A Framework for Efficient Signatures, Ring Signatures and Identity Based Encryption in the Standard Model

Zvika Brakerski* Yael Tauman Kalai†

February 13, 2010

Abstract

In this work, we present a generic framework for constructing efficient signature scheme, ring signature schemes, and identity based encryption schemes, all in the standard model (without relying on random oracles).

We start by abstracting the recent work of Hohenberger and Waters (Crypto 2009), and specifically their "prefix method". We show a transformation taking a signature scheme with a very weak security guarantee (a notion that we call a-priori-message unforgeability under static chosen message attack) and producing a fully secure signature scheme (i.e., existentially unforgeable under adaptive chosen message attack). Our transformation uses the notion of chameleon hash functions, defined by Krawczyk and Rabin (NDSS 2000) and the "prefix method". Constructing such weakly secure schemes seems to be significantly easier than constructing fully secure ones, and we present *simple* constructions based on the RSA assumption, the *short integer solution* (SIS) assumption, and the *computational Diffie-Hellman* (CDH) assumption over bilinear groups.

Next, we observe that this general transformation also applies to the regime of ring signatures. Using this observation, we construct new (provably secure) ring signature schemes: one is based on the *short integer solution* (SIS) assumption, and the other is based on the CDH assumption over bilinear groups. As a building block for these constructions, we define a primitive that we call *ring trapdoor functions*. We show that ring trapdoor functions imply ring signatures under a weak definition, which enables us to apply our transformation to achieve full security.

Finally, we show a connection between ring signatures and identity based encryption (IBE) schemes. Using this connection, and using our new constructions of ring signature schemes, we obtain two IBE schemes: The first is based on the *learning with error* (LWE) assumption, and is similar to the recently introduced IBE schemes of Peikert, Agrawal-Boyen and Cash-Hofheinz-Kiltz (2009); The second is based on the d-linear assumption over bilinear groups.

*Weizmann Institute of Science and Microsoft Research, zvika.brakerski@weizmann.ac.il.
†Microsoft Research, yael@microsoft.com.

1 Introduction

Digital signature schemes are one of the most fundamental cryptographic notions. It is well known that signature schemes that adhere to very strong security guarantees (that are formally defined in [GMR88]) can be constructed under the necessary assumption that one-way functions exist [NY89, Rom90]. However, the resulting scheme is highly inefficient, and does not suffice for practical purposes.

Efficient signature schemes. There has been a major effort to try and construct efficient signature schemes, even based on stronger primitives than one-way functions, such as collision-resistant hash functions or trapdoor permutations, or even based on specific number theoretic assumptions. This task, however, appears to be surprisingly hard.

One successful line of work was in the *random oracle model*, where many very efficient and simple schemes were constructed [FS86, Gam84, Sch91, Oka92, BR93, PS96, BLS04, GJKW07, GPV08]. However, in the standard model, constructing efficient schemes seems to be significantly harder. Indeed, until recently, all such schemes were either very complicated or relied on relatively strong assumptions such as strong-RSA or assumptions in bilinear groups [GHR99, CS00, BB04c, CL04, Wat05]. Very recently, efficient schemes were constructed and proven secure under standard assumptions: Hohenberger and Waters [HW09a, HW09b] constructed a signature scheme based on the standard RSA assumption; and Peikert [Pei09], Agrawal and Boyen [AB09] and Cash Hofheinz and Kiltz [CHK09], constructed similar schemes based on the lattice-related *short integer solution* (SIS) assumption (see Section 6.1 and [MR07, GPV08] for details on this assumption), which are, while based on standard assumptions, still quite complicated.

Ring Signatures. The notion of ring signatures was introduced in [RST01]: A ring signature scheme is a signature scheme with the property that a user can specify any set of possible signers that includes itself, and sign without revealing which member actually produced the signature. Ring signatures provide an elegant way to leak authoritative secrets in an anonymous way. Unlike the related notion of group signatures (see [CvH91]), ring signatures have no group managers, no setup procedures, no revocation procedures, and no coordination: any user can choose any set of possible signers that includes himself, and sign any message by using his secret key and the others public keys, without getting their approval or assistance. Many implementations of ring signatures having various properties were proposed in the literature (see [RST06, Section 7] for discussion), however their security was proven only in the random oracle model or under nonstandard assumptions. To the best of our knowledge, the question of constructing ring signature schemes that are secure in the standard model and under standard assumptions was not answered by previous works.

Identity based encryption. An identity-based encryption (IBE) scheme, a notion introduced by Shamir [Sha84], is an encryption scheme where the public key of each user is simply their identity. Each IBE scheme is associated with a pair (pp, msk) issued by a trusted authority, where pp are public parameters (sometimes referred to as the "master public key") and msk is the master secret key. Each user, uses their identity id as a public key, and obtains a corresponding secret key sk_{id} from the trusted authority. Each secret key sk_{id}, corresponding to identity id is a (possibly randomized) function of the identity id and the master secret key msk. An encryption of a message m, corresponding to identity id, is a randomized function of the message m, the identity id and the public parameters pp. The decryption of a ciphertext c corresponding to identity id, is a (possibly randomized) function of the ciphertext c, the identity id, and the secret key sk_{id} corresponding to the identity id.

Constructing secure IBE schemes based on standard assumptions, without using random oracles (or "interactive assumptions"), even under weak notions of security, has been a long standing open problem. Secure schemes were finally introduced based on assumptions in groups with bilinear maps, starting with the works of [CHK07, BB04a]. Recently, several schemes, which are quite similar, were introduced based on lattice assumptions [AB09, CHK09, Pei09].

1.1 Our Results

We present a formal connection between strong and weak notions of security for digital signature schemes, by abstracting the recent work of Hohenberger and Waters [HW09b]. We use these ideas to show similar connections in the regime of ring signatures, which enable us to construct ring signatures based on a new generic primitive: ring trapdoor functions, which can be constructed based on the SIS assumption or the CDH assumption over bilinear groups. We go on to show a connection between ring signatures and identity based encryption schemes, and finally present constructions of identity based encryption schemes based on standard lattice assumptions and on hardness assumptions in bilinear groups. More details follow.

Signature schemes. We reduce the task of constructing signature schemes that are existentially unforgeable under adaptive chosen message attack (e-cma), into the one of constructing signature schemes under a (seemingly) much weaker security notion, that we call *a-priori-message unforgeability under static chosen message attack* (a-scma). While in e-cma-security, the forger makes adaptive queries after seeing the verification key, and successfully forges if it can come up with an accepting signature for *any* new message; in a-scma-security, the forger must produce a signature for a random message (sampled by the challenger). Its queries may depend on this random message, but not on the verification key, and they are not adaptive.

Our reduction uses, as a first step, the reduction of [KR00] between adaptive and static chosen message attacks and, as a second step, it reduces existential to a-priori-message unforgeability, abstracting the ideas of [HW09b]. Our abstraction comes at a cost of a linear factor loss in the signature length. The computational complexity of signing and verifying also increases by the same factor, but this increase is completely parallelizeable.

We exemplify the simplicity of constructing secure signature schemes (in the standard model) using this methodology: we provide an explicit example based on the RSA assumption; additional examples, based on lattice assumptions and on assumptions in bilinear groups, follow from our constructions of ring signatures.

Ring signature schemes. We show that a very similar reduction to the one presented for signature schemes, also holds in the regime of ring signature schemes. Namely, we show that it is sufficient to construct ring signature schemes that are *a-priori-message ring unforgeable under static chosen message attack* (ar-scma), which is defined slightly differently from the standard signature variant: while the forger receives a random challenge message, and needs to specify the messages it wants to query on at the beginning of the experiment, it may still be adaptive in the selection of the set of users with respect to which the messages are signed. This makes the definition more complicated, but essentially the same tools are used in the reduction. The reduction in this case requires a slight variant of chameleon hash, which we demonstrate how to achieve in a very similar manner to the known chameleon hash constructions.

We then show that ar-scma-security is implied by ring trapdoor functions (see below). Our constructions of ring trapdoor functions under the SIS assumption and under CDH in bilinear groups (see below for details), therefore, imply ring signatures under these assumptions (we also show that they imply the variant of chameleon hash we require).

Identity based encryption. We consider two security notions for IBE (many other notions exist, which we do not discuss in this work). An IBE scheme is said to be *adaptively-secure* if an adversary, who is given properly generated public parameters, cannot break semantic security corresponding to any identity id^* of his choice, even after seeing the secret keys of an adaptively chosen set of identities (so long as id^* is not in this set). The weaker notion of *selective-security* requires the adversary to "declare" the value of id^* before seeing the public-parameters of the scheme.

It is well known that IBE schemes immediately yield secure signature schemes as follows: The generation of the verification key and the signing key of the signature scheme is identical to the generation of the public parameters and the master secret key. Namely, the verification key is pp and the signing key is msk. To sign a message id, compute a secret key sk_{id} corresponding to id. To verify, encrypt random messages using pp and id and see that they decrypt correctly using the alleged signature as secret-key.[1]

The converse is not necessarily true. Moreover, even if the signature scheme $\mathcal{S} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ has the special property that for any message id the pair $(id, \mathsf{Sign}(id))$ can be used as a public and a secret key pair for a public key encryption scheme, still it is not clear that $\mathcal{S}$ can be used to construct a secure IBE scheme. The naive attempt would be to construct an IBE scheme by generating public parameters and a master secret key using Gen; namely, $pp = vk$ and $msk = sk$. Then a user with identity id will be assigned a secret key $\mathsf{Sign}(id)$, and will use the pair $(id, \mathsf{Sign}(id))$ as public and secret keys for a public key encryption scheme. This attempt may at first seem promising. However, taking a closer look, one can see that the resulting IBE scheme will not necessarily be secure. The reason is that getting many secret keys corresponding to identities $id_1, \ldots, id_\ell$ does not allow one to compute a secret key for a new identity, but may still give enough information for breaking semantic security.

In contrast, we show that ring signature schemes that have a special property, similar to the one described above, can actually be used to construct selectively-secure IBE schemes. We refer to these as *encryption augmented ring signatures*. Furthermore, known techniques [BB04a, BB04b] for converting selectively-secure to adaptively-secure IBE, are applicable to IBE schemes constructed from encryption-augmented ring signatures (the technique of [BB04b] requires, in addition, a family of collision resistent hash functions).

Ring trapdoor functions. We present a new primitive called ring trapdoor functions and show that it can be used to produce ring signatures. The notion of ring trapdoor functions is a generalization of the standard notion of trapdoor functions. It is not only required that given f, y it is hard to find x such that $f(x) = y$, but it is also hard, given $f_1, \ldots, f_t, y$ to find $x_1, \ldots, x_t$ such that $\sum_{i=1}^{t} f_i(x_i) = y$, for any polynomial t. However, given a trapdoor for *any* of the functions f_i, one can efficiently generate such $x_1, \ldots, x_t$, and furthermore, it is impossible to tell, looking at $x_1, \ldots, x_t$, which of the t trapdoors was used generate them.

In addition, we relax the standard definition of trapdoor functions and only require that given f, x, y, one can efficiently verify that $f(x) = y$, we do not require that f is efficiently computable.

Instantiations under cryptographic assumptions. We show that ring trapdoor functions can be constructed under the SIS assumption (in fact, we only require the weaker inhomogenous

[1] A selectively-secure IBE implies a signature scheme that is secure against selective forgery and an adaptively-secure IBE implies a scheme that is secure against existential forgery.

variant, called ISIS). The ring signature scheme obtained from this family is shown to be encryption augmented under the LWE assumption, yielding a secure IBE scheme.

We further show a construction of ring signature schemes under the CDH assumption in bilinear groups. The construction carries a very similar structure to the SIS construction mentioned above.[2] We show that under the d-linear assumption, the resulting ring signature scheme can be encryption-augmented, thus obtaining an IBE encryption scheme under the d-linear assumption, for any $d \geq 2$. To the best of our knowledge, this is the first IBE scheme based solely on the d-linear assumption (the scheme of Waters [Wat09] relies on a combination of the 2-linear and the bilinear Diffie-Hellman assumptions).

1.2 Our Techniques

Let us first explain our reduction for standard signatures. We use a result of Krawczyk and Rabin [KR00] to reduce the adaptive notion of security, in which the forger makes adaptive queries, into a static one where the forger specifies all of its queries at the beginning of the experiment. We stress that in both cases, the forgery is *existential*, i.e. a successful forgery is a valid signature for any message of the forger's choosing, so long as this message was not queried on. This reduction uses a primitive called *chameleon hash functions* (see Section 2.1).

Our next reduction is from existential forgery, where the forger needs to forge a signature on a message of its choice, to a-priori-message forgery, where the forger is given a random message to forge on. We do this by abstracting the "prefix method" of [HW09b]. We sign a message μ of length m by considering all m prefixes of μ, applying a universal hash function to each of them, and signing the results separately. The new signature for μ is, thus, composed of m signatures of the m prefixes. Intuitively, the reason why this should be secure is that when the forger (for the existential attack) specifies the messages he wishes to get signatures for (recall that this is done before seeing the public-key), it actually "commits" to forging a signature for one of polynomially many messages. This is because the message it forges the signature of, must contain a prefix that only differs in the last bit from one of the prefixes of one of the messages it specified, and there are only polynomially many options for that. Therefore, the message that it forges a signature of is determined before the key generation, and thus is "a-priori" in some weak sense. The universal hash function (which is just XORing with a random string) is required so that all of these messages "look random", i.e. be distributed as the a-priori challenge. We also need to add a "tag" to each prefix we sign, in order to avoid "cut and paste" attacks between prefixes.

The ring signature variants of the aforementioned reductions are quite similar, with the exception of a minor technical difficulty resulting from the fact that "trusted" public parameters are not allowed in this setting (this also requires us to define a variant of the chameleon hash functions primitive mentioned above).

We are left with the task of constructing ring signature schemes, that need to be secure against a-priori-message forgery under static chosen message attack, from ring trapdoor functions. We recall that finding a pre-image for a set of ring trapdoor functions can be done, by definition, using a trapdoor for *any* of the functions in the set. We can, therefore, construct a ring signature scheme with a degenerate message space containing just one message. This is done by setting the verification key to be the description of the function and the signing key to be the trapdoor. To sign

[2] Our proof works by showing that it is CDH-hard to solve a set of random linear equations in the exponent. To the best of our knowledge, our proof technique, though simple, is novel and may be useful outside the scope of this paper.

the message using a set of verification keys (which are function descriptions), we use the signing key, which is a trapdoor for one of the functions in the set, to sample a pre-image respective to these functions (there is a minor technical issue of for which output we find a pre-image, but this can be resolved).

We extend this idea to make the signature also depend on the message (and not only on the subset of verification keys) by sampling $2m$ such functions for each user, where m is the message size (in fact we use $2m+1$ functions for a reason related to the complexity of the scheme and key lengths). Each message defines a subset of m functions for each user (each bit of the message selects one of two functions). We then sign with respect to the combined set of m times the number of users functions.

Our identity based encryption construction from encryption-augmented ring signatures is an abstraction of an idea used in many previous schemes, where the public-parameters contains $2d$ "challenges", where d is the bit-length of identities (or rather $2d+1$, which can improve efficiency), and each identity specifies a cardinality-d subset of challenges. These challenges are then "combined" to create the public-key for the encryption. We note the resemblance between this method and our construction of ring signatures described above. We demonstrate that verification keys of a ring signature scheme qualify as such "challenges", provided that there exists an encryption scheme whose public-keys are associated with a set of verification keys, and whose secret keys are associated with ring signatures for the public-key - which is exactly the definition of encryption-augmented ring signatures.

Once the above reductions and definitions are obtained, our lattice-based instantiation is essentially casting the known ideas on lattices (see e.g. [GPV08, AB09, Pei09, CHK09]) into our new framework. Essentially a ring trapdoor function is a multiplication of a matrix $\mathbf{A}$ (which is the description of the function) and a short vector $\mathbf{x}$ (the input).

For our bilinear based constructions, we consider matrix-vector multiplication "in the exponent", i.e. taking $g^{\mathbf{A}}$ and $g^{\mathbf{x}}$ (the vector $\mathbf{x}$ needs not be short here) and outputting $g^{\mathbf{Ax}}$, where g is a generator of the group. While this function is hard to compute, it is easy to verify given a bilinear map on our group. We show that it is CDH-hard to find $g^{\mathbf{x}}$ such that $g^{\mathbf{Ax}} = g^{\mathbf{y}}$, given $g^{\mathbf{A}}, g^{\mathbf{y}}$. This follows by presenting the CDH problem as a set of linear equations. If we are given g^a, g^b and we wish to compute g^{ab},[3] we can consider the linear equations (in the exponent) $g^{x_1} = g^a$ and $g^{x_2} = g^{bx_1}$. Given g^{x_1}, g^{x_2} that solve this set of equations, it holds that $g^{x_2} = g^{ab}$. We use random self reducibility to show that solving a set of random equations is also sufficient.

1.3 Paper Organization

Section 2 contains preliminaries and definitions. Section 3 contains our security reductions and constructions for standard signature schemes. In Section 4 we define the notion of ring trapdoor functions and show that it implies ring signatures. The details and proofs of the security reductions for ring signatures are deferred to Appendix A. Section 5 contains our construction of identity based encryption from encryption augmented ring signatures (defined there). In Section 6 we show how to construct ring trapdoor functions and encryption-augmented ring signature schemes based on specific cryptographic assumptions.

[3] Note that for this simplified explanation, g is some fixed generator, which is not the case in the CDH problem where g is random, but this issue can be overcome.

2 Preliminaries

We denote scalars in plain lowercase ($x \in \{0,1\}$), vectors in bold lowercase ($\mathbf{x} \in \{0,1\}^k$) and matrices in bold uppercase ($\mathbf{X} \in \{0,1\}^{k \times k}$). All vectors are column vectors by default, a row vector is denoted $\mathbf{x}^T$. The i^{th} coordinate of $\mathbf{x}$ is denoted x_i.

For a scalar (usually a group element) g and a matrix $\mathbf{X} \in \mathbb{Z}^{k \times n}$ (or a vector, as a special case), we let $g^{\mathbf{X}}$ denote a $k \times n$ matrix such that $(g^{\mathbf{X}})_{i,j} = g^{(\mathbf{X})_{i,j}}$.

Consider a bit-string $x \in \{0,1\}^n$, for some $n \in \mathbb{N}$. We use x_i to denote i^{th} bit of x, and $x_{\leq i}$ to denote the i^{th} prefix of x, i.e. $x_{\leq i} = x_1 \cdots x_i$. We let e_i denote the i^{th} unit string, i.e. $e_i = 0^{i-1}10^{n-i}$. We use $\oplus$ to denote the bitwise XOR operation between strings.

