However, $\mathcal{B}$ is able to efficiently compute

$$[F((x_1,\ldots,x_\ell),n)]_1 = \left[\prod_{i=1}^{\ell} x_i^{b_i(n)}\right]_1$$

for all values of n from $[1]_1$,$[x_j]_1$ and the x_i, $i \in \{1,\ldots,\ell\}\setminus\{j\}$. This is sufficient to properly simulate a public key of scheme Σ.

Simulation of signatures. Using the signatures received from the SUF-naCMA challenger, $\mathcal{B}$ is able to answer all signature queries for all messages m with $H_{2^j}(m) \neq M^*$. To this end, it computes

$$\sigma_m = \left[1/\prod_{i=1}^{\ell}(H_{2^i}(m)+x_i)\right]_2 = [1/(H_{2^j}(m)+x_j)]_2^{\prod_{i=1,i\neq j}^{\ell}(H_{2^i}(m)+x_i)}.$$

Extraction. If $\mathcal{A}$ outputs (m^*,σ^*), then $\mathcal{B}_j$ proceeds as follows. If $\mathcal{A}$ has selected a message m^* with $H_{2^j}(m^*) = M^*$, then $\mathcal{B}$ computes and outputs

$$\sigma' = (\sigma^*)^{\prod_{i=1,i\neq j}^{\ell}(x_i+H_{2^i}(m^*))}.$$

Note that this is efficiently computable by $\mathcal{B}$, because it "knows" x_i for all $i \neq j$. Note also that if σ^* is a valid forgery with respect to Σ, message m^*, and public key pk, then we have

$$\begin{aligned}
\sigma' &= (\sigma^*)^{\prod_{i=1,i\neq j}^{\ell}(x_i+H_{2^i}(m^*))} \\
&= [1/u(m^*)]_2^{\prod_{i=1,i\neq j}^{\ell}(x_i+H_{2^i}(m^*))} \\
&= [1/\prod_{i=1}^{\ell}(x_i+H_{2^i}(m^*))]_2^{\prod_{i=1,i\neq j}^{\ell}(x_i+H_{2^i}(m^*))} \\
&= [1/(x_j+H_{2^j}(m^*))]_2 = [1/(x_j+M^*)]_2.
\end{aligned}$$

Thus, σ' is a valid forgery for the Boneh-Boyen scheme Σ', message M^*, and public key pk'.

Analysis. The analysis of the success probability of $\mathcal{B}$ is identical to the analysis from the proof of Theorem 5, and yields identical bounds.

References

1. Christoph Bader, Tibor Jager, Yong Li, and Sven Schäge. On the impossibility of tight cryptographic reductions. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 273–304. Springer, Heidelberg, May 2016.

2. Mihir Bellare, Viet Tung Hoang, and Sriram Keelveedhi. Instantiating random oracles via UCEs. In Canetti and Garay [19], pages 398–415.
3. Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for Waters' IBE scheme. In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *LNCS*, pages 407–424. Springer, Heidelberg, April 2009.
4. Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93*, pages 62–73. ACM Press, November 1993.
5. Mihir Bellare, Igors Stepanovs, and Stefano Tessaro. Contention in cryptoland: Obfuscation, leakage and UCE. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part II*, volume 9563 of *LNCS*, pages 542–564. Springer, Heidelberg, January 2016.
6. Florian Böhl, Dennis Hofheinz, Tibor Jager, Jessica Koch, Jae Hong Seo, and Christoph Striecks. Practical signatures from standard assumptions. In Johansson and Nguyen [33], pages 461–485.
7. Dan Boneh and Xavier Boyen. Efficient selective-ID secure identity based encryption without random oracles. In Cachin and Camenisch [18], pages 223–238.
8. Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Matthew Franklin, editor, *CRYPTO 2004*, volume 3152 of *LNCS*, pages 443–459. Springer, Heidelberg, August 2004.
9. Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Cachin and Camenisch [18], pages 56–73.
10. Dan Boneh and Xavier Boyen. Efficient selective identity-based encryption without random oracles. *Journal of Cryptology*, 24(4):659–693, October 2011.
11. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 213–229. Springer, Heidelberg, August 2001.
12. Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT 2003*, volume 2656 of *LNCS*, pages 416–432. Springer, Heidelberg, May 2003.
13. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. In Colin Boyd, editor, *ASIACRYPT 2001*, volume 2248 of *LNCS*, pages 514–532. Springer, Heidelberg, December 2001.
14. Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, September 2004.
15. Dan Boneh, Emily Shen, and Brent Waters. Strongly unforgeable signatures based on computational Diffie-Hellman. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *PKC 2006*, volume 3958 of *LNCS*, pages 229–240. Springer, Heidelberg, April 2006.
16. Zvika Brakerski and Yael Tauman Kalai. A framework for efficient signatures, ring signatures and identity based encryption in the standard model. Cryptology ePrint Archive, Report 2010/086, 2010. `http://eprint.iacr.org/2010/086`.
17. Christina Brzuska, Pooya Farshim, and Arno Mittelbach. Indistinguishability obfuscation and UCEs: The case of computationally unpredictable sources. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 188–205. Springer, Heidelberg, August 2014.
18. Christian Cachin and Jan Camenisch, editors. *EUROCRYPT 2004*, volume 3027 of *LNCS*. Springer, Heidelberg, May 2004.