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

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

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

$$\mathrm{Dist}_{X,Y}\mathrm{Adv}[\mathcal{A}] = |\Pr[\mathcal{A}(X_k) = 1] - \Pr[\mathcal{A}(Y_k) = 1]| = \operatorname{negl}(k) \ .$$

We often abbreviate and write $\mathrm{DistAdv}[\mathcal{A}]$ when X, Y are clear from the context.

2.1 Collision Resistance and Chameleon Hash Functions

Collision resistant functions. A family of collision resistant hash functions is a family $\mathcal{H} = \{\mathcal{H}_k\}_{k \in \mathbb{N}}$ of collections $\mathcal{H}_k = \{h : \mathcal{X}_k \to \mathcal{Y}_k\}$ (we sometimes omit the subscript when it is clear from the context), such that

1. There exists a distribution over $\mathcal{H}_k$, which we will slightly abuse notation and denote by $\mathcal{H}_k$ as well, such that it is efficient to sample a description h of a function distributed according to $\mathcal{H}_k$. We denote this process by $h \xleftarrow{\$} \mathcal{H}_k$ and associate the function with its description.

2. Given a description h as described above, it is efficient to compute the associated function on any input. Namely, for all $x \in \mathcal{X}_k$, it is efficient to compute $h(x)$.

3. It is hard to find collisions in a function that was properly sampled. Namely, for any polynomial time adversary $\mathcal{A}$ it holds that

$$\mathrm{Col}_{\mathcal{H}}\mathrm{Adv}[\mathcal{A}] = \Pr_{h \xleftarrow{\$} \mathcal{H}_k}[(h(x_1) = h(x_2)) \wedge (x_1 \neq x_2) : (x_1, x_2) \leftarrow \mathcal{A}(1^k, h)] = \operatorname{negl}(k) \ .$$

Chameleon hash functions. A family of chameleon hash functions [KR00] is a family $\mathcal{H} = \{\mathcal{H}_k\}_{k\in\mathbb{N}}$ of collections of functions $\mathcal{H}_k = \{h : \mathcal{M}_k \times \mathcal{R}_k \to \mathcal{Y}_k\}$ (the subscripts will sometimes be omitted), mapping a message $\mu \in \mathcal{M}_k$ and randomness $r \in \mathcal{R}_k$ to a range $\mathcal{Y}_k$. Intuitively, this is a family of collision resistent hash functions with trapdoor: it is possible to sample a function from the family together with a trapdoor. Given an input to the function, the trapdoor enables to find another input that collides with it. Furthermore, it can even do so when the $\mathcal{M}_k$ part of the new input is given. Formally, we require that the following hold.

1. The family $\mathcal{H}$ is collision resistent as described above, over the domain $\mathcal{X}_k = \mathcal{M}_k \times \mathcal{R}_k$.
2. There exists a distribution over $\mathcal{R}_k$, which we will slightly abuse notation and denote by $\mathcal{R}_k$ as well, such that for all $\mu \in \mathcal{M}_k$, the distributions $(h, h(m, r))$ and (h, y) are statistically indistinguishable, where $h \xleftarrow{\$} \mathcal{H}_k$, $r \xleftarrow{\$} \mathcal{R}_k$ and y is uniform in $\mathcal{Y}_k$.
3. **Chameleon property:** There is an efficient sampling algorithm that outputs a pair (h, h^{-1}) such that
   (a) The marginal distribution of h is statistically indistinguishable from $\mathcal{H}_k$.
   (b) The value h^{-1} is a description of an efficiently computable function (again, we associate the function with its description) such that for all $\mu, \mu' \in \mathcal{M}_k$, $r \in \mathcal{R}_k$ it holds that $h^{-1}(\mu', \mu, r)$ is statistically indistinguishable from the distribution $r' \xleftarrow{\$} \mathcal{R}_k|(h(\mu', r') = h(\mu, r))$.

In this work we require a slightly different (seemingly stronger, though syntactically incomparable) flavor of the chameleon property. Intuitively, we can think of the trapdoor of the standard chameleon hash as a pre-image sampling algorithm, sampling a value r' such that $h(m', r') = y$, which only works given another pre-image (m, r) of y. Our new requirement is that the trapdoor works a little differently: in order to sample a pre-image for y, it may still require some "witness" w, which is related to the way that y was generated, but now w (or rather, the relation between y and w) is "global" and is not specific to one function. Formally, we replace property 3 of the standard definition with the following.

3′. **Chameleon property with witness sampling:**[4] There is an efficient sampling algorithm that outputs a pair (h, h^{-1}) such that
   (a) The marginal distribution of h is statistically indistinguishable from $\mathcal{H}_k$.
   (b) There exists a relation $\mathsf{Valid} \subseteq \mathcal{Y}_k \times \{0,1\}^*$, and an efficient algorithm that samples $(y, w) \in \mathsf{Valid}$ such that the marginal distribution of y is statistically indistinguishable from $U(\mathcal{Y}_k)$. If $(y, w) \in \mathsf{Valid}$ we say that w is a *valid witness* for y.
   (c) The value h^{-1} is a description of an efficiently computable function (again, we associate the function with its description) such that for all $\mu' \in \mathcal{M}$, $y \in \mathcal{Y}_k$ and a valid witness w for y, it holds that $h^{-1}(\mu', y, w)$ is statistically indistinguishable from the distribution $r' \xleftarrow{\$} \mathcal{R}_k|(h(\mu', r') = y)$.

[4]We also call this "the gecko property", see Figure 1 in page 34 for illustration.

(d) The family remains hard to invert, even given a witness. Namely, for any polynomial time $\mathcal{A}$ and for $h \xleftarrow{\$} \mathcal{H}_k$ and (y, w) that are sampled as described above, it holds that

$$\mathrm{Inv}'_{\mathcal{H}}\mathrm{Adv}[\mathcal{A}] = \Pr_{h,(y,w)}[h(\mu, r) = y : (\mu, r) \leftarrow \mathcal{A}(1^k, h, y, w)] = \mathrm{negl}(k) \ .$$

Implementations. We sketch two implementations of chameleon hash functions with witness sampling, that are variants of previous constructions of chameleon hash functions.

- **Based on lattice assumptions.** In the lattice-based chameleon hash function, defined in [GPV08, Pei09], a function is represented by matrices $\mathbf{A} \in \mathbb{Z}_q^{k \times m_1}, \mathbf{B} \in \mathbb{Z}_q^{k \times m_2}$, where k is the security parameter, q is an odd prime and $m_1, m_2 \gg k \log q$ (the exact values of the parameters are related to the lattice reduction, see [Pei09] for details). The message space is $\mathcal{M} = \{\mathbf{x} \in \mathbb{Z}_q^{m_1} : \|\mathbf{x}\|_2 \le \beta_1\}$, the randomness domain is $\mathcal{R} = \{\mathbf{r} \in \mathbb{Z}_q^{m_2} : 0 < \|\mathbf{r}\|_2 \le \beta_2\}$, and the randomness distribution is a discrete Gaussian over $\mathbb{Z}_q^{m_2}$. The range is $\mathcal{Y} = \mathbb{Z}_q^k$. A function in the family is defined by $h_{\mathbf{A},\mathbf{B}}(\mathbf{x}, \mathbf{r}) = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{r}$. The hardness of collision follows from the short integer solution (SIS) assumption, that is related to approximating the short vector problem in lattices (see Section 6.1). The trapdoor is a short basis for the lattice whose parity-check matrix is $\mathbf{B}$.

  In this case h^{-1}, does not require m, r, only the value y. This immediately implies our witness sampling variant, where the empty string (or any other string) can be used as a valid witness. For details on this function, see [GPV08, Pei09].

- **Based on the discrete logarithm assumption.** The discrete logarithm assumption in a cyclic group $\mathbb{G}$ of prime order p, with a canonical generator g for $\mathbb{G}$, is that given a random element $g' \in \mathbb{G}$, it is hard to compute x such that $g' = g^x$.[5] Under this assumption, [KR00] showed how to construct "standard" families of chameleon hash functions. We present a variant with witness sampling defined as follows.

  Let $\mathcal{M} = \mathcal{R} = \mathbb{Z}_p$ and $\mathcal{Y} = \mathbb{G}$. To sample a function in the family, sample $g_1, g_2 \xleftarrow{\$} \mathbb{G}$ and define the function $h_{g_1,g_2}(\mu, r) = g_1^{\mu} \cdot g_2^{r}$. To sample a value-witness pair, sample $w \xleftarrow{\$} \mathbb{Z}_p$ and set $y = g^w$, where g is a canonical generator for $\mathbb{G}$.

  To sample a function in the family together with a trapdoor, we sample $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_p$ and set the description of the function to be $(g_1, g_2) = (g^{x_1}, g^{x_2})$, and the trapdoor to be (x_1, x_2). Given the trapdoor (x_1, x_2) and input $(\mu', w, y = g^w)$, we can find r' such that $x_1\mu' + x_2 r' = w$, and thus output an appropriate pre-image.

  Hardness to invert, even given a witness follows by the following argument. Assume there exists an invertor-with-witness $\mathcal{A}$ for our function family. Given an input $g' = g^x$ for the discrete logarithm problem (where x is unknown), we sample $w \xleftarrow{\$} \mathbb{Z}_p$, set $y = g^w$ and consider the function h described by $(g_1, g_2) = (g', g'^t)$, for $t \xleftarrow{\$} \mathbb{Z}_p$. Given a pre-image μ, r, it holds that $x(\mu + tr) = w \pmod p$, where x is the discrete logarithm of g'. It is easy to extract x from this equation and solve the discrete logarithm problem.

[5] In another flavor of the assumption, g is a random generator and not a canonical one. It is immediate that our assumption is no stronger than this.

2.2 Signature Schemes

A signature scheme is a tuple $\mathcal{S} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ of ppt algorithms such that

- $\mathsf{Gen}(1^k)$ (k is the security parameter) outputs a verification key vk and a signing key sk.
- $\mathsf{Sign}(sk, \mu)$, given a signing key sk and a message $\mu \in \mathcal{M}$, where $\mathcal{M}$ is the *message space* of the scheme, outputs a signature $\sigma \in \{0,1\}^*$.
- $\mathsf{Ver}(vk, \mu, \sigma)$, given a verification key vk, a message μ and a signature σ, either accepts or rejects, we often interpret these as values in $\{0,1\}$.

Correctness. The correctness requirement of a signature scheme is that for any $\mu \in \mathcal{M}$, setting $(vk, sk) \leftarrow \mathsf{Gen}(1^k)$, $\sigma \leftarrow \mathsf{Sign}(sk, \mu)$, it holds that $\mathsf{Ver}(vk, \mu, \sigma)$ accepts with all but negligible probability (over all randomness in the experiment).

Security. In this work, we consider several notions of security for signature schemes, the strongest and most desirable being existential unforgeability under adaptive chosen message attacks (e-cma). We present a generic transformation for converting schemes that adhere to weaker security notions into e-cma-secure signature schemes. All of the security notions that we consider are defined by an interactive experiment conducted between a challenger and a forger. The forger's goal is to *win* in the experiment (the definition of winning is part of the definition of the experiment). For a signature scheme $\mathcal{S}$, a security notion sec, and a forger $\mathcal{F}$, the sec-*advantage* of $\mathcal{F}$, denoted by $\mathrm{Forge}^{\mathrm{sec}}_{\mathcal{S}}\mathrm{Adv}[\mathcal{F}]$, is the probability that $\mathcal{F}$ wins in the sec-experiment. The advantage is a function of the security parameter k. The scheme $\mathcal{S}$ is sec-*secure* if for any polynomial time forger $\mathcal{F}$ it holds that $\mathrm{Forge}^{\mathrm{sec}}_{\mathcal{S}}\mathrm{Adv}[\mathcal{F}] = \mathrm{negl}(k)$.

- **Existential unforgeability under (adaptive) chosen message attack (**e-cma**).** This notion is defined by the following experiment. First, the challenger runs $(vk, sk) \leftarrow \mathsf{Gen}(1^k)$ to obtain a key pair for the signature scheme, and sends vk to the forger. The forger can then make polynomially many (adaptive) queries of the form $\mu_i \in \mathcal{M}$ to which the challenger answers with $\sigma_i \xleftarrow{\$} \mathsf{Sign}(sk, \mu_i)$. Finally the forger outputs (μ^*, σ^*). The forger *wins* if both $\mathsf{Ver}(vk, \mu^*, \sigma^*) = 1$ and $\mu^* \notin \{\mu_i\}$.[6]
- **Existential unforgeability under static chosen message attack (**e-scma**).** In this experiment, the forger first specifies a list of messages $\{\mu_i\}$ and only then the challenger samples (vk, sk), computes $\{\sigma_i\}$ and sends $(vk, \{\sigma_i\})$ to the forger. Finally, the forger sends (μ^*, σ^*). The event of the forger winning in the experiment is defined in the same way. Namely, the forger wins if $\mathsf{Ver}(vk, \mu^*, \sigma^*) = 1$ and $\mu^* \notin \{\mu_i\}$.
- **A-priori-message unforgeability under static chosen message attack (**a-scma**).** In this experiment, the challenger first samples $\mu^* \xleftarrow{\$} \mathcal{M}$ and sends it to the forger. Then the e-scma-experiment is conducted, where the forger is required to sign the message μ^*. Namely, upon receiving a message μ^*, the forger sends $\{\mu_i\}$, the challenger sends $(vk, \{\sigma_i\})$, and the the forger returns σ^*. The forger wins if $\mathsf{Ver}(vk, \mu^*, \sigma^*) = 1$ and $\mu^* \notin \{\mu_i\}$.[7]

[6] We remark that the condition $\mathsf{Ver}(vk, \mu^*, \sigma^*) = 1$ is not well defined if Ver is randomized. To make the definition precise, we have the challenger also sample a random tape with which $\mathsf{Ver}(vk, \mu^*, \sigma^*)$ is executed.

[7] We remark that the *selective-message* variant, where the forger defines μ^* at the beginning of the experiment, rather than having to forge on a random message, is easily reducible to the a-priori case. The reduction will augment a random string to the verification key, and XOR any message with that string prior to signing.

2.3 Ring Signatures

A ring signature scheme is a tuple $\mathcal{R} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ of ppt algorithms such that

- $\mathsf{Gen}(1^k)$ (k is the security parameter) outputs a verification key vk and a signing key sk.
- $\mathsf{Sign}(sk, T, \mu)$, given a signing key sk, a set of verification keys T and a message $\mu \in \mathcal{M}$, where $\mathcal{M}$ is the *message space* of the scheme, outputs a signature $\sigma \in \{0,1\}^*$.
- $\mathsf{Ver}(T, \mu, \sigma)$, given a set of verification keys T, a message μ and a signature σ, either accepts or rejects (we often interpret these as values in $\{0,1\}$).

Correctness. The correctness requirement of a ring signature scheme is that for any $\mu \in \mathcal{M}$ and any polynomial $t \in \mathbb{N}$, setting $(vk_i, sk_i) \leftarrow \mathsf{Gen}(1^k)$ for all $i \in [t]$, $T = \{vk_i\}_{i \in [t]}$, and $\sigma \leftarrow \mathsf{Sign}(sk_1, T, \mu)$, it holds that $\mathsf{Ver}(T, \mu, \sigma)$ accepts with all but negligible probability (over all randomness in the experiment).[8]

Anonymity. The anonymity requirement is that for any $\mu \in \mathcal{M}$ and any polynomial $t \in \mathbb{N}$, setting $(vk_i, sk_i) \leftarrow \mathsf{Gen}(1^k)$ for all $i \in [t]$, $T = \{vk_i\}_{i \in [t]}$, and signatures $\sigma_1 \leftarrow \mathsf{Sign}(sk_1, T, \mu)$, $\sigma_2 \leftarrow \mathsf{Sign}(sk_2, T, \mu)$, it holds that the distributions $(\{sk_i\}, T, \mu, \sigma_1)$ and $(\{sk_i\}, T, \mu, \sigma_2)$ are computationally indistinguishable (where all of our constructions in fact achieve statistical indistinguishability).[9]

Security. Similar to the case of signatures (see Section 2.2), we consider several notions of security for ring signature schemes. The definitions, again, use the notion of challenger-forger games.

- **Existential ring unforgeability under (adaptive) chosen message attack (**er-cma**).** This notion is defined by the following experiment. First, the forger sends a unary value 1^t to the challenger. We assume for simplicity and w.l.o.g. that t is a deterministic function of the security parameter. Then, the challenger runs $(vk_\ell, sk_\ell) \leftarrow \mathsf{Gen}(1^k)$ for all $\ell \in [t]$ to obtain key pairs for the signature scheme, and sends $\{vk_\ell\}_{\ell \in [t]}$ to the forger. The forger can then make polynomially many (adaptive) queries of the form (μ_i, I_i, j_i) where $\mu_i \in \mathcal{M}$, $I_i \subseteq [t]$, $j_i \in I_i$, to which the challenger answers with $\sigma_i \leftarrow \mathsf{Sign}(sk_{j_i}, T_i, \mu_i)$, where $T_i = \{vk_\ell\}_{\ell \in I_i}$. Finally the forger outputs (μ^*, I^*, σ^*).

  We consider two flavors of this attack: *full*-existential unforgeability (fer-cma) and *weak*-existential unforgeability (wer-cma). The two flavors differ in the definition of the forger *winning* event. In both we require that $\mathsf{Ver}(T^*, \mu^*, \sigma^*) = 1$, where $T^* = \{vk_\ell\}_{\ell \in I^*}$, but while for fer-cma we require that $(\mu^*, I^*) \notin \{(\mu_i, I_i)\}$, for wer-cma we require that $\mu^* \notin \{\mu_i\}$.

  By definition, the wer-cma-advantage of any forger is no more than its fer-cma-advantage, thus constructing a fer-cma-secure scheme is at least as hard as constructing a wer-cma-secure one. We show in Appendix A that in fact a wer-cma-secure scheme implies a fer-cma-secure one.

- **Existential ring unforgeability under static chosen message attack (**er-scma**).** This notion is similar to the e-scma notion in standard signatures. The experiment is defined

[8] Notice that the set $T = \{vk_i\}_{i \in [t]}$ is an *unordered* set, and therefore it suffices to require correctness only when signing with sk_1.

[9] As in the correctness requirement, the fact that the set $T = \{vk_i\}_{i \in [t]}$ is an *unordered* set, implies that it suffices to require anonymity only with respect to sk_1 and sk_2.

as follows. The forger first sends 1^t and $\{\mu_i\}$. The challenger sends $\{vk_\ell\}_{\ell\in[t]}$, which are obtained by running $(vk_\ell, sk_\ell) \leftarrow \mathsf{Gen}(1^k)$ for all $\ell \in [t]$. Then the forger can adaptively select (I_i, j_i) and the challenger answers with $\sigma_i \leftarrow \mathsf{Sign}(sk_{j_i}, T_i, \mu_i)$, where $T_i = \{vk_\ell\}_{\ell\in I_i}$. The final message of the forger is (μ^*, I^*, σ^*), and it wins if $\mathsf{Ver}(T^*, \mu^*, \sigma^*) = 1$, where $T^* = \{vk_\ell\}_{\ell\in I^*}$, and $\mu^* \notin \{\mu_i\}$. Note that we only consider the "weak" variant of this notion.

- **A-priori-message ring unforgeability under static chosen message attack (**ar-scma**).** This notion is similar to the a-scma notion in standard signatures. The difference between this and er-scma is that $\mu^* \xleftarrow{\$} \mathcal{M}$ is sampled by the challenger and sent to the forger in the beginning of the experiment. Again, only the weak variant of this notion is considered. Explicitly, in this experiment, the forger first sends 1^t; then the challenger samples μ^*, and sends it to the forger; the forger sends $\{\mu_i\}$; the challenger sends $\{vk_\ell\}_{\ell\in[t]}$; then the forger is allowed to make t queries of the form (I_i, j_i), and the challenger answers with $\sigma_i \leftarrow \mathsf{Sign}(sk_{j_i}, T_i, \mu_i)$; at the end the forger outputs (I^*, σ^*), and it wins if $\mathsf{Ver}(T^*, \mu^*, \sigma^*) = 1$ and $\mu^* \notin \{\mu_i\}$.

2.4 Public-Key Encryption and Identity Based Encryption

2.4.1 Public-Key Encryption

A public-key encryption scheme is a tuple $\mathcal{E} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ of ppt algorithms such that

- $\mathsf{Gen}(1^k)$ (k is the security parameter) outputs a public (encryption) key pk and a secret (decryption) key sk.
- $\mathsf{Enc}(pk, \mu)$, given a public-key pk and a message $\mu \in \mathcal{M}$, where $\mathcal{M}$ is the *message space* of the scheme, outputs a ciphertext $c \in \{0,1\}^*$.
- $\mathsf{Dec}(sk, c)$, given a secret-key sk and a ciphertext c, outputs a message $\mu \in \mathcal{M}$.

Correctness. The correctness requirement for a public-key encryption scheme is that for any $\mu \in \mathcal{M}$, setting $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$, $c \leftarrow \mathsf{Enc}(pk, \mu)$, $\mu' \leftarrow \mathsf{Dec}(sk, c)$, it holds that $\mu' = \mu$ with all but negligible probability (over all randomness in the experiment).

Security. The only security notion for public-key encryption that we discuss in this work is *indistinguishability under chosen plaintext attacks* (CPA), defined by the following interactive experiment played between a challenger and an adversary. The challenger first computes $(pk, sk) \leftarrow \mathsf{Gen}(1^k)$ and sends pk to the adversary. The adversary then chooses two messages m_0 and m_1 from $\mathcal{M}$,[10] and sends m_0, m_1 to the challenger. The challenger flips a coin $b \xleftarrow{\$} \{0,1\}$, computes $c \leftarrow \mathsf{Enc}(pk, m_b)$, and sends c to the adversary. Finally, the adversary answers with a "guess" b', and it *wins* if $b' = b$.

The scheme is CPA-secure if for any polynomial time adversary $\mathcal{A}$ it holds that

$$\mathrm{CPA}_{\mathcal{E}}\mathrm{Adv}[\mathcal{A}] = \left|\Pr[b' = b] - \frac{1}{2}\right| = \mathrm{negl}(k) \ .$$

[10] Our definition has the flavor of "block-by-block" encryption, where the message space is bounded, as will be the case in the schemes we discuss. An alternative definition considers a message space $\mathcal{M} = \{0,1\}^*$ and thus additionally requires that $|m_0| = |m_1|$.

2.4.2 Identity Based Encryption

An identity-based encryption scheme is a tuple $\mathcal{IBE} = (\mathsf{Setup}, \mathsf{Extract}, \mathsf{Enc}, \mathsf{Dec})$ of ppt algorithms such that

- $\mathsf{Setup}(1^k)$ (k is the security parameter) outputs public-parameters pp and master secret-key msk.
- $\mathsf{Extract}(pp, msk, id)$, given public parameters pp, a master secret-key msk and an identity $id \in \mathcal{ID}$, where $\mathcal{ID}$ is *identity space* of the scheme, outputs sk_{id}, a secret-key for identity id.
- $\mathsf{Enc}(pp, id, \mu)$, given public parameters pp, an identity id and a message $\mu \in \mathcal{M}$, where $\mathcal{M}$ is the *message space* of the scheme, outputs a ciphertext $c \in \{0,1\}^*$.
- $\mathsf{Dec}(pp, sk_{id}, c)$, given the public parameters pp, a secret-key sk_{id} and a ciphertext c, outputs a message $\mu \in \mathcal{M}$.

Correctness. The correctness requirement for identity-based encryption is that for any $id \in \mathcal{ID}$ and $\mu \in \mathcal{M}$, setting $(pp, msk) \leftarrow \mathsf{Setup}(1^k)$, $c \leftarrow \mathsf{Enc}(pp, id, \mu)$, $sk_{id} \leftarrow \mathsf{Extract}(pp, msk, id)$, $\mu' \leftarrow \mathsf{Dec}(pp, sk_{id}, c)$, it holds that $\mu' = \mu$ with all but negligible probability (over all randomness in the experiment).

Security. In this work we consider two notions of security, both of which are defined by an interactive experiment played between a challenger and an adversary. In the beginning of the experiment, the challenger flips a hidden coin $b \xleftarrow{\$} \{0,1\}$ before the experiment begins, which affects its behavior in the game. The goal of the adversary is to guess the value of b: its final message is a bit $b' \in \{0,1\}$ which can be interpreted as its guess as to the value of b, it *wins* if $b' = b$. An IBE scheme $\mathcal{IBE}$ is sec-secure if for any polynomial time adversary $\mathcal{A}$ it holds that

$$\mathrm{IBE}^{\mathrm{sec}}_{\mathcal{IBE}}\mathrm{Adv}[\mathcal{A}] = \left|\Pr[b' = b] - \frac{1}{2}\right| = \mathrm{negl}(k) \ .$$

- **Selective-identity (**sel**) security (against chosen plaintext attacks).** Before the experiment, the challenger first flips the hidden coin $b \xleftarrow{\$} \{0,1\}$. The first message is sent by the adversary and contains an identity $id^* \in \mathcal{ID}$. Then the challenger computes $(pp, msk) \leftarrow \mathsf{Setup}(1^k)$ and sends pp to the adversary. The adversary can then make polynomially many (adaptive) queries of the form $id \in \mathcal{ID}$. If $id \neq id^*$, the challenger runs $sk_{id} \leftarrow \mathsf{Extract}(pp, msk, id)$ and sends sk_{id} to the adversary (otherwise it returns $\perp$). After this phase, the adversary computes $m_0, m_1 \in \mathcal{M}$ and sends them to the challenger. The challenger computes $c \leftarrow \mathsf{Enc}(pp, id^*, m_b)$ and sends c to the adversary. The adversary can make additional $id \in \mathcal{ID}$ queries which are answered as before. Finally the adversary outputs its guess b'.
- **Adaptive-identity (**ad**) security (against chosen plaintext attacks).** Before the experiment, the challenger first flips the hidden coin $b \xleftarrow{\$} \{0,1\}$. The challenger then computes $(pp, msk) \leftarrow \mathsf{Setup}(1^k)$ and sends pp to the adversary. The adversary can then make polynomially many (adaptive) queries of the form $id \in \mathcal{ID}$. For each query, the challenger runs $sk_{id} \leftarrow \mathsf{Extract}(pp, msk, id)$ and sends sk_{id} to the adversary. The adversary then computes $id^* \in \mathcal{ID}$ and $m_0, m_1 \in \mathcal{M}$ and sends them to the challenger. If $id^* \notin \{id\}$, where $\{id\}$ is the set of identities that the adversary queried before, then the challenger computes

$c \leftarrow \mathsf{Enc}(pp, id^*, m_b)$ and sends c to the adversary (otherwise it sends $\perp$). The adversary can make additional $id \in \mathcal{ID}$ queries which are answered by $sk_{id} \leftarrow \mathsf{Extract}(pp, msk, id)$ if $id \neq id^*$ (and by $\perp$ otherwise). Finally the adversary outputs its guess b'.

3 A New Perspective on Recent Signature Schemes

In this section, we introduce a general method for amplifying security of signature schemes. More specifically, our method converts any signature scheme that is a-priori-message unforgeable under static chosen message attacks (a-scma) into one that is existentially unforgeable under adaptive chosen message attacks (e-cma). This method uses a family of chameleon hash functions.[11] In fact, we only show how to convert the a-scma-secure scheme into one that is existentially unforgeable under *static* chosen message attack e-scma (this part does not require the chameleon hash), and then use the known reduction from e-cma to e-scma and chameleon hash, presented in [KR00].

We present an overview of our transformation in Section 3.1 and provide the actual construction in Section 3.2. Section 3.3 provides simple constructions of signature schemes based on the above methodology.

3.1 Overview of Our Transformation

As mentioned above, we are given a signature scheme that is a-priori-message unforgeable (under static chosen message attacks), and we wish to transform it into scheme which is existentially unforgeable (under static chosen message attacks). The idea, abstracted from the construction of Hohenberger and Waters [HW09b], is as follows.

The first attempt, is to sign a message μ of length m by considering all m prefixes of μ, and signing each of them separately. The new signature for μ is thus composed of m signatures of the m prefixes. Intuitively, the reason why this should be secure is that when the forger (for the existential attack) specifies the messages he wishes to get signatures for (recall that this is done before seeing the public-key), it actually "commits" to forging a signature for one of polynomially many messages. This is because the message it forges the signature of, must contain a prefix that only differs in the last bit from one of the prefixes of one of the messages it specified, and there are only polynomially many options for that. Therefore, the message that it forges a signature of is determined before the key generation, and thus is "a-priori" in some weak sense.

We would like to claim that our a-priori-message is contained in this polynomial-sized set, and that the forger cannot tell which one it is. In our second attempt, therefore, we add a random string α to the verification-key of the scheme, and require that the prefixes are XORed with α prior to being signed (this can be seen as applying a universal hash function to the prefixes). This enables us to choose α wisely so as to embed the a-priori message as a random element in the said polynomial-sized set.

A final delicate point that we need to address is the following. We explained that we sign by taking all prefixes, XORing them with α and signing each one separately. But α is m bits long, which is significantly longer than many of the prefixes. We, therefore, need to concatenate zeros to the end of the prefix before XORing with α, which leads to the following problem.

[11] Here the original notion of chameleon hash is sufficient and our new flavor is not required, though it is also useable.

Consider a forger that only queries a single message, one whose last bit is 1. Namely, it queries $\mu \in \{0,1\}^m$ such that $\mu_m = 1$. A forgery for $\mu^* = \mu_1 \cdots \mu_{m-1}0$ can be obtained as follows: the forger already has signatures for all prefixes of length $(m-1)$ or less. In addition, the m^{th} prefix (containing the entire message) is identical to the $(m-1)^{\text{th}}$ (since we concatenated a zero bit to the end of the prefix to make it m bit long), so it can just use the same signature for this prefix as well, forging a signature for μ^*. To solve this problem, we will sign not only the string resulting from the XOR operation, but also the length of the prefix this string originated from. Thus preventing this sort of "cut and paste" attacks.

The formal reduction and proof are provided below.

3.2 The Construction and Security Reduction

We present a reduction that given a signature scheme $\mathcal{S} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ that is a-scma-secure, produces a scheme $\mathcal{S}' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ that is e-scma-secure. The message space of $\mathcal{S}'$ is $\{0,1\}^m$, and we require that the message space of $\mathcal{S}$ can be parsed as $\mathcal{M} = [m] \times \{0,1\}^m$. The reduction is defined as follows.

- $\mathsf{Gen}'(1^k)$. Generate $(vk, sk) \leftarrow \mathsf{Gen}(1^k)$ and sample $\alpha \xleftarrow{\$} \{0,1\}^m$. Return the verification key $vk' = (vk, \alpha)$ and the signing key $sk' = sk$.
- $\mathsf{Sign}'(sk', \mu)$. Recall that $sk' = sk$, a signing key for the scheme $\mathcal{S}$, and denote $u^{(i)} = \mu_{\leq i} \| 0^{m-i}$ (recall that $\mu_{\leq i} = \mu_1 \cdots \mu_i$). The signing algorithm computes $\sigma^{(i)} \leftarrow \mathsf{Sign}(sk, (i, u^{(i)} \oplus \alpha))$ for all $i \in [m]$ and outputs $\sigma = \{\sigma^{(i)}\}_{i \in [m]}$.
- $\mathsf{Ver}'(vk', \mu, \sigma)$. Recall that $vk' = (vk, \alpha)$ and parse σ as $\{\sigma^{(i)}\}_{i \in [m]}$. Then Ver' runs $\mathsf{Ver}(vk, (i, u^{(i)} \oplus \alpha), \sigma^{(i)})$ for all $i \in [m]$ and accepts if and only if all of them accepted.

The reduction incurs a factor m overhead in the computational complexity and in the length of the signature. Note, however, that the added computation is parallelizeable so a circuit for Sign' has the same depth as one for Sign, but m times the size.

The completeness of the reduction is immediate. The following theorem states its security properties.

Theorem 3.1. *For any forger $\mathcal{F}'$, there exists a forger $\mathcal{F}$ such that*

$$\mathrm{Forge}_{\mathcal{S}'}^{\text{e-scma}}\mathrm{Adv}[\mathcal{F}'] \leq mq \cdot \mathrm{Forge}_{\mathcal{S}}^{\text{a-scma}}\mathrm{Adv}[\mathcal{F}] \ .$$

Where q is a polynomial upper bound on the number of queries made by $\mathcal{F}'$ in a e-scma *experiment.*

Proof. The forger $\mathcal{F}$ simulates $\mathcal{F}'$ as follows.

1. $\mathcal{F}$ gets a random message $(i^*, \mu^*) \in [m] \times \{0,1\}^m$ from its challenger.
2. $\mathcal{F}$ simulates $\mathcal{F}'$ to obtain the list of messages $\{\mu_j\}_{j \in [q]}$ that $\mathcal{F}'$ wishes to get signatures for, and computes the prefixes $\{u_j^{(i)}\}_{(i,j) \in [m] \times [q]}$.
3. $\mathcal{F}$ samples $j^* \xleftarrow{\$} [q]$ and sets $\alpha = \mu^* \oplus u_{j^*}^{(i^*)} \oplus e_{i^*}$, where $e_{i^*} = 0^{i^*-1}10^{m-i^*}$ is the $i^{*\text{th}}$ unit bit-vector (note that α is uniformly distributed since μ^* is).

   Intuitively, it is here that $\mathcal{F}$ guesses a prefix that $\mathcal{F}'$ is going to sign, and embeds his challenge (i^*, μ^*) in α accordingly.

4. $\mathcal{F}$ sends $\{(i, u_j^{(i)} \oplus \alpha)\}_{(i,j)\in[m]\times[q]}$ to the challenger as the list of messages to be signed. The challenger returns a verification key vk and signatures $\{\sigma_j^{(i)}\}_{(i,j)\in[m]\times[q]}$.

5. $\mathcal{F}$ sends $vk' = (vk, \alpha)$ to $\mathcal{F}'$, along with the signatures $\{\sigma_j^{(i)}\}_{(i,j)\in[m]\times[q]}$ (note that these are distributed exactly as the output of Sign' on $\{\mu_j\}_{j\in[q]}$).

6. When $\mathcal{F}'$ returns $\hat{\mu}, \hat{\sigma} = \{\hat{\sigma}^{(i)}\}_{i\in[m]}$, $\mathcal{F}$ returns $\sigma^* = \hat{\sigma}^{(i^*)}$.

To analyze the performance of $\mathcal{F}$, consider a case where $\mathcal{F}'$ wins in the simulated a-scma experiment. Since $\hat{\mu} \notin \{\mu_j\}_{j\in[q]}$, there exists $i' \in [m]$ such that $\hat{u}^{(i'-1)} \in \{u_j^{(i'-1)}\}_{j\in[q]}$ but $\hat{u}^{(i')} \notin \{u_j^{(i')}\}_{j\in[q]}$. Since the view of $\mathcal{F}'$ is independent of (i^*, j^*), it holds that

$$\Pr[(i^* = i') \wedge (\hat{u}^{(i'-1)} = u_{j^*}^{(i'-1)})] \geq 1/(mq) \ .$$

Consider the case where indeed $i^* = i'$ and $\hat{u}^{(i'-1)} = u_{j^*}^{(i'-1)}$. Note that in such case

$$\hat{u}^{(i^*)} = u_{j^*}^{(i^*)} \oplus e_{i^*} = \mu^* \oplus \alpha.$$

By definition,

$$\mathsf{Ver}(vk, \mu^*, \sigma^*) = \mathsf{Ver}(vk, \hat{u}^{(i^*)} \oplus \alpha, \hat{\sigma}^{(i^*)}) = 1.$$[12]

Further, the fact that $\hat{u}^{(i^*)} \notin \{u_j^{(i^*)}\}_{j\in[q]}$ implies that $(i^*, \mu^*) \notin \{(i, u_j^{(i)} \oplus \alpha)\}_{(i,j)\in[m]\times[q]}$. We conclude that in this case, $\mathcal{F}$ wins in the a-scma experiment, and the result follows. $\square$

3.3 Instantiations

In this section, we show that the reductions presented above capture a significant part of the complication in constructing efficient signature schemes based on standard assumptions. This suggests that the conceptual framework of first constructing an a-scma-secure scheme and then applying our reduction, can be beneficial for future constructions. Our first example shows that, in a sense, the recent scheme of [HW09b] can be interpreted as applying the above framework to the signature scheme of Gennaro, Halevi, and Rabin [GHR99], which results in reducing the required assumption.[13] We also briefly explain how this framework can be employed to obtain signature schemes based on lattice assumptions or based on the CDH assumption over bilinear groups, using ideas from Section 4 as building blocks.

We start with a construction based on the RSA assumption [RSA78]: Let k be the security parameter. Let p, q be uniformly sampled k-bit primes, and define $N = pq$. Let e be a uniformly sampled element in $\mathbb{Z}_N$, where $\varphi(N) = (p-1)(q-1)$. The RSA assumption is that given (N, e) and $g \xleftarrow{\$} \mathbb{Z}_N^*$, it is computationally hard to compute $g^{1/e} \pmod N$.