19. Ran Canetti and Juan A. Garay, editors. *CRYPTO 2013, Part II*, volume 8043 of *LNCS*. Springer, Heidelberg, August 2013.
20. Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. ACM Press, May 1998.
21. Dario Catalano, Dario Fiore, and Luca Nizzardo. Programmable hash functions go private: Constructions and applications to (homomorphic) signatures with shorter public keys. In Rosario Gennaro and Matthew J. B. Robshaw, editors, *CRYPTO 2015, Part II*, volume 9216 of *LNCS*, pages 254–274. Springer, Heidelberg, August 2015.
22. Jean-Sébastien Coron. Optimal security proofs for PSS and other signature schemes. In Lars R. Knudsen, editor, *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 272–287. Springer, Heidelberg, April / May 2002.
23. Alex Escala, Gottfried Herold, Eike Kiltz, Carla Ràfols, and Jorge Villar. An algebraic framework for Diffie-Hellman assumptions. In Canetti and Garay [19], pages 129–147.
24. Shimon Even, Oded Goldreich, and Silvio Micali. On-line/off-line digital signatures. *Journal of Cryptology*, 9(1):35–67, 1996.
25. Marc Fischlin and Nils Fleischhacker. Limitations of the meta-reduction technique: The case of Schnorr signatures. In Johansson and Nguyen [33], pages 444–460.
26. Marc Fischlin, Anja Lehmann, Thomas Ristenpart, Thomas Shrimpton, Martijn Stam, and Stefano Tessaro. Random oracles with(out) programmability. In Masayuki Abe, editor, *ASIACRYPT 2010*, volume 6477 of *LNCS*, pages 303–320. Springer, Heidelberg, December 2010.
27. Eduarda S. V. Freire, Dennis Hofheinz, Kenneth G. Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 513–530. Springer, Heidelberg, August 2013.
28. Goichiro Hanaoka, Takahiro Matsuda, and Jacob C. N. Schuldt. On the impossibility of constructing efficient key encapsulation and programmable hash functions in prime order groups. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO 2012*, volume 7417 of *LNCS*, pages 812–831. Springer, Heidelberg, August 2012.
29. Dennis Hofheinz, Tibor Jager, and Eike Kiltz. Short signatures from weaker assumptions. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 647–666. Springer, Heidelberg, December 2011.
30. Dennis Hofheinz, Tibor Jager, and Edward Knapp. Waters signatures with optimal security reduction. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 66–83. Springer, Heidelberg, May 2012.
31. Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In David Wagner, editor, *CRYPTO 2008*, volume 5157 of *LNCS*, pages 21–38. Springer, Heidelberg, August 2008.
32. Susan Hohenberger and Brent Waters. Short and stateless signatures from the RSA assumption. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 654–670. Springer, Heidelberg, August 2009.
33. Thomas Johansson and Phong Q. Nguyen, editors. *EUROCRYPT 2013*, volume 7881 of *LNCS*. Springer, Heidelberg, May 2013.
34. Saqib A. Kakvi and Eike Kiltz. Optimal security proofs for full domain hash, revisited. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 537–553. Springer, Heidelberg, April 2012.

35. Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS 2000*. The Internet Society, February 2000.
36. Steve Lu, Rafail Ostrovsky, Amit Sahai, Hovav Shacham, and Brent Waters. Sequential aggregate signatures and multisignatures without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *LNCS*, pages 465–485. Springer, Heidelberg, May / June 2006.
37. Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, 2004. `http://eprint.iacr.org/2004/332`.
38. Ron Steinfeld, Josef Pieprzyk, and Huaxiong Wang. How to strengthen any weakly unforgeable signature into a strongly unforgeable signature. In Masayuki Abe, editor, *CT-RSA 2007*, volume 4377 of *LNCS*, pages 357–371. Springer, Heidelberg, February 2007.
39. Brent R. Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 114–127. Springer, Heidelberg, May 2005.
40. Mark Zhandry. The magic of ELFs. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part I*, volume 9814 of *LNCS*, pages 479–508. Springer, Heidelberg, August 2016.
41. Fangguo Zhang, Reihaneh Safavi-Naini, and Willy Susilo. An efficient signature scheme from bilinear pairings and its applications. In Feng Bao, Robert Deng, and Jianying Zhou, editors, *PKC 2004*, volume 2947 of *LNCS*, pages 277–290. Springer, Heidelberg, March 2004.