We next give a high level description of a signature scheme $\mathcal{S} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, which is a-scma-secure under the RSA assumption. We note that this scheme is very similar to the scheme

[12] To be precise, in the case where Ver is probabilistic we need to assume w.l.o.g. that the executions of Ver in both experiments run with the same random tape.

[13] The actual scheme of [HW09b] is much more efficient than the one obtained here, but this is achieved using specific properties of the assumption.

of [GHR99], which is proven there to be e-scma-secure (a stronger notion than our a-scma-security) under the strong-RSA assumption (a stronger assumption than standard RSA that we use here).

One ingredient that we use, and was used in [GHR99, HW09b], is a family of efficiently computable hash functions that map the message space to uniformly distributed prime numbers in $\mathbb{Z}_N^*$.[14] We require that collisions in h (chosen randomly from in the family) do not exist or are hard to find. In addition, we require that given a uniform prime e in the range, it is possible to sample a function h and an input x that are uniformly distributed such that $h(x) = e$. The details of the implementation are not essential to this example and are omitted.[15]

- $\mathsf{Gen}(1^k)$. Generate two random k-bit safe primes $p, q \in \{0,1\}^k$, let $N = pq$, and sample $g \xleftarrow{\$} \mathbb{Z}_N^*$. In addition, sample a hash function h as described above. Set the verification key to be $vk = (N, g, h)$, and the signing key to be $sk = (p, q)$.
- $\mathsf{Sign}(sk, \mu)$. The signing algorithm first computes $e = h(\mu)$. Then, using the factorization of N, the algorithm computes $\sigma = g^{1/e} \pmod N$, and outputs σ as the signature for μ.
- $\mathsf{Ver}(vk, \mu, \sigma)$. Recall that $vk = (N, g, h)$. Accept if and only if $\sigma^e = g \pmod N$, where $e = h(\mu)$.

Completeness is immediate. We next claim that the signature scheme $\mathcal{S} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$, described above, is a-scma secure under the RSA assumption. The proof is very similar to the proof in [GHR99], and we sketch it here for the sake of completeness.

Suppose that there exists a ppt forger $\mathcal{F}$ that succeeds in the a-scma experiment with non-negligible probability. Consider a ppt algorithm $\mathcal{A}$, described below, that makes use of $\mathcal{F}$ and attempts to break the RSA assumption, for the case that the exponent e is a random prime in $\mathbb{Z}_N$.[16] The algorithm $\mathcal{A}$ takes as input a triplet (N, g, e), where N is a product of two random k-bit primes, g is a random element in $\mathbb{Z}_N^*$, and e is a random prime in $\mathbb{Z}_N$, and attempts to compute $g^{1/e} \pmod N$, as follows.

1. Sample h, μ^* such that $h(\mu^*) = e$, and send μ^* as the challenge to the forger $\mathcal{F}$.
2. Upon receiving a list of messages $\mu_1, \ldots, \mu_\ell$ from the forger $\mathcal{F}$, let $e_i = h(\mu_i)$, and use the extended gcd algorithm to compute $a, b \in \mathbb{Z}$ such that $a \cdot e + b \cdot \prod_i e_i = 1$. Let $t = g^{a \cdot e_1 \cdots e_\ell}$, and feed the forger $\mathcal{F}$ with the verification key $vk = (N, t)$, and with signatures $\sigma_1, \ldots, \sigma_\ell$, where
$$\sigma_i = g^{a \prod_{j \neq i} e_j} \pmod N \ .$$
Note that $\sigma_i^{h(\mu_i)} = t \pmod N$, as required.
3. Upon receiving a signature σ^* from $\mathcal{F}$, check whether $\sigma^* = t^{1/e}$. If this is not the case (i.e., if $\mathcal{F}$ failed to forge a signature on μ^*), then abort. Otherwise,
$$\sigma^* = t^{1/e} = g^{\frac{a \cdot e_1 \cdots e_\ell}{e}} = g^{\frac{1-be}{e}} = g^{1/e} g^{-b} \ ,$$

[14] The work of [GHR99, HW09b] used a chameleon hash function for this purpose, however to get security based on the standard RSA assumption (as opposed to the strong RSA assumption), one needs to use a family of deterministic functions, as was done in [HW09b].

[15] As was mentioned above, in [GHR99] the function h is a chameleon hash function; in [HW09b] it is a pseudorandom function, XORed with a random value, with the restriction that the output is prime.

[16] It is sufficient to prove for this case, see [HW09b].

where a, b are the numbers obtained from the extended gcd algorithm above. Output $g^{1/e} = \sigma^* \cdot g^b$.

The following two facts establish the result. First, we note that $\mathcal{A}$ simulates the a-scma experiment in a statistically-close manner; and second that $\mathcal{A}$ succeeds in breaking the RSA assumption whenever the forger $\mathcal{F}$ succeeds in forging a signature in the a-scma experiment simulated by $\mathcal{A}$. We refer the reader to [GHR99] for a more detailed proof.

So far, we presented an a-scma-secure signature scheme based on the RSA assumption, where both the scheme and the proof were relatively simple. We also construct a simple a-scma-secure signature scheme based on the existence of *ring trapdoor functions*, a primitive formally defined in Section 4.1 below. We note that ring trapdoor functions can be easily instantiated based on the lattice-related ISIS assumption and based on the CDH assumption over bilinear groups (see Section 6 for details). Again, both the scheme and the proof are relatively simple. However, we omit the construction here, since in Section 4.3, we construct *ring* signature schemes based on the existence of ring trapdoor functions (without significant complications). The standard signature scheme is very similar (simply taking the ring of verification keys to be the single verification key of the user). We refer the reader to Section 4.3 for details.

4 Ring Signatures in the Standard Model

In this section we show how to construct ring signature schemes in the standard model (namely, without random oracles). Our main tool is a new primitive that we call *ring trapdoor functions* and is presented in Section 4.1. We construct ring signature schemes from ring trapdoor functions in two steps. We first reduce the "ultimate" er-cma security notion to the weak ar-scma notion. This reduction is very similar to the reduction for standard signatures (given in Section 3.2), and is formally stated and proven in Appendix A. Then we construct an ar-scma-secure scheme based on ring trapdoor functions. This construction is formally stated and proven in Section 4.3. In Section 4.2 we provide a high level description of both steps.

4.1 Ring Trapdoor Functions

In this section we define the notion of *ring trapdoor functions*. We defer the instantiations to Section 6, where we show how to construct such a family of ring trapdoor functions under the (inhomogenous) short integer solution (ISIS) assumption, or under the computational Diffie-Hellman assumption in bilinear groups. We refer the reader to Sections 6.1 and 6.2, respectively, for the definitions of these assumptions.

Intuitively, the notion of ring trapdoor functions is a generalization of the standard notion of trapdoor functions: It is not only required that given f, y it is hard to find x such that $f(x) = y$, but it is also hard, given $f_1, \ldots, f_t, y$ to find $x_1, \ldots, x_t$ such that $\sum_{i=1}^{t} f_i(x_i) = y$, for any polynomial t. However, given a trapdoor for *any* of the functions f_i, one can efficiently generate such $x_1, \ldots, x_t$, and furthermore, it is impossible to tell, looking at $x_1, \ldots, x_t$, which of the t trapdoors was used generate them. Note that this definition requires that the range of all the functions can be interpreted as a group, so that the addition is well defined (we also need a neutral element as demonstrated in the formal definition).

In addition, we relax the standard definition of trapdoor functions and only require that given f, x, y, one can efficiently verify that $f(x) = y$, we do not require that f is efficiently computable.

Essentially, the relation of ring trapdoor functions to ring signatures is as follows: think of the function as the verification key and the trapdoor as the signing key; given a set of functions and a target value y, each of the trapdoors can be used to generate (essentially) the same distribution, which is otherwise hard to generate. Of course many details are missing from this intuitive idea, such as the relation to the message to be signed. (More details can be found in Section 4.2, and a formal exposition (construction and proof) can be found in Section 4.3).

Definition 4.1 (ring trapdoor functions)**.** *A family of (one-way)* **ring trapdoor functions** *is a collection of functions $\mathcal{T} = \{\mathcal{T}_k\}_{k\in\mathbb{N}}$, where $\mathcal{T}_k = \{f : X_k \to \mathbb{G}_k\}$, $X = \{X_k\}_{k\in\mathbb{N}}$ is a collection of efficiently recognizable sets, and $\mathbb{G} = \{\mathbb{G}_k\}_{k\in\mathbb{N}}$ is a collection of commutative groups (in this definition we use additive notation for operations in $\mathbb{G}_k$) where group operations can be performed efficiently, such that*

1. **Sampling.** *Given 1^k, one can efficiently sample $f \in \mathcal{T}_k$. We abuse notation and also denote by $\mathcal{T}_k$ the distribution induces by this sampling algorithm.*

   We stress that f may not be efficiently computable.

2. **Zero.** *For every $k \in \mathbb{N}$ there exists an efficiently recognizable element $\xi_k \in X_k$, such that for all $f \in \mathcal{T}_k$ it holds that $f(\xi_k) = 0$. We will use 0 to denote both the identity element of $\mathbb{G}_k$ and ξ_k, so this requirement will be written as $f(0) = 0$. The distinction between the two will be clear from the context.*

3. **Verifiability.** *For every $k \in \mathbb{N}$ and every polynomial t, given any $f_1, \ldots, f_t \in \mathcal{T}_k$, any $x_1, \ldots, x_t \in X_k$, and any $y \in \mathbb{G}_k$, one can efficiently verify that $\sum_{i\in[t]} f_i(x_i) = y$. Note that we do not require that f is efficiently computable.*

4. **Ring one-way.** *For every polynomial t, given $f_1, \ldots, f_t \leftarrow \mathcal{T}_k$ and $y \xleftarrow{\$} \mathbb{G}_k$, it is computationally hard to find $x_1, \ldots, x_t \in X_k$ such that $\sum_{i\in[t]} f_i(x_i) = y$. Formally, we require that for any polynomial time adversary $\mathcal{A}$, any polynomial t, it holds that*

$$\mathrm{RingInv}^t_{\mathcal{T}}\mathrm{Adv}[\mathcal{A}] = \Pr\left[\sum_{i\in[t]} f_i(x_i) = y : \begin{array}{c} f_1, \ldots, f_t \leftarrow \mathcal{T}_k, y \xleftarrow{\$} \mathbb{G}_k \\ (x_1, \ldots, x_t) \leftarrow \mathcal{A}(1^k, f_1, \ldots, f_t, y) \end{array}\right] = \mathrm{negl}(k) \ .$$

5. **Trapdoor.** *Given 1^k one can efficiently sample a function-trapdoor pair (f, td) such that the marginal distribution of f is statistically indistinguishable from $\mathcal{T}_k$ and such that the following holds:*[17] *For any polynomial t, given any $f_1, \ldots, f_t \in \mathcal{T}_k$, together with a trapdoor td_i for f_i, and given any $y \in \mathbb{G}_k$, one can efficiently sample $x_1, \ldots, x_t \in X_k$ such that $\sum_{i\in[t]} f_i(x_i) = y$. Furthermore, using td_j that corresponds to f_j instead of td_i will result in a statistically indistinguishable distribution of $(td_1, \ldots, td_t, x_1, \ldots, x_t)$.*[18]

[17] Note the function-trapdoor sampler can be used to sample functions and satisfy the sampling property above. However, there may exist a more efficient implementation for a function sampler with no trapdoor.

[18] In fact, computational indistinguishability suffices for our purposes, but our implementations have the statistical property which makes the proof slightly simpler.

4.2 Overview of Our Construction

Our construction has two main components. The first is a reduction of er-cma security to ar-scma security (this reduction is formally presented in Appendix A) and the second is a construction of ar-scma-secure ring signatures from ring trapdoor functions (this construction is formally presented in Section 4.3 below).

4.2.1 The Security Reduction

The reduction from er-cma to ar-scma is similar to the respective e-cma to a-scma reduction in "standard" signatures (presented in Section 3.2), but requires attention to a few additional issues.

First of all, we reduce full er-cma security to weak er-cma security. In the former, the forger is successful even if it forges a signature for a message that it queried before, so long as the queried subset of users is different from the set of users it forges on. In the latter, the forger must forge a signature for a message that it did not query before. This reduction is quite straightforward: given a ring signature scheme with weak er-cma security, we create a scheme with full er-cma security by making the signing algorithm sign a tuple containing the message and all verification keys in the user set. Therefore signing a message that was previously queried, without using the same user set, corresponds to signing a new tuple, which is assumed to be hard. This reduction is formalized in Appendix A.1.

We then reduce weak er-cma security to (weak) er-scma-security, where in er-scma, the attack is static; namely the forger needs to specify all of its queries in advance. This is done using (our variant of) chameleon hash functions (see Section 2.1). Given an er-scma secure scheme, we augment the verification key with a chameleon hash function. To sign, we first apply the hash function to the message, using some randomness r, and then sign the output and return the signature and the value of r used. This is secure since we can sample our chameleon hash function along with a trapdoor and generate r in an a-posteriori manner as is done in [KR00] for standard signatures. A delicate point, however, is which hash function should be used in the signing process (recall that the ring signing algorithm takes a number of verification keys, each having its own hash function). One solution is to use one hash function as a "public parameter" for all users, but in ring signatures we usually wish to avoid public parameters (as this requires trusting the entity that generates them). We show that taking any deterministic function of the set of verification keys suffices to achieve security. It is here that we need to rely on our variant of chameleon hash, and cannot simply use the original definition. This reduction is formalized in Appendix A.2.

The final step is going from er-scma security to ar-scma security, in which the challenge message is a-priori determined, and randomly chosen by the challenger. This reduction is very similar to the reduction for the case of standard signature schemes, as presented in Section 3. Namely, the idea is to apply a universal hash function to each prefix of the message and then sign each of these separately. Here, again, there is an issue of which universal hash function to use (in case we wish to sign w.r.t. more than one verification key). This is resolved similarly to the previous reduction, by taking an arbitrary deterministic function of the set of verification keys. This reduction is formalized in Appendix A.3.

4.2.2 Our Construction

We construct ar-scma-secure ring signatures from ring trapdoor functions. This construction resembles the selective-identity IBE constructions of [CHK09, AB09, Pei09]. First, let us consider the simpler case of ar-scma-security for a message space that only contains one message (i.e. $\mathcal{M} = \{0\}$). Note that in this case, there is no query phase. Consider the ring signature scheme whose key generation samples a functions-trapdoor pair (f, td) and an element $y \xleftarrow{\$} \mathbb{G}$ and sets the signing key to be td and the verification key to be (f, y). To sign the message $\mu = 0$ w.r.t. some set of t verification keys, we select one of the y components using an arbitrary (but fixed) deterministic function of the set of verification keys, and use the one trapdoor we have to sample a vector $(x_1, \ldots, x_t)$ such that $\sum f_i(x_i) = y$, and use this vector as the signature. Completeness, anonymity and security follow from the properties of the ring trapdoor function.

Going from message space $\mathcal{M} = \{0\}$ to $\mathcal{M} = \{0,1\}^m$, for a polynomial m, is done as follows. Instead of sampling just one function per user, we sample $2m+1$ functions per user: f_0 and $f_{i,b}$ for $i \in [m]$ and $b \in \{0,1\}$. Each message μ specifies a set of $m+1$ messages, containing f_0 and f_{i,μ_i}. To sign a message μ w.r.t. a set of t verification keys, gather the functions that are relevant to μ from all verification keys, a total of $t \cdot (m+1)$ functions, and sample an input vector corresponding to the output y (where y is determined the same way as before). Note that it is always sufficient to have a trapdoor for f_0 because it is always included in the set, therefore $f_{i,b}$ can be sampled without a trapdoor.[19] Security follows since once the message μ^* is determined, it defines a set of functions for which the forger needs to find appropriate inputs, which would break the ring one-way property. The formal construction and proof can be found in Section 4.3 below.

4.3 Ring Signatures from Ring Trapdoor Functions

Consider the following signature scheme $\mathcal{R}[\mathcal{T}, m]$, whose ar-scma-security we will establish below.

- **Parameters.** The scheme is parameterized by a collection $\mathcal{T} = \{\mathcal{T}_k\}_{k\in\mathbb{N}}$ of ring trapdoor functions and by a polynomial parameter $m = m(k)$. The message space is $\mathcal{M} = \{0,1\}^m$.
- **Key generation.** $\mathsf{Gen}(1^k)$ samples a pair (f_0, td_0) from $\mathcal{T}_k$ and additional $2m$ functions (that can be sampled without a trapdoor) $\langle f_{i,b}\rangle_{i\in[m],b\in\{0,1\}}$ from $\mathcal{T}_k$. It additionally samples $y \xleftarrow{\$} \mathbb{G}_k$. It sets $sk = td_0$ and $vk = (f_0, \langle f_{i,b}\rangle_{(i,b)\in[m]\times\{0,1\}}, y)$.
- **Signing.** $\mathsf{Sign}(sk, T, \mu)$ runs as follows. Recall that T is a set of verification-keys of the form $vk_\ell = (f_0^{(\ell)}, \langle f_{i,b}^{(\ell)}\rangle_{(i,b)\in[m]\times\{0,1\}}, y_\ell)$ for $\ell \in [\tau]$ where $\tau = |T|$. Let $y = y_\ell$ for ℓ for which $f_0^{(\ell)}$ is lexicographically first.[20]

  By the trapdoor property of $\mathcal{T}$, we can use $sk = td_0^{(\ell)}$ (for some $\ell \in [\tau]$) to sample $\langle x_i^{(\ell)}\rangle_{\ell\in[\tau],i\in\{0\}\cup[m]}$ such that
  $$\sum_{\ell\in[\tau]} f_0^\ell(x_0^{(\ell)}) + \sum_{\substack{\ell\in[\tau]\\ i\in[m]}} f_{i,\mu_i}(x_i^{(\ell)}) = y \ .$$

  The signature returned is $\sigma = \langle x_i^{(\ell)}\rangle_{\ell\in[\tau],i\in\{0\}\cup[m]}$.
- **Verifying.** $\mathsf{Ver}(T, \mu, \sigma = \langle x_i^{(\ell)}\rangle_{\ell\in[\tau],i\in\{0\}\cup[m]})$ runs as follows. It automatically rejects if $x_i^{(\ell)} \notin X_k$ for some i, ℓ. Otherwise it takes the value of y from the lexicographically first f_0

[19] We note that the function f_0 is added only for the sake of efficiency. By adding f_0, each user can store only td_0 as its secret key, and always use td_0 to sample the relevant inputs.

[20] This choice of y is arbitrary, any deterministic selection from $\{y_\ell\}_{\ell\in[\tau]}$ will do.

and accepts if and only if

$$\sum_{\ell \in [\tau]} f_0^{\ell}(x_0^{(\ell)}) + \sum_{\substack{\ell \in [\tau] \\ i \in [m]}} f_{i,\mu_i}(x_i^{(\ell)}) = y \ .$$

This can be done efficiently using the verifiability property of $\mathcal{T}$.

Completeness follows from the verifiability property of $\mathcal{T}$. Anonymity follows from the trapdoor property. Security is proven in the following lemma.

Lemma 4.1. *Let $\mathcal{F}$ be a forger for the* ar-scma*-experiment of $\mathcal{R}[\mathcal{T}, m]$. Then there exists an adversary $\mathcal{A}$ such that*

$$\text{Forge}_{\mathcal{R}}^{\text{ar-scma}}\text{Adv}[\mathcal{F}] \leq t \cdot \text{RingInv}_{\mathcal{T}}^{(m+1)t}\text{Adv}[\mathcal{A}] + \text{negl}(k) \ ,$$

Where 1^t is the value of the first message of $\mathcal{F}$ (recall that we assume that this is a deterministic function of k).

Proof. The adversary $\mathcal{A}$ takes as input $(m+1)t$ functions from $\mathcal{T}_k$ and a value $y \in \mathbb{G}_k$. Let us denote these functions by $\hat{f}_{i,\ell}$ for $i \in \{0\} \cup [m]$ and $\ell \in [t]$. The adversary $\mathcal{A}$ runs as follows.

1. $\mathcal{A}$ simulates the first round of $\mathcal{F}$ to obtain 1^t.
2. $\mathcal{A}$ samples $\mu^* \xleftarrow{\$} \{0,1\}^m$ and interprets $\hat{f}_{i,\ell}$ for $\ell \in [t]$ as either $f_0^{(\ell)}$ (if $i = 0$) or $f_{i,\mu_i^*}^{(\ell)}$ (if $i \neq 0$).
3. $\mathcal{A}$ samples $(f_{i,1-\mu_i^*}^{(\ell)}, td_{i,1-\mu_i^*}^{(\ell)})$ for all $(\ell, i) \in [t] \times [m]$. It also samples $\ell^* \xleftarrow{\$} [t]$ and sets $y_{\ell^*} = y$ and $y_\ell \xleftarrow{\$} \mathbb{G}_k$ for all $\ell \neq \ell^*$.
4. $\mathcal{A}$ sends μ^* to $\mathcal{F}$ and gets the list of queries $\{\mu\}$ from $\mathcal{F}$. If μ^* appears in the list then $\mathcal{A}$ halts. Note that in such case $\mathcal{F}$ cannot win in the experiment since success in the ar-scma experiment was defined w.r.t. weak forgery, in which $\mu^* \notin \{\mu_i\}$ (see Section 2.3).
5. $\mathcal{A}$ sends the verification keys $vk_1, \ldots, vk_t$ to $\mathcal{F}$, where $vk_\ell = (f_0^{(\ell)}, \langle f_{i,b}^{(\ell)} \rangle_{(i,b) \in [m] \times \{0,1\}}, y_\ell)$.
6. $\mathcal{A}$ simulates the query phase of $\mathcal{F}$: for a query (I, j) related to a message μ, $\mathcal{A}$ chooses an index i for which $\mu_i \neq \mu_i^*$ and uses $td_{i,1-\mu_i^*}^{(j)}$ to sample $\langle x_i^{(\ell)} \rangle_{\ell \in I, i \in \{0\} \cup [m]}$ such that

$$\sum_{\ell \in I} f_0^{\ell}(x_0^{(\ell)}) + \sum_{\substack{\ell \in I \\ i \in [m]}} f_{i,\mu_i}(x_i^{(\ell)}) = y \ ,$$

   where y is y_ℓ for ℓ with the lexicographically first f_0.
7. At the end, $\mathcal{F}$ outputs $(I^*, \sigma^* = \langle x_i^{*(\ell)} \rangle_{\ell \in I^*, i \in \{0\} \cup [m]})$.
8. $\mathcal{A}$ outputs $\hat{x}_{i,\ell}$ as follows. For all $\ell \in I^*$ and $i \in \{0\} \cup [m]$, set $\hat{x}_{i,\ell} = x_i^{*(\ell)}$. For all other ℓ, i set $\hat{x}_{i,\ell} = 0$.

The above simulates the actual ar-scma-experiment up to a negligible statistical distance, except when $\mu^* \in \{\mu\}$, in which case $\mathcal{F}$ cannot win in the experiment. Consider the case where $\mathcal{F}$ wins in the experiment. Then

$$\begin{aligned}\sum_{\substack{i\in\{0\}\cup[m]\\ \ell\in[t]}} \hat{f}_{i,\ell}(\hat{x}_{i,\ell}) &= \sum_{\substack{i\in\{0\}\cup[m]\\ \ell\in I^*}} \hat{f}_{i,\ell}(\hat{x}_{i,\ell}) + \sum_{\substack{i\in\{0\}\cup[m]\\ \ell\notin I^*}} \underbrace{\hat{f}_{i,\ell}(\hat{x}_{i,\ell})}_{0} \\ &= \sum_{\ell\in I^*} f_0^\ell(x_0^{*(\ell)}) + \sum_{\substack{i\in[m]\\ \ell\in I^*}} f_{i,\mu_i^*}^\ell(x_i^{*(\ell)}) = y' \ ,\end{aligned}$$

where y' is y_ℓ for ℓ with the lexicographically first f_0 (inside I^*). Since the view of $\mathcal{F}$ is independent of ℓ^*, it holds that $y' = y_{\ell^*}$ with probability $1/t$, and the result follows. □

We remark that the scheme in fact adheres to a somewhat stronger security definition: one in which μ^* is defined by the forger (before seeing the verification keys) rather than being randomly selected by the challenger. This requires only a slight modification to the security proof. We further remark that the scheme $\mathcal{R}[\mathcal{T}, m]$ can actually be proven to be er-scma-secure, thus saving the overhead of using the reduction in Appendix A.3. However, this seems to require a significantly more complicated proof, similar to the proof of the static signature scheme in [Pei09]. Our goal in this work is partially to simplify and abstract the ideas in [Pei09].

5 Identity Based Encryption in the Standard Model

In this section, we present a generic method for constructing identity based encryption (IBE) schemes in the standard model (namely, without random oracles), using *encryption-augmented ring signature schemes* - a new notion we define below. The recent constructions of IBE schemes from lattice assumptions [CHK09, AB09, Pei09] can be seen as following this approach. This establishes a connection between ring signature schemes and IBE schemes.

In Section 5.1 we present the new notion (we give instantiations under cryptographic assumptions in Section 6). We give an outline of our selective-identity IBE scheme in Section 5.2 and provide a formal description of the scheme in Section 5.3. In Section 5.4 we explain how to use known techniques from [BB04a, BB04b] to get adaptive-identity IBE.

5.1 Encryption-Augmented Ring Signatures

In what follows, we present the notion of encryption-augmented ring signature schemes. Intuitively, this is a ring signature scheme for which any set of verification keys can be used as a public-key for an encryption scheme, whose secret key is a ring signature of some (arbitrary) message w.r.t. this set.

Definition 5.1 (encryption-augmented ring signatures). *A ring signature scheme $\mathcal{R} = (\mathsf{Gen}_\mathcal{R}, \mathsf{Sign}_\mathcal{R}, \mathsf{Ver}_\mathcal{R})$ is* **encryption-augmented** *if there exists a semantically secure public-key encryption scheme $\mathcal{E}_\mathcal{R}[t] = (\mathsf{Gen}_\mathcal{E}, \mathsf{Enc}_\mathcal{E}, \mathsf{Dec}_\mathcal{E})$ whose key-generation procedure $\mathsf{Gen}_\mathcal{E}(1^k)$ runs as follows: It generates $(vk_i, sk_i) \leftarrow \mathsf{Gen}_\mathcal{R}(1^k)$ for all $i \in [t]$ and sets the public-key to $pk = \{vk_i\}_{i\in[t]}$ and the secret-key to $sk = \sigma \leftarrow \mathsf{Sign}_\mathcal{R}(sk_1, \{vk_i\}_{i\in[t]}, 0)$, where 0 is some (arbitrary) fixed message.*

We note that it is conceivable that an encryption-augmented ring signature scheme has a message space containing only a single message 0.

Remark. Interestingly, we only need to explicitly require the anonymity property from $\mathcal{R}$, we do not even use the verification algorithm $\mathsf{Ver}_{\mathcal{R}}$. he correctness and security of the encryption scheme can be used to define a verification algorithm for which correctness and (some notion of) security hold. Consider $\mathsf{Ver}_{\mathcal{R}}(T, 0, \sigma)$ that verifies a signature by running $r' \leftarrow \mathsf{Dec}_{\mathcal{E}}(\sigma, \mathsf{Enc}_{\mathcal{E}}(T, r))$ and checking whether $r' = r$, for random values of r, and accepting if they all accept. This verification algorithm is obviously correct (by correctness of $\mathcal{E}$) and it is also secure (in some sense) since forging a signature for an unseen message implies generating a proper secret-key for a given public-key (a slightly more complicated argument is required to formally prove this, but for this informal discussion we do not get into more details).

5.2 Overview of Our IBE Scheme

Constructing selective-identity IBE from encryption-augmented ring signatures is quite straightforward. The public parameters are a set of $2d + 1$ verification keys vk_0 and $vk_{i,b}$ for $i \in [d]$ and $b \in \{0, 1\}$, where $\{0, 1\}^d$ is the identity space. The master secret key is the signing key sk_0 that corresponds to vk_0. Each identity id is associated with the subset $\{vk_0\} \cup \{vk_{i,id_i} : i \in [d]\}$ of the verification keys (one can intuitively think of this set as the public key of id). The respective secret-key is generated by using sk_0 to sign the message 0 w.r.t. the set of verification keys $\{vk_0\} \cup \{vk_{i,id_i} : i \in [d]\}$ corresponding to id.

Selective-identity security follows since the adversary commits to the identity id^* before it sees the public-parameters, thus one can generate them in such a way that the verification keys for id^* correspond to the public-key for the encryption scheme $\mathcal{E}_{\mathcal{R}}$, while the secret-keys for all the other public-keys can be computed.

5.3 Identity Based Encryption from Encryption-Augmented Ring Signatures

Let $\mathcal{R} = (\mathsf{Gen}_{\mathcal{R}}, \mathsf{Sign}_{\mathcal{R}}, \mathsf{Ver}_{\mathcal{R}})$ be a ring signature scheme that is augmented with the encryption scheme $\mathcal{E}_{\mathcal{R}}[n] = (\mathsf{Gen}_{\mathcal{E}}, \mathsf{Enc}_{\mathcal{E}}, \mathsf{Dec}_{\mathcal{E}})$. Consider the following scheme $\mathcal{IBE}[\mathcal{R}, d]$ whose identity space is $\{0, 1\}^d$.

- $\mathsf{Setup}(1^k)$. Sample $(vk_0, sk_0) \leftarrow \mathsf{Gen}_{\mathcal{R}}(1^k)$ and additional $2d$ verification keys $\langle vk_{i,b}\rangle_{i\in[d],b\in\{0,1\}}$ for $\mathcal{R}$.[21] Set $msk = sk_0$ and $pp = (vk_0, \langle vk_{i,b}\rangle_{(i,b)\in[d]\times\{0,1\}})$.

- $\mathsf{Extract}(pp, msk, id)$. For $id \in \{0, 1\}^d$, consider the set $vk_{id} = \{vk_0, \{vk_{i,id_i}\}_{i\in[d]}\}$. Use sk_0 to generate $sk_{id} \leftarrow \mathsf{Sign}_{\mathcal{R}}(sk_0, vk_{id}, 0)$. Return sk_{id}, a ring signature of the message 0 w.r.t. the set of verification keys vk_{id}.

- $\mathsf{Enc}(pp, id, \mu)$. Encrypt a message μ by running the encryption algorithm of $\mathcal{E}_{\mathcal{R}}[d + 1]$ with the public-key $pk = vk_{id}$, and output $c \leftarrow \mathsf{Enc}_{\mathcal{E}}(vk_{id}, \mu)$.

- $\mathsf{Dec}(pp, sk_{id}, c)$. Run the decryption algorithm of $\mathcal{E}_{\mathcal{R}}[d + 1]$ with secret-key sk_{id} and ciphertext c. Output $\mu \leftarrow \mathsf{Dec}_{\mathcal{E}}(sk_{id}, c)$.

[21] Similarly to our construction of ring signatures, $vk_{i,b}$ can be generated without a signing key if such process is more efficient.

Correctness follows from the correctness of $\mathcal{E}_{\mathcal{R}}[d+1]$. We next prove the security of $\mathcal{IBE}[\mathcal{R},d]$. In the following lemma we prove selective-identity security. Standard techniques can be used to achieve full security and extend the identity space to $\{0,1\}^*$. See Section 5.4 for details.

Lemma 5.1. *For any adversary $\mathcal{A}$ there exists an adversary $\mathcal{B}$ such that*

$$\mathrm{IBE}_{\mathcal{E}}^{\mathrm{sel}}\mathrm{Adv}[\mathcal{A}] \leq \mathrm{CPA}_{\mathcal{E}_{\mathcal{R}}[d+1]}\mathrm{Adv}[\mathcal{B}] + \mathrm{negl}(k) \ .$$

Proof. Given a public-key $\{\hat{vk}_0, \{\hat{vk}_i\}_{i\in[d]}\}$ for $\mathcal{E}_{\mathcal{R}}[d+1]$, the adversary $\mathcal{B}$ runs as follows.

1. It simulates $\mathcal{A}$ to obtain id^*.
2. It sets $vk_0 = \hat{vk}_0$, $vk_{i,id_i^*} = \hat{vk}_i$, and samples $(vk_{i,1-id_i^*}, sk_{i,1-id_i^*}) \leftarrow \mathsf{Gen}_{\mathcal{R}}(1^k)$ for all $i \in [d]$. It sends the public parameters $pp = (vk_0, \langle vk_{i,b}\rangle_{(i,b)\in[d]\times\{0,1\}})$ to $\mathcal{A}$.
3. For each query $id \neq id^*$ made by $\mathcal{A}$, $\mathcal{B}$ finds $i \in [d]$ for which $id_i \neq id_i^*$ and uses $sk_{i,1-id_i^*}$ to compute $sk_{id} \leftarrow \mathsf{Sign}(sk_{i,id_i}, vk_{id}, 0)$, and returns it to $\mathcal{A}$.
4. When $\mathcal{A}$ decides on messages m_0 and m_1, $\mathcal{B}$ forwards them to the CPA challenger. The challenge ciphertext c is forwarded from the challenger back to $\mathcal{A}$.
5. Additional sk_{id} queries are answered exactly as before.
6. When $\mathcal{A}$ returns a bit b', $\mathcal{B}$ forwards b' to the challenger.

The analysis is straightforward. The distributions of pp and sk_{id} returned by $\mathcal{B}$ are computationally indistinguishable from the real selective-IBE experiment (this follows from the anonymity of the ring signature scheme). Whenever $\mathcal{A}$ wins in the selective-IBE experiment, $\mathcal{B}$ wins in the CPA experiment by definition. The result follows. □

5.4 From Selective Security to Adaptive Security

In the previous section, we presented a construction of selective-identity secure IBE scheme with identity space $\{0,1\}^d$, for a polynomially bounded d. In this section we explain how to use known techniques to overcome two drawbacks of such schemes: selective-identity security and bounded identity space. We explain below how to achieve an adaptive-identity secure scheme using techniques from [BB04a, BB04b]. Going from bounded to unbounded identity space can be done using the standard technique of adding a collision resistent hash function to the public parameters and hashing the identities into the bounded message space; we omit the details.

From here on we focus on achieving adaptive-identity IBE security (see Section 2.4.2 for definition). We discuss two possible methods. The first method, first used in [BB04a], incurs a 2^d decrease in the security guarantee and therefore requires that the original scheme has $2^{-d}\cdot\mathrm{negl}(k)$ security. The second, introduced in [BB04b], does not require changing the assumption, but incurs an efficiency loss in the construction.

- **Exponential reduction.** Since the message space of our scheme is $\{0,1\}^d$, one can simply guess the value of the target identity id^* with probability 2^{-d}, thus enabling to achieve adaptive security with a loss of 2^{-d}. Namely, it holds that $\mathrm{IBE}_{\mathcal{E}}^{\mathrm{ad}}\mathrm{Adv}[\mathcal{A}] \leq 2^d \cdot \mathrm{IBE}_{\mathcal{E}}^{\mathrm{sel}}\mathrm{Adv}[\mathcal{A}]$. Thus if $\mathrm{IBE}_{\mathcal{E}}^{\mathrm{sel}}\mathrm{Adv}[\mathcal{A}] \leq 2^{-d}\cdot\mathrm{negl}(k)$, which in our case depends on the parameters of the ring signature scheme and the augmented encryption scheme, then full security follows. We refer the reader to [BB04a] for more details.

- **Using admissible hash functions.** In [BB04b] a primitive called "admissible biased binary hash function" is introduced, and is shown to exist if collision resistent hash functions exist. This primitive can be used to transform selective-identity secure schemes of certain structure into adaptive-identity secure ones. The required structure is exactly the one our reduction from encryption-augmented ring-signature scheme has: each bit of the identity chooses between two optional "challenges" in the public-parameters, and the "combined challenge" of these d challenges is used for the encryption. Knowing a trapdoor for at least one of the challenges suffices as a trapdoor for the combined challenge, but if no such trapdoor is known then the combined challenge is hard. We refer the reader to [BB04b] for more details.

6 Instantiations

We present constructions of ring trapdoor functions and encryption-augmented ring signature schemes under lattice assumptions and under assumptions in bilinear groups. Specifically, in the lattice domain, we present a family of ring trapdoor functions under a short integer solution assumption (defined below), and show that the resulting ring signature scheme is encryption-augmented under the learning with errors (LWE) assumption. In the bilinear groups domain, we present a family of ring trapdoor functions under the computational Diffie-Hellman assumption in bilinear groups (defined below), and then show how to augment the resulting ring signature scheme with a secure encryption scheme under the d-linear assumption in bilinear groups (for any $d \geq 2$).

6.1 Lattice Assumptions

Let $q \in \mathbb{N}$, $m \in \mathbb{N}$, $\beta \in \mathbb{Z}$ be functions of the security parameter k. The *inhomogenous short integer solution* ($\mathrm{ISIS}_{q,m,\beta}$) assumption is that given $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{k \times m}$, $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^k$, finding $\mathbf{x} \in \mathbb{Z}_q^m$ such that $\mathbf{A} \cdot \mathbf{x} = \mathbf{v}$ and $\|\mathbf{x}\|_2 \leq \beta$ is computationally hard. The *short integer solution* (SIS) assumption is almost identical, only we set $\mathbf{v} \leftarrow \mathbf{0}$, instead of sampling a random vector, and disallow $\mathbf{x} = \mathbf{0}$ as a valid solution. Note that the SIS assumption immediately implies ISIS. These assumptions are both related to worst-case lattice problems. Specifically, if q is an odd prime such that $q \geq \beta \cdot \omega(\sqrt{k \log k})$, then $\mathrm{SIS}_{q,m,\beta}$ and $\mathrm{ISIS}_{q,m,\beta}$, for a polynomial m, are at least as hard as approximating the decision shortest vector problem to within a worst-case factor of $\tilde{O}(\beta\sqrt{k})$ [MR07, GPV08].

In this section we show how to construct ISIS-based ar-scma-secure ring trapdoor functions. In order to obtain full er-cma-security we require our flavor of chameleon hash, which is based on SIS. Thus the resulting ring signature scheme is SIS-based.[22]

6.1.1 Ring Trapdoor Functions

We define a family $\mathcal{T} = \{\mathcal{T}_k\}_{k \in \mathbb{N}}$ of ring trapdoor functions (recall Definition 4.1) as follows.

- **Parameters.** For every $k \in \mathbb{N}$ the construction of $\mathcal{T}_k$ is parameterized by an odd prime modulus $q \in \mathbb{N}$, such that q is super-polynomial in the security parameter k.[23] In addition

[22] In fact, in our range of parameters, the SIS and ISIS assumptions can be shown to be equivalent, so this point is of marginal importance.

[23] The requirement on q is due to our security reduction. We remark that if there is a known polynomial upper bound on the number of functions t for which the ring one-way property needs to hold, then we can use a polynomial value for q (which depends on the bound for t).

we set $m = (5+\delta)k \log q$ (for some real value $\delta > 0$). We also define $s = L \cdot \omega(\sqrt{\log k})$, where $L = O(\sqrt{k \log q})$ is a value taken from previous work.

The function family domain is $X_k = \{\mathbf{x} \in \mathbb{Z}_q^m : \|\mathbf{x}\|_2 \le s \cdot \sqrt{m}\}$ with zero element $\xi = \mathbf{0}^m$ and its range is $\mathbb{Z}_q^k$, with standard vector addition as group operation.

- **Sampling.** Given 1^k, sampling a function in $\mathcal{T}_k$ is done by sampling a random matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{k \times m}$. The function $f_{\mathbf{A}}$ is defined by $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$.

  Sampling a function-trapdoor pair is done using the procedure of [AP09, Theorem 3.2]. They show that for our parameter range, it is efficient to sample $(\mathbf{A}, \mathbf{S}) \in \mathbb{Z}_q^{k \times m} \times \mathbb{Z}_q^{m \times m}$ such that $\mathbf{A}$ is statistically indistinguishable from uniform and such that $\mathbf{S}$ is a trapdoor for $\mathbf{A}$.[24]

  The verifiability property follows immediately since $f_{\mathbf{A}}$ is efficiently computable.

- **Trapdoor property.** In [GPV08, Section 5.3.2] the following is shown. There exists a family of efficiently samplable distributions $\mathcal{X} = \{\mathcal{X}_k\}_{k \in \mathbb{N}}$ (parameterized by q, m, s, all which are functions of k),[25] such that

  1. $\Pr_{\mathbf{x} \leftarrow \mathcal{X}_k}[\|\mathbf{x}\|_2 \le s \cdot \sqrt{m}] = 1 - \mathrm{negl}(k)$.
  2. For any $k \in \mathbb{N}$, any $\mathbf{v} \in \mathbb{Z}_q^k$, and any function trapdoor pair $(\mathbf{A}, \mathbf{S})$, obtained using the sampling algorithm above, one can efficiently sample from the distribution $\mathbf{x} \leftarrow \mathcal{X}$ conditioned on $\mathbf{A} \cdot \mathbf{x} = \mathbf{v}$ (up to a negligible statistical distance).

  Therefore, given any $\mathbf{A}_1, \ldots, \mathbf{A}_t \in \mathbb{Z}_q^{k \times m}$, together with a trapdoor $\mathbf{S}_i$ for $\mathbf{A}_i$, and given any value $\mathbf{v} \in \mathbb{Z}_q^k$, one can sample $\mathbf{x}_j \leftarrow \mathcal{X}$ for all $j \ne i$, and then use $\mathbf{S}_i$ to sample $\mathbf{x}_i$ from (a distribution that is statistically close to) the distribution $\mathcal{X}$ conditioned on $\mathbf{A}_i \cdot \mathbf{x}_i = \mathbf{v} - \sum_{j \ne i} \mathbf{A}_j \cdot \mathbf{x}_j$. Clearly using $\mathbf{S}_j$ for $j \ne i$ will result in a negligibly close distribution.

- **Ring one-way.** The ring one way property is proven via the following lemma.

  Lemma 6.1. *Let $\mathcal{A}$ be a ring invertor for $\mathcal{T}$. Then there exists an adversary $\mathcal{B}$ such that*

  $$\mathrm{RingInv}_{\mathcal{T}}^{t}\mathrm{Adv}[\mathcal{A}] \le \mathrm{ISIS}_{q,tm,s\sqrt{tm}}\mathrm{Adv}[\mathcal{B}] \ .$$

  Recall that solving $\mathrm{ISIS}_{q,tm,s\sqrt{tm}}$ is at least as hard as approximating the (decision) shortest vector problem in the worst case to within a factor of $\tilde{O}(s\sqrt{tm}\sqrt{k})$, namely, a $\mathrm{poly}(k)$ approximation factor.

  Proof. The adversary $\mathcal{B}$ gets as input $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{k \times tm}$ and $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^k$. It parses $\mathbf{A}$ as $\mathbf{A}_1 \| \cdots \| \mathbf{A}_t$ where $\mathbf{A}_i \in \mathbb{Z}_q^{k \times m}$ and sends $\mathbf{A}_1, \ldots, \mathbf{A}_t$ and $\mathbf{v}$ to $\mathcal{A}$. Upon receiving $\mathbf{x}_1, \ldots, \mathbf{x}_t$ from $\mathcal{A}$, it returns $\mathbf{x} = \mathbf{x}_1 \| \cdots \| \mathbf{x}_t$.

  The analysis is straightforward. If $\mathcal{A}$ succeeds then $\|\mathbf{x}\|_2 = \sqrt{\sum_{i \in [t]} \|\mathbf{x}_i\|_2^2} \le s\sqrt{tm}$ and $\mathbf{A} \cdot \mathbf{x} = \sum_{i \in [t]} \mathbf{A}_i \cdot \mathbf{x}_i = \mathbf{v}$. The result follows. $\square$

[24] We hide lattice-related terms that are not necessary to understand our construction, and may unnecessarily complicate our presentation. For the sake of completeness, we mention that the trapdoor $\mathbf{S}$ is a basis for the lattice defined by the parity check matrix $\mathbf{A}$, where the length of the orthogonalization of $\mathbf{S}$ is at most L.

[25] The distribution $\mathcal{X}$ is a "discrete Gaussian" distribution of appropriate parameters; we refer the reader to [GPV08] for details.

6.1.2 Encryption-Augmented Ring Signatures

Consider the ring signature scheme presented in Section 4, instantiated with the family of ring trapdoor functions $\mathcal{T}$ described above. In order to augment this scheme with public-key encryption, we recall the "dual" scheme of [GPV08, Section 7.1], which is based on the learning with errors (LWE) assumption, whose public key is a matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{k\times m'}$ (for some polynomial m') and a value $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^k$ and whose secret key is a vector $\mathbf{x} \in \mathbb{Z}_q^{m'}$ of "small enough" norm such that $\mathbf{A}\cdot\mathbf{x} = \mathbf{v}$. It follows that $\mathcal{R}[\mathcal{T}]$ can be augmented with this scheme (where the vector $\mathbf{v}$ is selected to be $\mathbf{v}_i$ of the lexicographically first verification key). For the details of the scheme and the exact parameters we refer the reader to [GPV08].

6.2 Bilinear Group Assumptions

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

We note that bilinear maps can be defined more generally. Specifically one could consider a mapping $\mathbb{G}\times\mathbb{G}' \to \mathbb{G}_1$ where $\mathbb{G}$, $\mathbb{G}'$ have an efficiently computable isomorphism. Our results extend to this case as well.

Let $\mathbb{G} = \{\mathbb{G}_k\}_{k\in\mathbb{N}}$ be a family of groups, where each group $\mathbb{G}_k$ is of order p, and where p is a k-bit prime. The computational Diffie-Hellman (CDH) assumption on $\mathbb{G} = \{\mathbb{G}_k\}_{k\in\mathbb{N}}$ is that given a random generator h for $\mathbb{G}_k$ and random elements h^a, h^b, it is hard to compute h^{ab}. Namely, for any polynomial time $\mathcal{A}$ it holds that

$$\mathrm{CDHAdv}[\mathcal{A}] = \Pr_{h\xleftarrow{\$}\mathbb{G}_k,a,b\xleftarrow{\$}\mathbb{Z}_p}[\mathcal{A}(1^k, h, h^a, h^b) = h^{ab}] = \mathrm{negl}(k)\ .$$

This is assumed to hold even in some groups with bilinear maps (where $e(h, h)^{ab}$ is easy to compute).

In this section we show how to construct CDH-based ar-scma-secure ring trapdoor functions. In order to obtain full er-cma-security we require our flavor of chameleon hash, based on the discrete logarithm assumption. Since discrete logarithm is implied by CDH, the resulting er-cma-scheme is secure under CDH (in bilinear groups).

The d-linear (dLIN) assumption was first introduced in [BBS04] for $d = 2$ and was later extended to a family of assumptions parameterized by d [Kil07, Sha07]. A matrix form of these assumptions was introduced in [NS09] and it was proven ([NS09, Lemma A.1]) that the matrix form is implied by the standard form. In this work we only refer to the matrix form of this family of assumptions.

Let $\mathbb{G} = \{\mathbb{G}_k\}_{k\in\mathbb{N}}$ be a family of groups as above, and fix an (arbitrary) generator $g = g_k$ in each group $\mathbb{G}_k$ (in what follows we omit the subscript k from g_k to avoid cluttering of notation). The d-linear problem over $\mathbb{G} = \{\mathbb{G}_k\}_{k\in\mathbb{N}}$ is the following: Given $g^{\mathbf{A}}$, where $\mathbf{A} \in \mathbb{Z}_p^{(d+1)\times(d+1)}$, distinguish between the case that $\mathbf{A} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_p^{(d+1)\times(d+1)})$ and the case that $\mathbf{A} \xleftarrow{\$} \mathrm{Rk}_{d+1}(\mathbb{Z}_p^{(d+1)\times(d+1)})$ (where $\mathrm{Rk}_i(S)$ is the set of rank i matrices over the set S). The dLIN *assumption* is that for any polynomial time adversary $\mathcal{A}$ it holds that

$$\mathrm{Lin}_d\mathrm{Adv}[\mathcal{A}] = \left| \Pr_{\mathbf{A}\xleftarrow{\$}\mathrm{Rk}_d(\mathbb{Z}_p^{(d+1)\times(d+1)})}[\mathcal{A}(1^k, g^{\mathbf{A}}) = 1] - \Pr_{\mathbf{A}\xleftarrow{\$}\mathrm{Rk}_{d+1}(\mathbb{Z}_p^{(d+1)\times(d+1)})}[\mathcal{A}(1^k, g^{\mathbf{A}}) = 1] \right| = \mathrm{negl}(k)\ .$$

The 1-linear assumption is identical to the decisional Diffie-Hellman (DDH) assumption, which is false in groups with bilinear maps. It is widely assumed, however, that there exist groups with bilinear maps where the dLIN assumption holds, even for $d = 2$. We further remark that for any polynomial d, the d-linear assumption in a group with bilinear map implies the CDH assumption in that group (i.e. the CDH assumption is no stronger than the dLIN assumption).

We will also use the following simple lemma.

Lemma 6.2. *Consider a finite field $\mathbb{F}$ of order q. For any $n, m \in \mathbb{N}$ such that $m \geq n$, the distance between the distributions $U(\mathbb{F}^{n \times m})$ and $U(\mathrm{Rk}_n(\mathbb{F}^{n \times m}))$ is at most $1/(q^{m-n} \cdot (q-1))$.*

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

6.2.1 Ring Trapdoor Functions

We define a family $\mathcal{T} = \{\mathcal{T}_k\}_{k \in \mathbb{N}}$ of ring trapdoor functions (recall Definition 4.1), as follows.

- **Parameters.** For every $k \in \mathbb{N}$, consider groups $\mathbb{G}$, $\mathbb{G}_1$ of order p, where p is a k-bit prime, with an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$. Fix an arbitrary generator g for $\mathbb{G}$; it follows that $e(g, g)$ is a generator for $\mathbb{G}_1$.

  An additional parameter is a polynomially bounded integer $d \geq 2$, which affects the representation size and computational complexity of the function family. For improved efficiency, we want to take d to be as small as possible. Indeed, for the construction of ring trapdoor functions, $d = 2$ is sufficient to achieve security under the CDH assumption. However, when constructing an encryption-augmented ring signature scheme (in Section 6.2.2) we rely on the d-linear assumption with possibly larger values of d, which results in a (possibly) weaker assumption. We therefore present a family of ring trapdoor functions parameterized by d.

  The function domain is $X = \mathbb{G}^d$, with zero element $\xi = g^{\mathbf{0}}$, and its range is $\mathbb{G}^d$ as well (which forms a group using coordinate-wise group operations).

- **Sampling.** Given 1^k, sampling a function in $\mathcal{T}_k$ is done by sampling a matrix $g^{\mathbf{A}} \xleftarrow{\$} \mathbb{G}^{d \times d}$ (note that $\mathbf{A}$ is not known explicitly). The function $f_{g^{\mathbf{A}}}$ is defined by $f_{g^{\mathbf{A}}}(g^{\mathbf{x}}) = g^{\mathbf{A} \cdot \mathbf{x}}$. We note that this function is hard to compute under the CDH assumption.

  Sampling a function-trapdoor pair is done by sampling $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_p^{d \times d}$ and returning $(f, td) = (g^{\mathbf{A}}, \mathbf{A}^{-1})$ (if $\mathbf{A}$ is not invertible, the sampler fails). Clearly, the marginal distribution of f is statistically close to sampling a uniform function in $\mathcal{T}_k$, e.g. by Lemma 6.2.

  The verifiability property follows using the bilinear mapping. Given $g, g^{\mathbf{A}}, g^{\mathbf{x}}, g^{\mathbf{v}}$, one can efficiently compute $e(g, g)^{\mathbf{A} \cdot \mathbf{x}}$ and check whether the result is equal to $e(g, g)^{\mathbf{v}}$.

- **Trapdoor property.** Given any $g^{\mathbf{A}_1}, \ldots, g^{\mathbf{A}_t}$, together with a trapdoor $\mathbf{A}_i^{-1}$, and given any $g^{\mathbf{v}}$, we show how to sample $g^{\mathbf{x}_1}, \ldots, g^{\mathbf{x}_t}$ such that $\sum_{i \in [t]} \mathbf{A}_i \cdot \mathbf{x}_i = \mathbf{v}$, thus proving the trapdoor property. The sampler will sample $\mathbf{x}_j \xleftarrow{\$} \mathbb{Z}_q^d$ for all $j \neq i$ and then set $g^{\mathbf{x}_i} = g^{\mathbf{A}_i^{-1} \cdot (\mathbf{v} - \sum_{j \neq i} \mathbf{A}_j \cdot \mathbf{x}_j)}$. Note that this is efficiently computable since $\mathbf{A}_i^{-1}$ and $\mathbf{x}_j$ are explicitly known.

The resulting distribution of $g^{\mathbf{x}_1}, \ldots, g^{\mathbf{x}_t}$ is independent of i, up to a negligible statistical distance. This can be seen by considering the case where all $\mathbf{A}_i$'s are invertible, in which case the vectors $\mathbf{x}_1, \ldots, \mathbf{x}_t$ are uniformly distributed in the solution space of $\sum_{i \in [t]} \mathbf{A}_i \cdot \mathbf{x}_i = \mathbf{v}$.

- **Ring one-way.** The ring one-way property is proven by the following lemma.

Lemma 6.3. *Let $\mathcal{A}$ be a ring invertor for $\mathcal{T}$. Then if $d = 2$ then there exists an adversary $\mathcal{B}$ such that*

$$\left(\mathrm{RingInv}^t_{\mathcal{T}}\mathrm{Adv}[\mathcal{A}]\right)^2 \leq \mathrm{CDHAdv}[\mathcal{B}] + O(1/p) \ .$$

and if $d \geq 3$ then there exists an adversary $\mathcal{B}$ such that

$$\mathrm{RingInv}^t_{\mathcal{T}}\mathrm{Adv}[\mathcal{A}] \leq \mathrm{CDHAdv}[\mathcal{B}] + O(1/p) \ .$$

Proof. We recall that the adversary $\mathcal{A}$ gets as input $g^{\mathbf{A}}, g^{\mathbf{v}}$ where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_p^{d \times td}$, $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_p^d$ and outputs (when successful) $g^{\mathbf{x}} \in \mathbb{G}^{td}$ such that $\mathbf{A} \cdot \mathbf{x} = \mathbf{v}$.

Given an input h, h^a, h^b for CDH, let $r \in \mathbb{Z}_p$ be such that $h = g^r$ (recall that g is a known canonical generator and h is a uniformly selected generator); we stress that r is not explicitly known. Using this notation, we can write the input as (g^r, g^{ra}, g^{rb}) and the required output is $h^{ab} = g^{rab}$.

We start by giving a proof for the case $d = 3$, which immediately extends to all $d \geq 3$. The case of $d = 2$ requires a slightly more delicate treatment, and is addressed last.

For $d = 3$, consider

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \\ -rb & 1 & 0 \\ 0 & 1 & -r \end{bmatrix} \in \mathbb{Z}_p^{3 \times 3} \text{ and } \mathbf{z} = \begin{bmatrix} ra \\ 0 \\ 0 \end{bmatrix} \in \mathbb{Z}_p^3 \ .$$

Note that $g^{\mathbf{C}}, g^{\mathbf{z}}$ are explicitly known. In addition, since $\mathbf{C}$ is invertible ($r \neq 0$ since h is a generator), there exists a unique $\mathbf{x} \in \mathbb{Z}_p^3$ such that $\mathbf{C} \cdot \mathbf{x} = \mathbf{z}$, specifically this vector is

$$\mathbf{x} = \mathbf{C}^{-1} \cdot \mathbf{z} = [ra, r^2ab, rab]^T.$$

If our ring trapdoor function invertor $\mathcal{A}$ was guaranteed to find $g^{\mathbf{x}}$ for *any* $g^{\mathbf{A}}, g^{\mathbf{v}}$, then we would be done, since we could have fed it with $g^{\mathbf{C}}, g^{\mathbf{z}}$ and find g^{rab} as required. However, this will not work since $\mathbf{C}, \mathbf{z}$ are not uniformly distributed and since $\mathbf{C}$ does not have the right dimensions ($\mathcal{A}$ expects $g^{\mathbf{A}} \in \mathbb{G}^{3 \times 3t}$).

To overcome this, the CDH adversary $\mathcal{B}$ runs as follows. It samples $\mathbf{L} \xleftarrow{\$} \mathrm{Rk}_3(\mathbb{Z}_p^{3 \times 3})$, $\mathbf{R} \xleftarrow{\$} \mathrm{Rk}_3(\mathbb{Z}_p^{3 \times 3t})$. It sets $g^{\mathbf{A}} = g^{\mathbf{L} \cdot \mathbf{C} \cdot \mathbf{R}}$ and $g^{\mathbf{v}} = g^{\mathbf{L} \cdot \mathbf{z}}$ and applies $\mathcal{A}$ to $g^{\mathbf{A}}, g^{\mathbf{v}}$ to obtain $g^{\mathbf{x}}$. If $\mathcal{A}$ succeeded, then $\mathbf{A} \cdot \mathbf{x} = \mathbf{v}$, i.e. $\mathbf{L} \cdot \mathbf{C} \cdot \mathbf{R} \cdot \mathbf{x} = \mathbf{L} \cdot \mathbf{z}$ and thus $\mathbf{R} \cdot \mathbf{x} = [ra, r^2ab, rab]^T$ and $g^{\mathbf{e_3} \cdot \mathbf{R} \cdot \mathbf{x}} = g^{rab}$ as required.

This approach can be extended to $d > 3$ in a straightforward manner, we define

$$\mathbf{C}' = \left[\begin{array}{c|c} \mathbf{C} & \mathbf{0}^{3 \times (d-3)} \\ \hline \mathbf{0}^{(d-3) \times 3} & \mathbf{I}_{(d-3)} \end{array}\right] \text{ and } \mathbf{z}' = \left[\begin{array}{c} \mathbf{z} \\ \hline \mathbf{0}^{(d-3)} \end{array}\right]$$

so that $\mathbf{C}'^{-1} \cdot \mathbf{z}' = [ra, r^2ab, rab, 0, \ldots, 0]^T$. The adversary $\mathcal{B}$ will sample $\mathbf{L} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_p^{d \times d})$, $\mathbf{R} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_p^{d \times dt})$ and proceed exactly as before.

Since the matrix $\mathbf{A}$ and vector $\mathbf{v}$ are (jointly) $O(1/p)$-uniform by Lemma 6.2, then the success probability of $\mathcal{A}$ on these inputs is at least $\text{RingInv}_{\mathcal{T}}^{t}\text{Adv}[\mathcal{A}] - O(1/p)$ and the claim (for $d \geq 3$) follows.

For $d = 2$, we take a similar approach, but this time we need a two-step process. We define

$$\mathbf{C} = \begin{bmatrix} 1 & 0 \\ -rb & 1 \end{bmatrix} \in \mathbb{Z}_p^{2\times 2} \text{ and } \mathbf{z} = \begin{bmatrix} ra \\ 0 \end{bmatrix} \in \mathbb{Z}_p^2 ,$$

and note that $\mathbf{C}^{-1} \cdot \mathbf{z} = [ra, r^2ab]^T$. As above, $\mathcal{B}$ randomizes $\mathbf{C}$ and $\mathbf{z}$, by sampling $\mathbf{L} \xleftarrow{\$} \text{Rk}_2(\mathbb{Z}_p^{2\times 2})$, $\mathbf{R} \xleftarrow{\$} \text{Rk}_2(\mathbb{Z}_p^{2\times 2t})$, and applies $\mathcal{A}$ to $g^{\mathbf{A}} = g^{\mathbf{L}\cdot\mathbf{C}\cdot\mathbf{R}}$ and $g^{\mathbf{v}} = g^{\mathbf{L}\cdot\mathbf{z}}$ to obtain $g^{\mathbf{x}}$. In this case, if $\mathcal{A}$ succeeds then $g^{\mathbf{e_2}^T\cdot\mathbf{R}\cdot\mathbf{x}} = g^{r^2ab}$.

Next $\mathcal{B}$ considers

$$\mathbf{D} = \begin{bmatrix} 1 & 0 \\ 1 & -r \end{bmatrix} \in \mathbb{Z}_p^{2\times 2} \text{ and } \mathbf{w} = \begin{bmatrix} r^2ab \\ 0 \end{bmatrix} \in \mathbb{Z}_p^2$$

Conditioned on $\mathcal{A}$'s success previously, $g^{\mathbf{D}}$, $g^{\mathbf{w}}$ are explicitly known. As before, $\mathcal{B}$ samples $\mathbf{L}' \xleftarrow{\$} \text{Rk}_2(\mathbb{Z}_p^{2\times 2})$, $\mathbf{R}' \xleftarrow{\$} \text{Rk}_2(\mathbb{Z}_p^{2\times 2t})$ and applies $\mathcal{A}$ to $g^{\mathbf{A}} = g^{\mathbf{L}'\cdot\mathbf{D}\cdot\mathbf{R}'}$ and $g^{\mathbf{v}} = g^{\mathbf{L}'\cdot\mathbf{w}}$ to obtain $g^{\mathbf{x}'}$. Again, if $\mathcal{A}$ is successful then $\mathbf{L}' \cdot \mathbf{D} \cdot \mathbf{R}' \cdot \mathbf{x}' = \mathbf{L}' \cdot \mathbf{w}$. Using the same reasoning above, if $\mathcal{A}$ is successful then $g^{\mathbf{e_2}^T\cdot\mathbf{x}'} = g^{rab}$ as required.

Here, again, the inputs to $\mathcal{A}$ in both times are (jointly) $O(1/p)$-uniform, and therefore the probability that $\mathcal{A}$ succeeds in both times is at least $\left(\text{RingInv}_{\mathcal{T}}^{t}\text{Adv}[\mathcal{A}]\right)^2 - O(1/p)$ and the result follows. $\square$

6.2.2 Encryption-Augmented Ring Signatures

Consider the following public-key encryption scheme $\mathcal{E}$, which is described, for the sake of simplicity, as a bit-encryption scheme (a scheme whose message space contains only one bit), but can be extended to multiple bit messages.

- **Parameters.** For every security parameter $k \in \mathbb{N}$, consider groups $\mathbb{G}$, $\mathbb{G}_1$ of order p, where p is a k-bit prime, with an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_1$. Fix an arbitrary generator g for $\mathbb{G}$. Fix additional parameters $d \geq 2$ and t, both which are polynomially bounded.

- **Key generation.** Sample $g^{\mathbf{A}} \xleftarrow{\$} \mathbb{G}^{d\times dt}$, $\mathbf{x} \xleftarrow{\$} \mathbb{Z}_p^{dt}$, compute $g^{\mathbf{v}} = g^{\mathbf{A}\cdot\mathbf{x}}$ and output $pk = (g^{\mathbf{A}}, g^{\mathbf{v}})$, $sk = g^{\mathbf{x}}$.

- **Encryption.** To encrypt a bit b, sample $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^d$ and output ciphertext $c = (g^{\mathbf{r}^T\cdot\mathbf{A}}, h \cdot g^{\mathbf{r}^T\cdot\mathbf{v}})$, where $h = g^b$.

- **Decryption.** On ciphertext $c = (g^{\mathbf{w}^T}, g^z)$, compute $e(g,g)^{z-\mathbf{w}^T\cdot\mathbf{x}}$. Output the bit b such that $e(g,g)^{z-\mathbf{w}^T\cdot\mathbf{x}} = e(g,g)^b$.

Correctness follows since for a proper ciphertext $e(g,g)^{z-\mathbf{w}^T\cdot\mathbf{x}} = e(h,g)$. Let us give an informal explanation on how the above scheme can be extended to a key-encapsulation mechanism of approximately $\log p$ bits. The encryption algorithm will sample h uniformly from $\mathbb{G}$ and in addition it will sample a seed s for a randomness extractor. The new ciphertext will be $c = (g^{\mathbf{r}^T\cdot\mathbf{A}}, h \cdot g^{\mathbf{r}^T\cdot\mathbf{v}}, s)$

and the encapsulated key will be $\mathsf{ext}_s(e(h,g))$. Clearly this value can be retrieved in the decryption. The same security proof (see below) works in this case as well.

Consider the ring signature scheme presented in Section 4, instantiated with the family of ring trapdoor functions $\mathcal{T}$ described above (in Section 6.2.1). If $\mathcal{E}$ is indeed secure, then it can trivially be augmented to the ring signature scheme (for the value of $\mathbf{v}$ just use the "lexicographically first" $\mathbf{v}_i$).

The proof of CPA security follows.

Lemma 6.4. *Consider a* CPA *adversary* $\mathcal{A}$ *for* $\mathcal{E}$*, there exists an adversary* $\mathcal{B}$ *for* dLIN *such that*

$$\mathrm{CPA}_{\mathcal{E}}\mathrm{Adv}[\mathcal{A}] \le \mathrm{Lin}_d\mathrm{Adv}[\mathcal{B}] + O(1/p)\ .$$

Proof. Given a matrix $g^{\mathbf{M}}$ such that $\mathbf{M} \xleftarrow{\$} \mathrm{Rk}_{d'}(\mathbb{Z}_p^{(d+1)\times(dt+1)})$, $d' \in \{d, d+1\}$ (note that distinguishing in this case is no harder than in the $(d+1)\times(d+1)$ case). Define $\mathbf{A} \in \mathbb{Z}_p^{d\times dt}$, $\mathbf{v} \in \mathbb{Z}_p^d$, $\mathbf{w} \in \mathbb{Z}_p^{dt}$, $z \in \mathbb{Z}_p$ such that

$$\left[\begin{array}{c|c} \mathbf{A} & \mathbf{v} \\ \hline \mathbf{w}^T & z \end{array}\right] = \mathbf{M}\ .$$

The adversary $\mathcal{B}$ simulates the CPA game for $\mathcal{A}$. First it sends $pk = (g^{\mathbf{A}}, g^{\mathbf{v}})$ and when $\mathcal{A}$ sends messages m_0, m_1, it computes the corresponding h_0, h_1, then flips a coin $b \xleftarrow{\$} \{0,1\}$ and sends $c = (g^{\mathbf{w}}, h_b \cdot g^z)$ to $\mathcal{A}$. When $\mathcal{A}$ returns b', $\mathcal{B}$ returns 1 if and only if $b' = b$.

We prove by a series of hybrids (experiments).

- **Hybrid** H_0**.** In this hybrid we run $\mathcal{A}$ with $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_p^{d\times dt}$, $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_p^d$ and $\mathbf{w}^T = \mathbf{r}^T \cdot \mathbf{A}$, $z = \mathbf{r}^T \cdot \mathbf{v}$ for $\mathbf{r} \xleftarrow{\$} \mathbb{Z}_p^d$. Then $\mathrm{CPA}_{\mathcal{E}}\mathrm{Adv}[\mathcal{A}] = \left|\Pr_{H_0}[b' = b] - \frac{1}{2}\right|$.

- **Hybrid** H_1**.** We now change the distribution of $\mathbf{M}$ (which is composed of $\mathbf{A}, \mathbf{v}, \mathbf{w}, z$) and let $\mathbf{M} \xleftarrow{\$} \mathrm{Rk}_d(\mathbb{Z}_p^{(d+1)\times(dt+1)})$. It holds that $|\Pr_{H_1}[b' = b] - \Pr_{H_0}[b' = b]| \le O(1/p)$.

- **Hybrid** H_2**.** We now let $\mathbf{M} \xleftarrow{\$} \mathrm{Rk}_{d+1}(\mathbb{Z}_p^{(d+1)\times(dn+1)})$. By definition $|\Pr_{H_2}[b' = b] - \Pr_{H_1}[b' = b]| = \mathrm{Lin}_d\mathrm{Adv}[\mathcal{B}]$.

- **Hybrid** H_3**.** We now let $\mathbf{M} \xleftarrow{\$} \mathbb{Z}_p^{(d+1)\times(dt+1)}$. Then $|\Pr_{H_3}[b' = b] - \Pr_{H_2}[b' = b]| \le O(1/p)$. In addition, in this hybrid $\mathcal{A}$'s view is independent of b and thus $\Pr_{H_3}[b' = b] = 1/2$.

Combining the above, we conclude that

$$\mathrm{CPA}_{\mathcal{E}}\mathrm{Adv}[\mathcal{A}] \le \mathrm{Lin}_d\mathrm{Adv}[\mathcal{B}] + O(1/p),$$

as desired.

□

References

[AB09] Shweta Agrawal and Xavier Boyen. Identity-based encryption from lattices in the standard model. Manuscript, July 2009. Available at `http://www.cs.stanford.edu/~xb/ab09/`.

[AP09] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. In Susanne Albers and Jean-Yves Marion, editors, *STACS*, volume 3 of *LIPIcs*, pages 75–86. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009.

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In Cachin and Camenisch [CC04], pages 223–238.

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In Franklin [Fra04], pages 443–459.

[BB04c] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In Cachin and Camenisch [CC04], pages 56–73.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Franklin [Fra04], pages 41–55.

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

[CC04] Christian Cachin and Jan Camenisch, editors. *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*. Springer, 2004.

[CHK07] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. *J. Cryptology*, 20(3):265–294, 2007. Preliminary version in Eurocrypt 2003.

[CHK09] David Cash, Dennis Hofheinz, and Eike Kiltz. How to delegate a lattice basis. Cryptology ePrint Archive, Report 2009/351, 2009. `http://eprint.iacr.org/`.

[CL04] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Franklin [Fra04], pages 56–72.

[CS00] Ronald Cramer and Victor Shoup. Signature schemes based on the strong rsa assumption. *ACM Trans. Inf. Syst. Secur.*, 3(3):161–185, 2000.

[CvH91] David Chaum and Eugène van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265, 1991.

[Fra04] Matthew K. Franklin, editor. *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*. Springer, 2004.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer, 1986.

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

[GHR99] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In *EUROCRYPT*, pages 123–139, 1999.

[GJKW07] Eu-Jin Goh, Stanislaw Jarecki, Jonathan Katz, and Nan Wang. Efficient signature schemes with tight reductions to the diffie-hellman problems. *J. Cryptology*, 20(4):493–514, 2007.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *STOC*, pages 197–206. ACM, 2008.

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

[HW09a] Susan Hohenberger and Brent Waters. Realizing hash-and-sign signatures under standard assumptions. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 333–350. Springer, 2009.

[HW09b] Susan Hohenberger and Brent Waters. Short and stateless signatures from the rsa assumption. In Halevi [Hal09], pages 654–670.

[Kil07] Eike Kiltz. Chosen-ciphertext secure key-encapsulation based on gap hashed diffie-hellman. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 282–297. Springer, 2007.

[KR00] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *NDSS*. The Internet Society, 2000.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

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

[NY89] Moni Naor and Moti Yung. Universal one-way hash functions and their cryptographic applications. In *STOC*, pages 33–43. ACM, 1989.

[Oka92] Tatsuaki Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest F. Brickell, editor, *CRYPTO*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53. Springer, 1992.

[Pei09] Chris Peikert. Bonsai trees (or, arboriculture in lattice-based cryptography). Cryptology ePrint Archive, Report 2009/359, 2009. `http://eprint.iacr.org/`.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *EUROCRYPT*, pages 387–398, 1996.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOC*, pages 387–394. ACM, 1990.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In Colin Boyd, editor, *ASIACRYPT*, volume 2248 of *Lecture Notes in Computer Science*, pages 552–565. Springer, 2001.

[RST06] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to leak a secret: Theory and applications of ring signatures. In *Essays in Theoretical Computer Science: in Memory of Shimon Even, volume 3895 of LNCS Festschrift*. Springer-Verlag, 2006.

[Sch91] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991.

[Sha84] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.

[Sha07] Hovav Shacham. A Cramer-Shoup encryption scheme from the Linear Assumption and from progressively weaker Linear variants. Cryptology ePrint Archive, Report 2007/074, February 2007. `http://eprint.iacr.org/`.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer, 2005.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In Halevi [Hal09], pages 619–636.

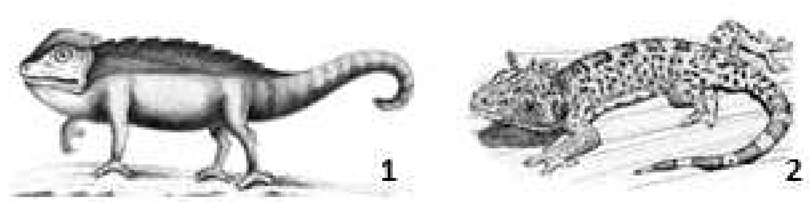

Figure 1: Illustrations of a Chameleon (1) and a Gecko (2).

A Security Reductions for Ring Signatures

We present the formal reductions between notions of security of ring signatures and their proof. For overview of all steps below, see Section 4.2.

A.1 From Weak Unforgeability to Full Unforgeability

We show that weak unforgeability (wer-cma-security) is sufficient in order to obtain full fer-cma-security (see Section 2.3 for definitions).

Let $\mathcal{R} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a wer-cma-secure ring signature scheme with message space $\mathcal{M} = \{0,1\}^*$. We construct a scheme $\mathcal{R}' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ with message space $\mathcal{M}' = \{0,1\}^*$ as follows.

- $\mathsf{Gen}'(1^k)$ runs $(vk, sk) \leftarrow \mathsf{Gen}(1^k)$ and outputs (vk, sk) (namely, the key generation doesn't change).
- $\mathsf{Sign}'(sk, T, \mu)$ runs $\sigma \leftarrow \mathsf{Sign}(sk, T, (T, \mu))$ and outputs σ.
- $\mathsf{Ver}'(T, \mu, \sigma)$ runs $\mathsf{Ver}(T, (T, \mu), \sigma)$ and outputs the result.

The correctness and anonymity of $\mathcal{T}'$ follow immediately from those of $\mathcal{T}$. The security reduction follows.

Lemma A.1. *Let $\mathcal{F}'$ be a forger for the* fer-cma-*experiment, there exists a forger $\mathcal{F}$ for the* wer-cma-*experiment such that*

$$\mathrm{Forge}_{\mathcal{R}'}^{\text{er-cma}}\mathrm{Adv}[\mathcal{F}'] \leq \mathrm{Forge}_{\mathcal{R}}^{\text{wer-cma}}\mathrm{Adv}[\mathcal{F}] \ .$$

Proof. Given access to $\mathcal{F}'$, the forger $\mathcal{F}$ runs as follows.

1. $\mathcal{F}$ obtains value 1^t from $\mathcal{F}'$ and send it to the challenger.
2. Upon receiving the set $\{vk_i\}_{i\in[t]}$ from the challenger, $\mathcal{F}$ forwards this set to $\mathcal{F}'$.
3. Simulate the queries of $\mathcal{F}'$: when $\mathcal{F}'$ makes a (μ, I, j), $\mathcal{F}$ sends the query $((\mu, T), I, j)$, where $T = \{vk_\ell\}_{\ell\in I}$, to the challenger and forwards its response σ back to $\mathcal{F}'$.
4. When $\mathcal{F}'$ returns a forgery (μ^*, I^*, σ^*), $\mathcal{F}$ returns $((\mu^*, T^*), I^*, \sigma^*)$, where $T^* = \{vk_\ell\}_{\ell\in I^*}$.

By definition, if $\mathcal{F}'$ wins then $(\mu^*, I^*) \notin \{(\mu, I)\}$, which implies that $(\mu^*, T^*) \notin \{(\mu, T)\}$. Furthermore, $\mathsf{Ver}'(\mu^*, T^*, \sigma^*) = \mathsf{Ver}((\mu^*, T^*), T^*, \sigma^*) = 1$. Thus $\mathcal{F}$ wins as well. □

A.2 From Static to Adaptive Chosen Message Attack

We now show that using chameleon hash functions, we can use an er-scma-secure ring signature scheme to construct an er-cma-secure one. The reduction is quite similar to the one presented in [KR00] in the context of standard signatures.

Let $\mathcal{H} \subseteq \mathcal{M} \times \mathcal{Q} \to \mathcal{Y}$ be a family of chameleon hash functions with witness sampling, as defined in Section 2.1. Let $\mathcal{R} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be a er-scma-secure ring signature scheme with message space $\mathcal{Y}$. Fix a deterministic and efficiently computable mapping φ such that for any set of verification keys $T = \{vk_\ell\}$, $vk^* \leftarrow \varphi(T)$ is such that $vk^* \in T$. We somewhat abuse notation and write $\ell \leftarrow \varphi(T)$ where the elements of T are assumed to be indexed in some way. We note that one simple instantiation of φ is taking the lexicographically first element in T.

We construct a scheme $\mathcal{R}' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ with message space $\mathcal{M}$ as follows.

- $\mathsf{Gen}'(1^k)$ runs $(vk, sk) \leftarrow \mathsf{Gen}(1^k)$ and also samples $h \xleftarrow{\$} \mathcal{H}$. It sets $vk' = (vk, h)$ and $sk' = sk$ and outputs (vk', sk').
- $\mathsf{Sign}'(sk, T, \mu)$ first computes $\ell \leftarrow \varphi(T)$. It then samples $r \xleftarrow{\$} \mathcal{R}$ and computes $y \leftarrow h_\ell(\mu, r)$. It then runs $\sigma \leftarrow \mathsf{Sign}(sk, T, y)$ and outputs $\sigma' = (r, \sigma)$.
- $\mathsf{Ver}'(T, \mu, \sigma')$ computes $\ell = \varphi(T)$. It parses σ' as (r, σ) and computes $y \leftarrow h_\ell(\mu, r)$. It then returns $\mathsf{Ver}(T, y, \sigma)$.

The correctness and anonymity properties of $\mathcal{T}'$ follow in a straightforward manner from those of $\mathcal{T}$.

Lemma A.2. *Let $\mathcal{F}'$ be a forger for the* er-cma-*experiment, there exists a forger $\mathcal{F}$ for the* er-scma-*experiment and adversaries $\mathcal{A}$, $\mathcal{B}$ for $\mathcal{H}$ such that*

$$\mathrm{Forge}^{\text{er-cma}}_{\mathcal{R}'}\mathrm{Adv}[\mathcal{F}'] \leq \mathrm{Forge}^{\text{er-scma}}_{\mathcal{R}}\mathrm{Adv}[\mathcal{F}] + t \cdot \mathrm{Col}_{\mathcal{H}}\mathrm{Adv}[\mathcal{A}] + qt \cdot \mathrm{Inv}'_{\mathcal{H}}\mathrm{Adv}[\mathcal{B}] \ ,$$

where 1^t is $\mathcal{F}'$'s first message and q is a polynomial upper bound on the number of queries $\mathcal{F}'$ makes.

Proof. The forger $\mathcal{F}$ is defined as follows.

1. $\mathcal{F}$ gets the value 1^t from $\mathcal{F}'$ and sends it to the challenger.
2. $\mathcal{F}$ samples (h_ℓ, h_ℓ^{-1}) from $\mathcal{H}$ for all $\ell \in [t]$.
3. $\mathcal{F}$ samples value-witness pairs (y_i, w_i) for all $i \in [q]$. It sends $\{y_i\}_{i\in[q]}$ to the challenger.
4. The challenger sends $\{vk_\ell\}_{\ell\in[t]}$ to $\mathcal{F}$, that sets $vk'_\ell = (vk_\ell, h_\ell)$ and sends $\{vk'_\ell\}_{\ell\in[t]}$ to $\mathcal{F}'$.
5. Upon receiving (μ_i, I_i, j_i) from $\mathcal{F}'$, $\mathcal{F}$ forwards (I_i, j_i) to the challenger and gets a signature $\sigma_i = \mathsf{Sign}(sk_{j_i}, T_i, y_i)$ in response, where $T_i = \{vk'_\ell\}_{\ell\in I_i}$. $\mathcal{F}$ computes $\ell_i \leftarrow \varphi(T_i)$ and samples $r_i \leftarrow h_{\ell_i}^{-1}(\mu_i, y_i, w_i)$. Then $\mathcal{F}$ returns $\sigma'_i = (r_i, \sigma_i)$ to $\mathcal{F}'$.
6. When $\mathcal{F}'$ concludes and returns (μ^*, I^*, σ'), $\mathcal{F}$ parses $\sigma' = (r^*, \sigma^*)$, it finds $T^* = \{vk'_\ell\}_{\ell\in I^*}$ and computes $\ell^* \leftarrow \varphi(T^*)$ and $y^* = h_{\ell^*}(\mu^*, r^*)$. It eventually outputs (y^*, I^*, σ^*).

Consider a case where $\mathcal{F}'$ wins in the experiment. If $y^* \notin \{y_i\}_{i\in[q]}$, then $\mathcal{F}$ described above also wins. Consider the case where $\mathcal{F}'$ wins and $y^* \in \{y_i\}_{i\in[q]}$. First, consider the case where $y^* = y_i$ for some $i \in [q]$ and in addition $\ell^* = \ell_i$, this will enable us to find a collision in h_{ℓ^*} using the adversary $\mathcal{A}$ defined below. Otherwise, in the case where $y^* = y_i$ for some $i \in [q]$ but $\ell^* \neq \ell_i$, we will be able to invert h_{ℓ^*} using the adversary $\mathcal{B}$ defined below.

The collision finding adversary $\mathcal{A}$ for the case $(y^* = y_i) \wedge (\ell^* = \ell_i)$ is defined as follows. On input h, $\mathcal{A}$ simulates the er-cma experiment for $\mathcal{F}'$ as follows. It simulates the first message to obtain 1^t and then samples $\hat{\ell} \xleftarrow{\$} [t]$. It then sets $h_{\hat{\ell}} = h$ and samples all other values according to their intended distributions. During the execution, $\mathcal{A}$ saves all values (μ, r, y) such that $y_i = h_{\hat{\ell}}(\mu, r)$ that is computed in order to answer the messages μ (note that it only needs to save the values on which it executed the input function $h = h_{\hat{\ell}}$). Upon receiving (μ^*, I^*, σ'), it checks whether $\ell^* = \hat{\ell}$ and in such case it searches the list for an entry (μ, r, y) with $y = y^*$. If such is found, $\mathcal{A}$ outputs (μ^*, r^*) and (μ, r). Recall that since $\mathcal{F}'$ wins in the experiment, then $\mu^* \neq \mu$ and thus the collision is not trivial.

Note that the view of $\mathcal{F}'$ in the simulation is independent of $\hat{\ell}$, therefore in the case where $\mathcal{F}'$ wins in the experiment and $(y^* = y_i) \wedge (\ell^* = \ell_i)$ for some $i \in [q]$, $\mathcal{A}$ wins with probability $1/t$.

The inverting adversary $\mathcal{B}$ for the case $(y^* = y_i) \wedge (\ell^* \neq \ell_i)$ is defined as follows. On inputs h, y, w, $\mathcal{B}$ simulates the er-cma experiment for $\mathcal{F}'$. It gets 1^t and samples $\hat{\ell}$ as $\mathcal{A}$ does, but samples all other h_i's together with h_i^{-1}. Furthermore it samples $\hat{i} \xleftarrow{\$} [q]$. For all queries except for the $\hat{i}^{\text{th}}$, $\mathcal{B}$ simulates the experiment as prescribed. In the $\hat{i}^{\text{th}}$ it checks whether $\ell_{\hat{i}} = \hat{\ell}$. If this is indeed the case, then this query is again simulated as prescribed. If $\ell_{\hat{i}} \neq \hat{\ell}$, however, then $\mathcal{B}$ sets $y_{\hat{i}} = y$ and samples $r_{\hat{i}} \leftarrow h_{\ell_{\hat{i}}}^{-1}(\mu_{\hat{i}}, y, w)$, and uses these to sign. In the end of the simulation, $\mathcal{B}$ checks if $\ell^* = \hat{\ell}$ and if $h(\mu^*, r^*) = y$, if this is the case, it returns (μ^*, r^*).

The view of $\mathcal{F}'$ is independent of $\hat{\ell}, \hat{i}$ and therefore if $\mathcal{F}'$ wins and $(y^* = y_i) \wedge (\ell^* \neq \ell_i)$ for some $i \in [q]$, then $\mathcal{B}$ succeeds in inverting with probability at least $1/(qt)$. $\square$

Note that the transformation in Appendix A.1 assumed that the scheme $\mathcal{R}$ we start with has message space $\{0,1\}^*$. On the other hand, the reduction in this section may produce schemes with bounded message space (depending on the domain of the family $\mathcal{H}$). This gap can be bridged by using a family of collision resistent hash functions using a "hash and sign" paradigm. We omit the details.

A.3 From Existential Unforgeability to A-Priori-Message Unforgeability

We now show that similar to the reduction in Section 3.2, we can use an ar-scma-secure ring signature scheme to construct an er-scma-secure one.

Let $\mathcal{R} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Ver})$ be an ar-scma-secure signature scheme with message space $\mathcal{M} = [m] \times \{0,1\}^m$. We consider a mapping φ as in Section A.2 above, and construct a scheme $\mathcal{R}' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Ver}')$ with message space $\mathcal{M}' = \{0,1\}^m$ as follows.

- $\mathsf{Gen}'(1^k)$. Generate $(vk, sk) \leftarrow \mathsf{Gen}(1^k)$ and sample $\alpha \xleftarrow{\$} \{0,1\}^m$. Return the verification key $vk' = (vk, \alpha)$ and the signing key $sk' = sk$.
- $\mathsf{Sign}'(sk', T, \mu)$. Recall that $sk' = sk$, a signing key for the scheme $\mathcal{R}$, and denote $u^{(s)} = \mu_{\leq s} \| 0^{m-s}$. Let $\ell \leftarrow \varphi(T)$. The signing algorithm computes $\sigma^{(s)} \leftarrow \mathsf{Sign}(sk, T, (s, u^{(s)} \oplus \alpha_\ell))$ for all $s \in [m]$, and outputs $\sigma = \{\sigma^{(s)}\}_{s \in [m]}$.
- $\mathsf{Ver}'(T, \mu, \sigma)$. Let $\ell \leftarrow \varphi(T)$ and parse σ as $\{\sigma^{(s)}\}_{s \in [m]}$. Then Ver' runs $\mathsf{Ver}(T, (s, u^{(s)} \oplus \alpha_\ell), \sigma^{(s)})$ for all $s \in [m]$, and accepts if and only if all of them accepted.

Correctness and anonymity of $\mathcal{R}'$ follow immediately from those of $\mathcal{R}$. Security is proven in the following lemma.

Lemma A.3. *For any forger $\mathcal{F}'$, there exists a forger $\mathcal{F}$ such that*

$$\text{Forge}_{\mathcal{S}'}^{\text{er-scma}}\text{Adv}[\mathcal{F}'] \leq mqt \cdot \text{Forge}_{\mathcal{S}}^{\text{ar-scma}}\text{Adv}[\mathcal{F}] + qt \cdot 2^{-m} \ .$$

Where q is a polynomial upper bound on the number of queries made by $\mathcal{F}'$ and 1^t is the value of $\mathcal{F}'$'s first message.

Proof. The forger $\mathcal{F}$ simulates $\mathcal{F}'$ as follows.

1. $\mathcal{F}$ simulates $\mathcal{F}'$ to obtain the value 1^t and forwards it to the challenger.

2. $\mathcal{F}$ gets a random message $(s^*, \mu^*) \in [m] \times \{0,1\}^m$ from its challenger.

3. $\mathcal{F}$ simulates $\mathcal{F}'$ to obtain the list of messages $\{\mu_i\}_{i\in[q]}$ that $\mathcal{F}'$ wants to get signatures for, and computes the prefixes $\{u_i^{(s)}\}_{(s,i)\in[m]\times[q]}$.

4. $\mathcal{F}$ samples $\hat{i} \xleftarrow{\$} [q]$, $\hat{\ell} \xleftarrow{\$} [t]$ and sets $\alpha_{\hat{\ell}} = \mu^* \oplus u_{\hat{i}}^{(s^*)} \oplus e_{s^*}$ (note that $\alpha_{\hat{\ell}}$ is uniformly distributed since μ^* is). For all $\ell \in [t] \setminus \{\hat{\ell}\}$, it samples $\alpha_\ell \xleftarrow{\$} \{0,1\}^m$.

5. $\mathcal{F}$ sends $\{(s, u_i^{(s)} \oplus \alpha_\ell)\}_{(i,s,\ell)\in[q]\times[m]\times[t]}$ to the challenger as the list of messages to be signed. Note that we XOR all prefixes with all possible αs.

6. The challenger sends the verification keys $\{vk_\ell\}_{\ell\in[t]}$. $\mathcal{F}$ sends $\{vk'_\ell = (vk_\ell, \alpha_\ell)\}_{\ell\in[t]}$ to $\mathcal{F}'$.

7. When $\mathcal{F}'$ makes a query (I_i, j_i), $\mathcal{F}$ makes the following $m \cdot t$ queries to the challenger: for each message $\{(s, u_i^{(s)} \oplus \alpha_\ell)\}_{(s,\ell)\in[m]\times[t]}$ (note that i is fixed at this point), $\mathcal{F}$ sends the query (j_i, I_i) and receives $\{\sigma_{i,\ell}^{(s)}\}_{s,\ell}$. Then $\mathcal{F}$ computes $\ell_i = \varphi(T_i)$ and sends $\sigma'_i = \{\sigma_{i,\ell_i}^{(s)}\}_{s\in[m]}$ as a response to $\mathcal{F}'$.

8. When $\mathcal{F}'$ returns $(\mu', I', \sigma' = \{\sigma'^{(s)}\}_{s\in[m]})$, $\mathcal{F}$ returns $(I^* = I', \sigma^* = \sigma'^{(s^*)})$.

The analysis is similar to that of Theorem 3.1. Consider a case where $\mathcal{F}'$ wins in the simulated experiment. Since $\mu' \notin \{\mu_i\}_{i\in[q]}$, there exists $s' \in [m]$ such that $u'^{(s'-1)} \in \{u_i^{(s'-1)}\}_{i\in[q]}$ but $u'^{(s')} \notin \{u_i^{(s')}\}_{i\in[q]}$. Since the view of $\mathcal{F}'$ is independent of $(s^*, \hat{i}, \hat{\ell})$, it holds that

$$\Pr[(\varphi(T^*) = \hat{\ell}) \wedge (s' = s^*) \wedge (u'^{(s'-1)} = u_{\hat{i}}^{(s'-1)})] \geq 1/(mqt) \ .$$

Consider the case where the above occurs, note that in such case $u'^{(s^*)} = u_{\hat{i}}^{(s^*)} \oplus e_{\hat{i}} = \mu^* \oplus \alpha_{\hat{\ell}}$. Since $\mathsf{Ver}'(T', \mu', \{\sigma'^{(s)}\}_{s\in[m]}) = 1$, then, by definition, also $\mathsf{Ver}(T', (s^*, \mu^*), \sigma'^{(s^*)}) = 1$. We need to make sure that $\mathcal{F}$ didn't "accidentally" make the query (s^*, μ^*) in the query phase. Recall that the set of queries that $\mathcal{F}$ makes is $\{(s, u_i^{(s)} \oplus \alpha_\ell)\}_{(i,s,\ell)\in[q]\times[m]\times[t]}$. For the queries where $s \neq s^*$, clearly there is no risk. For the set of queries where $s = s^*$ but $\ell = \hat{\ell}$, it holds that $u'^{(s^*)} \notin \{u_i^{(s^*)}\}_{i\in[q]}$ and therefore $(s^*, \mu^*) \notin \{(s^*, u_i^{(s^*)} \oplus \alpha_{\hat{\ell}})\}_{i\in[q]}$. It is left to check what happens when $s = s^*$ and $\ell \neq \hat{\ell}$, i.e. whether $(s^*, \mu^*) \in \{(s, u_i^{(s)} \oplus \alpha_\ell)\}_{(s,i)\in[m]\times[q]}$ for such ℓ. Since all such α_ℓ are sampled uniformly and independently of $u_i^{(s)}$, then for all $\ell \neq \hat{\ell}$, $i \in [q]$ it holds that $\Pr[(s^*, \mu^*) = (s^*, u_i^{(s^*)} \oplus \alpha_\ell)] = 2^{-m}$. Applying the union bound, the result follows. $\square$